Babylonian Journal of Machine Learning
Vol.2026, **pp**. 274–312
DOI: https://doi.org/10.58496/BJML/2026/017; ISSN: 3006-5429
https://mesopotamian.press/journals/index.php/BJML

Research Article

An Explainable Ai Framework for Multi-Class Attack Classification in Mqtt-Coap-Based Hybrid Energy Monitoring Networks

Ahmed Fakhri Abdullah[1,*], Omar Ahmed Razooqi[2], Oras Fadhil Khalaf[3], Ayman AbuBaker[4], Amer Alqutaish[5], Saidah Khan[6]

[1] *Ministry of Higher Education and Scientific Research, 10011, Baghdad, Iraq*

[2] *Hadith and its Sciences Department, Islamic Sciences College, Tikrit University, Tikrit, Iraq*

[3] *Computer Science Department, College of Applied Sciences, University of Samarra, Samarra 34010, Iraq*

[4] *Image Processing and Machine Learning, Faculty of Engineering and Technology, Applied Science Private University, Amman, Jordan*

[5] *Deanship of Development and Quality Assurance, King Faisal University, Al-Ahsa, 31982, Saudi Arabia*

[6] *Faculty of Computing, Griffith College, Dublin, Ireland*

ARTICLE INFO

Article History

Received 22 Jul 2026
Revised 27 Aug 2026
Accepted 25 Sep 2026
Published 07 Nov 2026

Keywords

MQTT,

CoAP,

False data injection,

Intrusion detection,

Hybrid renewable energy systems,

Explainable AI,

SHAP,

Evaluation leakage.

CC BY

ABSTRACT

Hybrid renewable-energy microgrids communicate through lightweight MQTT and CoAP links, exposing network traffic and sensor measurements to cyberattacks. This study presents an explainable framework for multi-class attack classification using reproducible message-level co-simulation of a photovoltaic–wind–diesel–battery microgrid. The simulation incorporates exact power balance, eleven sensors, six attack families, and five categories of challenging benign events. The dataset contains 86,400 thirty-second windows spanning 30 days, including 878 attack bursts. Sixty-eight features combine traffic statistics with energy-consistency residuals calibrated during a clean commissioning interval. Evaluation uses chronological partitioning, a purge gap, and block-bootstrap confidence intervals. Among seven classifiers, boosted trees with a five-window lag achieve a macro-F1 of 0.913 (95% confidence interval: 0.893–0.927), Matthews correlation coefficient of 0.907, accuracy of 0.978, and macro-averaged ROC-AUC of 0.974 on seven unseen days. Random partitioning inflates false-data-injection F1 by up to 13 percentage points. All bursts across five network-visible attack classes are detected, with class F1 scores of 0.93–0.98 and a 1.2% false-alarm rate on normal windows. However, window-level detection rates for false-data injection fall to 81% for bias attacks, 65% for ramp attacks, and at most 10% for energy-balance-preserving attacks. Halving attack intensity reduces macro-F1 by five percentage points, while transfer to a second site produces a twelve-point reduction before local adaptation. TreeSHAP explanations demonstrate faithfulness and stability, with deletion AUC of 0.22 versus 0.51 for random rankings, top-five overlap of 0.89, and approximately one-millisecond computation per window. These simulation-based findings highlight both classification effectiveness and limitations under stealthy attacks and site changes; code and data are released for replication.

1. INTRODUCTION

Hybrid renewable energy systems combine photovoltaic (PV) arrays, wind turbines, battery storage and a diesel genset behind a single point of control, and they are increasingly deployed in remote and islanded locations where grid reinforcement is uneconomic. Their supervisory layer depends on dense telemetry: every inverter, turbine controller, meter and weather sensor reports to a gateway several times per minute, and the control centre uses these reports to dispatch storage and backup generation. In practice that telemetry is carried by two lightweight application protocols: MQTT over TCP [1] and CoAP over UDP [2]. Both were designed for constrained devices, and neither authenticates the origin of a reading or protects its integrity unless the operator adds transport security and key management that many field deployments omit.

Three characteristics make this setting different from enterprise intrusion detection. First, an attack can act on two layers at once. A flood, an amplification abuse of an exposed CoAP endpoint, or a reconnaissance sweep is visible in the message

*Corresponding author. Email: alpasha_online@yahoo.com

stream, whereas a false-data injection attack (FDIA) [3, 4] leaves traffic statistics untouched and shows up only as a physically implausible measurement. Second, the benign background is unusually hostile to a classifier: sensor faults, link outages that are flushed in bursts, route changes and client reconnects all resemble attacks on some channel. Third, an operator must act on an alarm, so a bare class label is not enough; the decision has to be explained in terms that map onto equipment and protocol behaviour.

A growing literature applies machine learning and, increasingly, explainable AI (XAI) to this problem. We identify four weaknesses in how the evidence is usually produced. (i) *Evaluation leakage*: windows cut from the same attack burst are shuffled into training and test partitions, so a classifier can be rewarded for recognising a burst it has already seen [5, 6]. (ii) *Protocol confounding*: when each attack class is generated on a single transport, a model can separate classes by protocol identity rather than by attack behaviour. (iii) *Unverified explanations*: SHAP and LIME plots are presented as evidence of trustworthiness although attribution methods can be unstable, mutually inconsistent or manipulable [7–9]. (iv) *Missing physics*: network features alone cannot see an FDIA that was designed to be consistent with the grid's own measurement model [3].

This paper addresses these weaknesses with an end-to-end, fully reproducible study of multi-class attack classification in MQTT–CoAP monitoring networks. We make the following contributions.

1. **A message-level co-simulation.** We build a hybrid PV–wind–diesel–battery microgrid whose power balance is enforced exactly, instrument it with eleven sensors, and generate the resulting MQTT and CoAP message streams with protocol-level behaviour (retransmission back-off, duplicate message identifiers, time-stamp skew, TTL and round-trip baselines). Six attack families are injected, each on both transports where that is meaningful, together with five classes of benign hard negatives (Section 4).
2. **A physics-aware feature set.** We combine 45 traffic features with 21 energy-consistency features and two time-of-day features, calibrated only on a clean commissioning interval, including robust balance residuals, a heteroscedastic EWMA innovation, a CUSUM statistic, peer comparison and multi-scale signed-bias aggregation (Section 5.1).
3. **An evaluation protocol that controls leakage.** Training, validation and test data are separated chronologically with a purge gap, uncertainty is estimated by a block bootstrap, models are compared with a Friedman–Nemenyi test and Holm-corrected pairwise tests, and we quantify how much a conventional random split would have inflated the result (Section 5.3).
4. **Stress tests that go beyond accuracy.** We evaluate leave-one-variant-out generalisation across three FDIA variants, evasion by attack-intensity scaling in the problem space [10], cross-site transfer with few-shot adaptation, imbalance handling, and a leave-one-group-out feature ablation.
5. **A measured assessment of the explanations.** We compare TreeSHAP [11], KernelSHAP [12], our own tabular LIME [13] and Integrated Gradients [14] by deletion/insertion faithfulness, stability under input noise, cross-method agreement and group-level attribution, and we report the end-to-end latency of detection plus explanation.

The results are deliberately less flattering than those reported in much of the literature, and we regard that as a feature of the protocol. The best detector reaches a macro-F1 of 0.913 on seven held-out days; the stealthy balanced FDIA variant is practically invisible to every model we examined (at most 10 % of its windows are flagged), and benign sensor faults are the main source of false alarms. We discuss what these limits imply for deployment in Section 8.

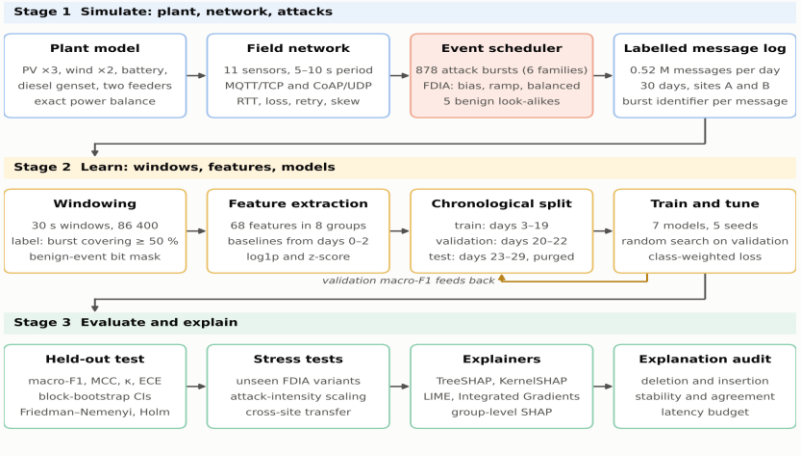


Fig. 1. Overview of the study. Stage 1 simulates the plant, the field network and the attack and benign events and records a labelled message log. Stage 2 turns the log into 30 s windows and 68 features, splits the days chronologically and tunes seven models on the validation days. Stage 3 evaluates the models on held-out days, stresses them and audits four explanation methods.

The remainder of the paper is organised as follows. Section 2 reviews related work. Section 3 states the system and threat model. Section 4 gives the mathematical model of the plant, the measurement and message processes, and the attacks. Section 5 describes the features, models, evaluation protocol and explanation methods. Section 6 documents the experimental set-up, Section 7 reports the results, Section 8 discusses their implications and limitations, and Section 9 concludes.

2. RELATED WORK

The literature that bears on this study falls into four groups: intrusion-detection data sets for IoT protocols, the methodology of evaluating learning-based detectors, the detection of false-data injection in power systems, and the evaluation of explanations in security. Table 1 summarises the closest work, and the discussion below states what each group does and does not establish for MQTT–CoAP monitoring of an energy system.

Intrusion-detection data for IoT protocols. General-purpose benchmarks such as CICIDS2017 [15] established the practice of publishing labelled flows and training classifiers on them. IoT-specific collections followed. MQTTset provides MQTT traffic with attacks against MQTT networks [16]; MQTT-IoT-IDS2020 simulates an MQTT network, injects scans and brute-force attacks, and shows that tree-based models reach an accuracy above 99.9 % on flow features but only about 65 % (random forest) to 89 % (decision tree) on packet features [17]; Edge-IIoTset covers IoT and IIoT test beds with fourteen attacks, MQTT among them [18]; and CICIoT2023 records thirty-three attacks against 105 devices and reports that a random forest reaches an accuracy of 0.992 but an F1 of only 0.714 on its 34 classes [19]. These collections are valuable for protocol-level traffic, but they share three properties. They are labelled at the packet or flow level, they contain no physical process whose measurements an attacker could falsify, and they are partitioned with random or stratified splits. Their idle or scripted normal traffic also contains few of the benign events (sensor faults, link outages that are flushed in bursts, route changes) that make real telemetry hard to classify.

Evaluation of learning-based detectors. Arp et al. catalogued recurring pitfalls in learning-based security research, among them sampling bias, spurious correlations and unrealistic data splits [5]. Pendlebury et al. showed that ignoring the temporal order of samples inflates malware-detection results and proposed time-aware evaluation [6]. Pierazzi et al. argued that robustness should be assessed in the problem space, by changing the object that is analysed, rather than by perturbing feature vectors [10]. We apply these principles to windowed telemetry, in which neighbouring windows of one attack burst are near-duplicates, and we measure the resulting inflation instead of assuming it.

False-data injection. Liu et al. showed that an attacker who knows the measurement model of a state estimator can construct an injection that leaves the residual-based bad-data test unchanged [3], and the subsequent literature, reviewed in [4], studies the attack and its countermeasures at the level of the power system. Machine-learning detectors for this attack are surveyed in [20]; they generally operate on power-system measurements rather than on the communication layer that delivers them. A microgrid with a few metered quantities runs no full state estimator, but the same reasoning applies to its energy balance, and we use it to define a balanced variant of the attack that is invisible to the balance check by construction (Section 4.3). The gap that remains is a joint treatment: attacks on the traffic and on the measurements, observed through one gateway and classified by one model 44-47].

Abuse of constrained protocols. UDP-based request–response protocols can be abused for reflection with a large bandwidth amplification factor [21]. The small request and larger response of CoAP [2] give it this property when an endpoint is reachable from spoofed sources, and MQTT [1] can be flooded or enumerated through its topic structure. We model both abuses together with the benign behaviours that resemble them.

Explainability for security. Explainable intrusion detection is surveyed in [22], which calls for the definition of explanation metrics. SHAP [11, 12], LIME [13] and Integrated Gradients [14] are the most common post-hoc explainers, and their reliability is contested: Shapley-value attributions can mislead as feature-importance measures [7], post-hoc explainers can be fooled by an adversary who controls the input distribution [8], and explanations can be unstable under small perturbations [9]. Warnecke et al. compared six explanation methods on four deep-learning security systems with criteria such as descriptive accuracy, completeness, stability and efficiency, and found that the methods differ substantially [23]. Deletion and insertion curves [24] and comprehensiveness and sufficiency [25] give model-based measures of faithfulness, and Rudin argues that interpretable models should be preferred where possible [26]. We follow the call for a rigorous science of interpretability [27] by auditing each method on faithfulness, stability and agreement, for tree ensembles and neural networks alike.

TABLE I. COMPARISON WITH THE CLOSEST PRIOR WORK. ENTRIES SUMMARISE WHAT EACH CITED PUBLICATION REPORTS; FIGURES MARKED "AS REPORTED" ARE QUOTED FROM THE RESPECTIVE PAPER AND ARE NOT COMPARABLE WITH OURS, BECAUSE DATA, LABELS AND SPLITS DIFFER. "PROCESS-AWARE" MEANS THAT THE DATA CONTAIN A PHYSICAL MEASUREMENT PROCESS THAT AN ATTACK CAN VIOLATE; "XAI" THAT EXPLANATION METHODS ARE EVALUATED QUANTITATIVELY.

Work	Scope	Attack coverage	Reported result or focus	Process-aware	XAI
CICIDS2017 [15]	Enterprise flow data set	DoS/DDoS, brute force, web, botnet, infiltration	Not IoT-specific	No	No
MQTTset [16]	MQTT data set for machine learning	Attacks against MQTT networks	MQTT only	No	No
MQTT-IoT-IDS2020 [17]	Simulated MQTT network; six classical models on packet, uni- and bi-directional flow features	Nmap and UDP scans, SSH and MQTT brute force	RF accuracy 99.98 % on flow features; 65–89 % on packet features (as reported)	No	No
Edge-IIoTset [18]	IoT/IIoT test bed, centralised and federated learning	14 attacks in five threat classes	Includes MQTT among its protocols	No	No
CICIoT2023 [19]	105 IoT devices; RF, DNN and linear baselines, 80/20 split	33 attacks in seven categories	RF accuracy 0.992 but F1 0.714 with 34 classes (as reported); no MQTT or CoAP	No	No
Sayghe *et al.* [20]	Survey of machine learning for FDIA detection	False-data injection in state estimation	Power-system measurements, no network layer	Yes	No
Neupane *et al.* [22]	Survey of explainable intrusion detection	—	Calls for metrics to evaluate explanations	—	Survey
Warnecke *et al.* [23]	Six explanation methods on four deep-learning security systems	Malware, PDF and vulnerability detection	Integrated Gradients and LRP best on their criteria	No	Yes
This study	**Message-level MQTT/CoAP co-simulation of a PV–wind–diesel–battery microgrid; seven models**	**Six attack families (three FDIA variants) and five benign look-alikes**	**Macro-F1 0.913 on a chronological test block**	**Yes**	**Yes**

Position of this study. No work in Table 1 combines a physical process, both transports, benign hard negatives, a leakage-controlled protocol and a quantitative audit of the explanations. Table 36 in the Discussion contrasts the resulting protocol with the practice in these papers.

3. SYSTEM AND THREAT MODEL

3.1 Monitoring Architecture

The monitored site comprises three PV arrays, two wind turbines, a battery energy-storage system, a diesel genset and two load feeders. Seven of the eleven field devices (three PV inverters, two wind controllers, the genset and the battery) publish their active power over MQTT over TCP every 5 s; the two feeder meters, the pyranometer and the anemometer expose CoAP resources and report every 10 s over UDP, with 70 % of messages sent as confirmable (CON) requests. A gateway observes the message stream; the control centre receives the decoded measurements and, for each non-overlapping 30 s window, computes a feature vector from the gateway's message log and the measurement matrix. The detector then assigns the window to one of seven classes: Normal, FDIA, Flood, Amplification, MITM, Replay or Recon.

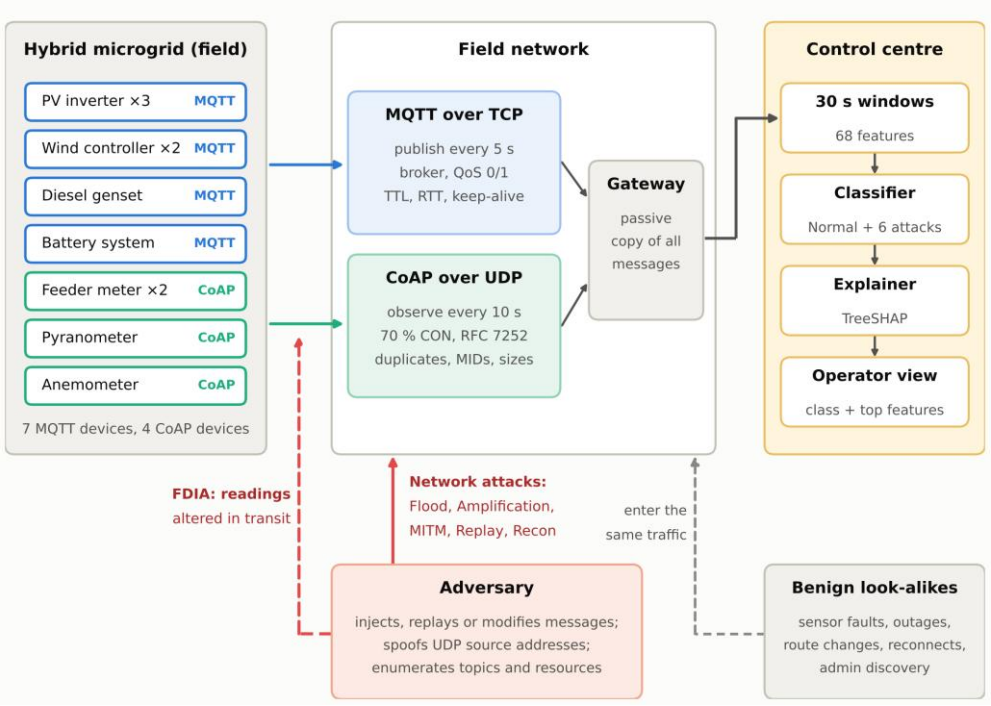


Fig. 2. Monitoring architecture and threat model. Seven MQTT and four CoAP devices of the microgrid reach the control centre through a gateway that passively mirrors every message. The adversary acts on the field network (solid arrows) or alters readings in transit (dashed arrow); benign look-alike events enter the same traffic.

3.2 Adversary

We consider an adversary who can inject, replay or modify messages on the field network, who can spoof UDP source addresses, and who can subscribe to or enumerate MQTT topics and CoAP resources. The adversary cannot break transport encryption where it exists but can exploit its absence or mis-configuration; we assume that the field devices themselves are not compromised, so that FDIA is realised by modifying readings in transit. For the balanced FDIA variant the adversary additionally knows the plant's power-balance relation and the feeder loss factor, which is the knowledge required to build an undetectable injection in the sense of Liu et al. [3]. The adversary does not observe the detector's decision boundary; Section 7.8 separately studies an adversary who lowers attack intensity to stay below detection thresholds.

3.3 Attack and benign taxonomy

Table 2 summarises the six attack families and the benign events with which they must not be confused. Attack bursts last 120–600 s, are mutually exclusive in time with a 60 s guard interval, and on average six bursts per family and day are requested over the 27 post-commissioning days and are placed at random times in random order, bursts that do not fit within 60 attempts being dropped, which leaves 878 bursts in total at site A (between 126 and 168 per family).

TABLE II. ATTACK FAMILIES, THEIR MECHANISM AT SITE A, AND THE FOOTPRINT THROUGH WHICH EACH CAN BE DETECTED.

Class	Layer	Mechanism (parameter range)	Principal footprint
FDIA (V1 bias, V2 ramp, V3 balanced)	Payload; both transports	Additive injection of 10–50 % of the reading on 1–2 sensors; V3 preserves the power balance	Energy residuals; no traffic footprint
Flood	MQTT or CoAP	4–300 extra messages s^{-1} from 1–6 sources	Rate, byte rate, source count, inter-arrival statistics
Amplification	CoAP	2–150 requests s^{-1} from 50–2000 spoofed sources, response/request ratio 4–28	Outbound/inbound byte ratio, source entropy, new sources
MITM	MQTT or CoAP	Delay 8–80 ms, TTL −1, 10–70 % tampered payloads, at most 3 % drops	RTT, TTL deviation, skew, sequence gaps, residual bias

Class	Layer	Mechanism (parameter range)	Principal footprint
Replay	MQTT or CoAP	30–100 % of messages of 1–3 devices replaced by readings 1 min–12 h old	Stale readings: innovation, flatness, resource residuals; skew
Recon	MQTT or CoAP	0.5–15 wildcard subscriptions or path probes s^{-1}	Wildcards, topic and path counts, 4.04 share

TABLE III. ATTACK MODELS: LAYER, ADVERSARY CAPABILITY, PARAMETER RANGES OF THE SIMULATOR AND THE OBSERVABLE THAT CARRIES MOST OF THE EVIDENCE. EVERY BURST LASTS 120–600 S.

Attack	Layer	Adversary capability	Parameters	Main observable
FDIA V1 (bias)	Measurement	Alter readings of 1–2 sensors in transit	$\gamma \in [0.10, 0.35]$ of the larger of reading magnitude and 0.2 P_r	Signed balance and resource residuals
FDIA V2 (ramp)	Measurement	As V1, growing linearly over the burst	$\gamma \in [0.20, 0.50]$, ramp $0 \rightarrow \gamma$	Multi-scale signed bias
FDIA V3 (balanced)	Measurement	Knows power-balance relation and λ; alters one generator and one feeder	Same γ range; residual unchanged	Only PV/irradiance, wind/speed, peer and innovation checks
Flood	Traffic	Send messages from 1–6 sources on one transport	Rate 4–300 s^{-1} (log-uniform)	Message and byte rate, source count
Amplification	Traffic	Spoof UDP sources against an exposed CoAP endpoint	50–2 000 sources; 2–150 s^{-1}; A = 4–28	Response/request bytes, source count
MITM	Traffic + measurement	Relay on 1–3 devices of one transport	Delay 8–80 ms; TTL −1; 10–70 % tampered by 5–30 %	RTT, TTL deviation, payload residual
Replay	Traffic + measurement	Replay old payload, sequence number and time-stamp on 1–3 devices	Fraction 0.3–1; lag 60–43 200 s	Sequence gaps, duplicates, skew, residual
Recon	Traffic	Enumerate topics (MQTT) or resource paths (CoAP)	0.5–15 s^{-1} (log-uniform)	Wildcards, 4.04 rate, distinct paths and ports

4. MATHEMATICAL MODEL

This section specifies the stochastic processes that generate the data. Every quantity below is implemented in the released simulator, and every constant quoted is the value used for site A; site B changes the ratings, climate, link quality and benign-event rates (Section 6).

TABLE IV. PRINCIPAL SYMBOLS USED IN SECTIONS 4 AND 5.

Symbol	Meaning	Unit
t, k, w	time, message index, window index	s, –, –
φ, δ, G_cs	latitude, declination, clear-sky irradiance (Haurwitz)	°, °, W m^{-2}
κ_i, ρ	clearness index of array i; common-cloud coupling (0.85)	–
P^pv_i, P^w_j, P^dg, P^b	PV, wind, diesel and battery power (battery > 0 discharges)	kW
$L = L_1 + L_2$, λ, χ	load, unmetered loss fraction (0.015), curtailment factor	kW, –, –
y_i, s_i	reading of device i and its noise standard deviation	kW, kW
T_i, R_k, δ_k	publication period, round-trip time, clock skew of message k	s, s, s
$\tilde{y}$, γ, $[t_0, t_1]$	value seen by the control centre, attack intensity, burst interval	kW, –, s
r^bal, ρ_j, ι_t	balance residual, resource residual, EWMA innovation	σ units
b_w^(10)	trailing ten-window mean of the signed residual (absolute value)	σ units

Symbol	Meaning	Unit
x ∈ R^68, K = 7	feature vector of a window; number of classes	–
f_θ, w_c	classifier; class weight (N / (K N_c))^½	–
φ_j, v(S)	attribution of feature j; value function of feature subset S	log-odds

4.1 Plant

Solar resource. With latitude φ, declination $\delta(d) = 23.45^\circ \sin(2\pi(284 + d)/365)$ and hour angle $h = 15^\circ(\tau - 12)$ for solar time τ in hours, the clear-sky global horizontal irradiance follows the Haurwitz model [28],

$$G_{\mathrm{cs}}(t) = 1098 \cos\theta_z \exp\left(-\frac{0.059}{\cos\theta_z}\right), \qquad \cos\theta_z = \sin\varphi\sin\delta + \cos\varphi\cos\delta\cos h, \tag{1}$$

and $G_{\mathrm{cs}} = 0$ when the sun is below the horizon. Cloud effects enter through a clearness index $\kappa_i(t)$ for each array,

$$\kappa_i(t) = \mathrm{clip}\big(\bar{\kappa}_d + \sigma_d\big[\sqrt{\rho}\, x_c(t) + \sqrt{1-\rho}\, x_i(t)\big],\, 0.05,\, 1.05\big), \tag{2}$$

where x_c and x_i are unit-variance Ornstein–Uhlenbeck processes with time constants of 480 s and 300 s, $\rho = 0.85$ couples the arrays through a common cloud field, and the daily regime (clear, partly cloudy, overcast) is drawn with probabilities $0.55/0.30/0.15$, setting the mean $\bar{\kappa}_d$ and the volatility σ_d. The available power of array i with rating P_i^{r} is

$$P_i^{\mathrm{pv}}(t) = 0.97\, P_i^{\mathrm{r}}\, \frac{\kappa_i G_{\mathrm{cs}}}{1000}\big[1 - 0.004\big(T_{c,i} - 25\big)\big], \qquad T_{c,i} = T_a(t) + 0.031\, \kappa_i G_{\mathrm{cs}}, \tag{3}$$

with an ambient temperature $T_a(t)$ that follows a diurnal sinusoid. The pyranometer observes $G = \kappa_c G_{\mathrm{cs}}$ with the common cloud field x_c only, which is why the pyranometer and the three arrays are correlated but not identical.

Wind resource. Two turbines share a Gaussian-copula AR(1) process mapped to a Weibull marginal with scale c and shape k,

$$v_j(t) = c\,\Big[-\ln\Big(1 - \Phi\left(z_j(t)\right)\Big)\Big]^{1/k}, \qquad z_j = \sqrt{0.9}\, z_c + \sqrt{0.1}\, u_j, \tag{4}$$

where Φ is the standard normal distribution function, z_c and u_j are Ornstein–Uhlenbeck processes with time constants of 2400 s and 900 s, and a diurnal modulation and 4 % multiplicative turbulence are applied afterwards. Power follows a cubic curve with cut-in, rated and cut-out speeds of 3, 12 and 25 m/s:

$$P_j^{\mathrm{w}} = P_j^{\mathrm{r}}\, \mathrm{clip}\left(\frac{v_j^3 - v_{\mathrm{in}}^3}{v_{\mathrm{r}}^3 - v_{\mathrm{in}}^3}, 0, 1\right) \mathbb{1}\big[v_j \le v_{\mathrm{out}}\big]. \tag{5}$$

Load. Feeder 1 follows a double-peak residential shape and feeder 2 a working-hours commercial shape with a weekend factor; both are scaled by a temperature factor $1 + 0.015\max(T_a - 28, 0)$ and perturbed by multiplicative AR(1) noise of 4 % standard deviation.

Dispatch and exact power balance. With an unmetered distribution loss $\lambda = 0.015$ of the load, the plant satisfies at every step

$$\chi(t)\big[P^{\mathrm{pv}}(t) + P^{\mathrm{w}}(t)\big] + P^{\mathrm{dg}}(t) + P^{\mathrm{b}}(t) = (1 + \lambda)\, L(t), \tag{6}$$

where $P^{\mathrm{b}} > 0$ denotes battery discharge, $L = L_1 + L_2$, and $\chi \in [0,1]$ is a curtailment factor that falls below one only when the surplus exceeds what the battery can absorb. The battery (250 kWh, 40 kW, round-trip efficiency $\eta^2 = 0.92$) is constrained to a state of charge between 0.20 and 0.95. The genset (80 kW rated, 25 kW minimum stable load) starts when the deficit exceeds the battery power or the state of charge falls below 0.22, and stops after 120 consecutive steps of surplus with a state of charge above 0.5. Equation (6) holds to within 10^{-14} kW in the simulation; the *measured* relation is perturbed only by sensor noise and by the unmetered loss λ, which is the same situation as in a real plant.

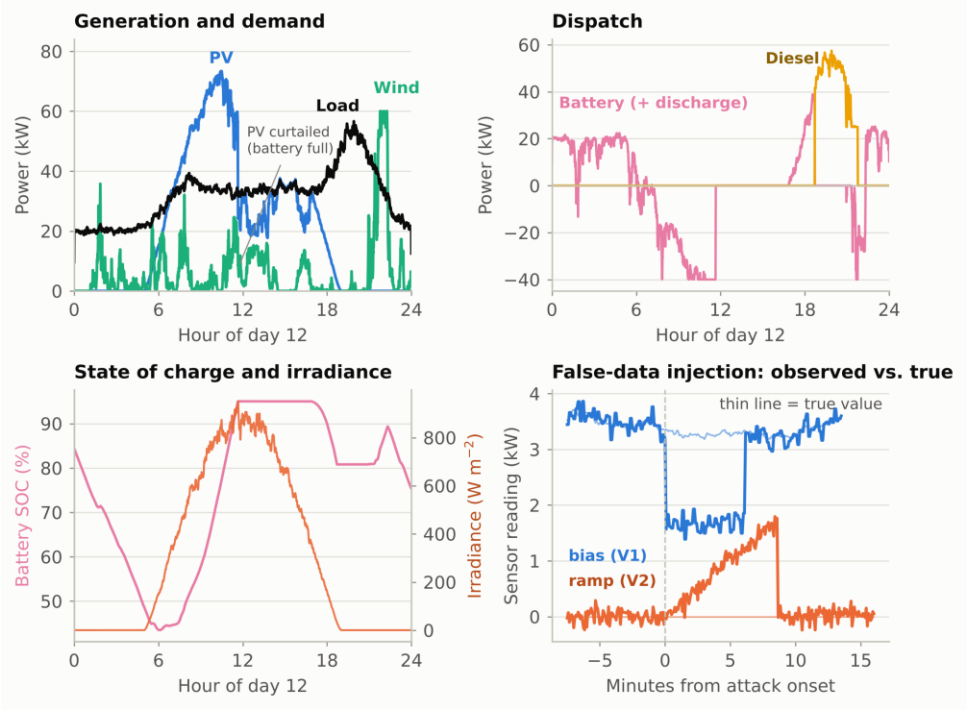


Fig. 3. One simulated day of site A (day 12). Top left: PV, wind and load power, with the PV curtailment that occurs when the battery is full. Top right: battery and diesel dispatch. Bottom left: battery state of charge and irradiance. Bottom right: a false-data injection as seen by the control centre (thick lines) against the true value (thin lines), for a bias (V1) and a ramp (V2) injection.

4.2 Measurement and message processes

Sensors. Device i reports $y_i = x_i + \epsilon_i$ with $\epsilon_i \sim \mathcal{N}(0, s_i^2)$ and $s_i = 0.004\,P_i^{\mathrm{r}} + 0.002\,|x_i|$ for the power channels, $s = 4 + 0.01\,|x|$ W m^{-2} for the pyranometer, and $s = 0.15$ m/s for the anemometer. Device i publishes at $t_k = kT_i + \phi_i$ with a random phase ϕ_i, a period T_i of 5 s (MQTT devices) or 10 s (CoAP devices) and 80 ms of sampling jitter. The control centre holds the last received value on a 5 s grid.

Network. Each message experiences a one-way delay of half a round-trip time $R_k = m_i\,R^{(\tau)}\,e^{\sigma_\tau \xi_k}$ with $\xi_k \sim \mathcal{N}(0,1)$, a per-device multiplier m_i, median $R^{(\tau)}$ of 45 ms (MQTT) or 120 ms (CoAP) and $\sigma_\tau = 0.35$ or 0.50. A confirmable CoAP request that is lost is retransmitted with exponential back-off, so a message that fails $n \in \{1,2,3\}$ times arrives $\sum_{m=1}^{n} 2^m\,u_m$ seconds late with $u_m \sim \mathcal{U}(1,1.5)$, following the ACK_TIMEOUT and ACK_RANDOM_FACTOR conventions of RFC 7252 [2]; a lost acknowledgement produces a duplicate request. Each device has a clock offset $\bar{\delta}_i \sim \mathcal{N}(0, 0.15^2)$ s. The gateway records for every message the arrival time a_k, the device time-stamp t_k, the sequence number, the size, the IP time-to-live (TTL) and, for acknowledged exchanges, the round-trip time. The *skew* of a message is $\delta_k = a_k - t_k - \bar{\delta}_i$, where $\bar{\delta}_i$ is calibrated on the commissioning interval.

Benign events (labelled Normal). Five event types are superimposed with the rates in Table 5: *sensor faults* (stuck-at, intermittent spikes, or linear drift of 5–20 % of rating over 2–20 min), *link outages* (the device buffers its readings, the control centre holds the last value, and on recovery the back-log is flushed at ten times the nominal rate with old device time-stamps), *route changes* (TTL ± 1 and a 10–40 % RTT increase for 0.5–6 h), *client reconnects* (CONNECT and SUBSCRIBE bursts) and *administrator discovery* (3–10 wildcard subscriptions or resource-discovery GETs within one minute).

TABLE V. BENIGN HARD NEGATIVES (LABELLED NORMAL). RATES ARE EVENTS PER DEVICE PER DAY (ADMINISTRATOR DISCOVERY: PER DAY FOR THE WHOLE SYSTEM).

Event	Site A	Site B	Effect	Confusable with
Sensor fault	0.25	0.4	Stuck value, spikes of 5–25 % of rating on 10 % of the samples, or linear drift to 5–20 %; 2–20 min	FDIA V1/V2, Replay

Event	Site A	Site B	Effect	Confusable with
Link outage + flush	0.35	0.6	Held value, then 10× burst with old time-stamps (1–10 min)	Flood, Replay
Route change	0.08	0.15	TTL ±1, RTT +10–40 % for 0.5–6 h	MITM
Client reconnect (MQTT)	0.4	0.7	CONNECT/SUBSCRIBE burst	Recon, Flood
Administrator discovery	1.0	1.0	3–10 wildcard requests in 10–60 s (system-wide)	Recon

4.3 Attack models

A campaign is active on $[t_0, t_1]$ with $t_1 - t_0 \sim \mathcal{U}(120{,}600)$ s. We write $\tilde{y}$ for the value seen by the control centre.

FDIA, variant V1 (bias). On a set $\mathcal{S}$ of one or two sensors, $\tilde{y}_s = y_s + \sigma_s \gamma \max(|y_s|, 0.2\, P_s^{\mathrm{r}})$ with sign $\sigma_s \in \{\pm 1\}$ and $\gamma \sim \mathcal{U}(0.10{,}0.35)$.

FDIA, variant V2 (ramp). The same injection is scaled by $\varrho(t) = (t - t_0)/(t_1 - t_0)$ and $\gamma \sim \mathcal{U}(0.20{,}0.50)$, which makes it a slowly growing drift that mimics a degrading sensor.

FDIA, variant V3 (balanced). Following the structure $\mathbf{a} = \mathbf{Hc}$ of Liu et al. [3], the attacker perturbs one generation-side device g and one load feeder ℓ so that the energy-balance residual is unchanged:

$$\tilde{y}_g = y_g + a, \qquad \tilde{y}_\ell = y_\ell + \frac{a}{1+\lambda}, \qquad a = \sigma\, \gamma \max\left(\left|y_g\right|, 0.2\, P_g^{\mathrm{r}}\right). \tag{7}$$

With the plant relation of Eq. (6) the residual $\sum_{i \in \mathcal{G}} \tilde{y}_i - (1+\lambda) \sum_\ell \tilde{y}_\ell$ equals its un-attacked value exactly. This variant is the theoretical limit of balance-based detection; it is only detectable through the *other* consistency relations (PV versus irradiance, wind versus wind speed, peer comparison, temporal innovation).

Flood. On one transport, $n_s \in \{1, \ldots, 6\}$ sources send extra messages (MQTT PUBLISH or CONNECT; CoAP GET or POST) as a Poisson process with rate λ_f drawn log-uniformly from $[4{,}300]$ s^{-1}.

Amplification (CoAP). A pool of $P \in [50{,}2000]$ spoofed source addresses (log-uniform) issues requests of $s_{\mathrm{req}} \sim \mathcal{U}(40{,}70)$ B at a rate $\lambda_a \in [2{,}150]$ s^{-1} (log-uniform) to which the exposed endpoint answers with $s_{\mathrm{resp}} \sim \mathcal{U}(300{,}1100)$ B; the bandwidth amplification factor is $A = s_{\mathrm{resp}}/s_{\mathrm{req}}$.

Man-in-the-middle (MITM). On one to three devices of one transport, the attacker adds a delay $d \sim \mathcal{U}(8{,}80)$ ms (which also lengthens the RTT by $2d$), decrements the TTL by one, tampers with a fraction $q \sim \mathcal{U}(0.1{,}0.7)$ of the payloads by a relative perturbation $\pm\gamma \max(|y|, 0.2P^{\mathrm{r}})$ with $\gamma \sim \mathcal{U}(0.05{,}0.30)$, drops at most 3 % and duplicates at most 2 % of the messages.

Replay. For a fraction $f \sim \mathcal{U}(0.3{,}1)$ of the messages of one to three devices, the payload (and the old sequence number and time-stamp) is replaced by the one recorded $\Delta \in [60{,}43200]$ s earlier (log-uniform).

Reconnaissance. On one transport the attacker issues wildcard subscriptions and topic enumeration (MQTT) or resource-path enumeration returning 4.04 responses (CoAP) at $\lambda_r \in [0.5{,}15]$ s^{-1} (log-uniform).

4.4 Classification task

The 30 s window w receives the label of the burst whose overlap with the window is largest provided that overlap is at least 0.5, otherwise Normal. A classifier $f_\theta : \mathbb{R}^{68} \to \Delta^6$ maps the window's feature vector to a probability vector over $K = 7$ classes. Models are trained with a class-weighted cross-entropy,

$$\mathcal{L}(\theta) = -\sum_n w_{y_n} \log f_\theta(\mathbf{x}_n)_{y_n}, \qquad w_c = \left(\frac{N}{K\, N_c}\right)^{1/2}, \tag{8}$$

where N_c is the number of training windows of class c. The square root tempers the weights so that the rare classes are emphasised without distorting the posterior as strongly as full balancing. Performance is summarised by the macro-F1, $\frac{1}{K}\sum_c 2\, P_c R_c/(P_c + R_c)$, which weights the six attack classes and Normal equally, and by the Matthews correlation coefficient [29].

5. METHODOLOGY

5.1 Feature extraction

The detector operates on non-overlapping 30 s windows ($W = 30$ s, six plant steps). Each window is described by 68 features in eight groups (Table 6). All calibration constants (per-device skew offsets, TTL baselines, noise scales, regression coefficients) are estimated on the first three days only, which contain no attack, and are frozen thereafter. Data from the commissioning interval are never used to train or evaluate a classifier.

TABLE VI. FEATURE GROUPS. THE 68 FEATURES ARE COMPUTED PER 30 S WINDOW.

Group	No.	Content
Traffic volume	14	message and byte rate, inter-arrival CV, fast-gap share, size mean/std, outbound/inbound ratio (per transport)
Source dynamics	8	distinct sources, source entropy, top-source share, new sources (per transport)
Temporal integrity	12	mean RTT, TTL deviation, sequence-gap rate, duplicate rate, median and p95 skew (per transport)
MQTT semantics	6	topic entropy, topic count, CONNECT and SUBSCRIBE rates, wildcard count, QoS-0 share
CoAP semantics	5	CON share, 4.04 share, distinct paths, distinct source ports, GET share
Energy residuals	13	balance residual (mean, max); resource, innovation, CUSUM and flatness per transport; peer residual
Energy bias (multi-scale)	8	signed-bias aggregates at 30 s and 5 min for balance, peer and per-transport channels
Context	2	sine and cosine of time of day
Total	68	

Traffic features. For each transport separately the gateway log of a window yields the message rate and byte rate, the coefficient of variation of the inter-arrival times $\mathrm{CV} = \sigma_{\Delta t}/\mu_{\Delta t}$, the fraction of inter-arrival gaps shorter than 10 ms, the mean and standard deviation of the request size and the response-to-request byte ratio. Source dynamics are the number of distinct sources, their Shannon entropy $H_{\mathrm{src}} = -\sum_s p_s \log_2 p_s$ with $p_s = n_s/n$, the share of the dominant source and the number of previously unseen source identifiers. Temporal-integrity features are computed over messages from known devices only: the mean RTT, the fraction of messages whose TTL differs from the device's baseline, the fraction with a sequence gap, the fraction with a duplicate identifier, and the median and 95th percentile of $|\delta_k|$, the calibrated time-stamp skew. Protocol semantics are, for MQTT, topic entropy, the number of distinct topics, CONNECT and SUBSCRIBE rates, the number of wildcard subscriptions and the QoS-0 share, and, for CoAP, the CON share, the 4.04 share, the number of distinct paths and source ports and the GET share. Features that are undefined in a window (for example the RTT of a window without an acknowledged exchange) are imputed with the commissioning median.

Physics-consistency features. Let $\mathbf{y}(t)$ be the control-centre measurement vector on the 5 s grid. Robust scales are estimated as $\hat{\sigma} = 1.4826\,\mathrm{MAD}$ over the commissioning days, with heteroscedastic models $\hat{\sigma}(x) = \hat{s}_0 + \hat{s}_1|x|$ fitted by least squares to $1.2533\,|\mathrm{res}|$ wherever noise depends on the signal level.

Balance residual. With $\mathcal{G}$ the seven generation and storage channels,

$$r^{\mathrm{bal}}(t) = \frac{\sum_{i\in\mathcal{G}} y_i(t) - \hat{\alpha}\left(y_{\ell_1}(t) + y_{\ell_2}(t)\right)}{\hat{\sigma}_{\mathrm{bal}}}, \qquad \hat{\alpha} = \frac{\sum_t X_t L_t}{\sum_t L_t^2}, \tag{9}$$

where $X_t = \sum_{i\in\mathcal{G}} y_i$ and $L_t = y_{\ell_1} + y_{\ell_2}$; $\hat{\alpha}$ absorbs the unmetered loss.

Resource consistency. For PV array i, $\rho_i^{\mathrm{pv}} = (y_i - \hat{c}_i G)/\hat{\sigma}_i(\hat{c}_i G)$, where G is the pyranometer reading and $\hat{c}_i$ is the least-squares irradiance-to-power gain over daylight samples. For turbine j, $\rho_j^{\mathrm{w}} = \left(y_j/P_j^{\mathrm{r}} - \hat{f}(v)\right)/\hat{\sigma}_w(v)$ with $\hat{f}$ a monotone power curve estimated in 0.5 m/s bins of the anemometer reading v. The pyranometer itself is checked against the irradiance implied by the arrays, $\bar{G}^{\mathrm{pv}} = \frac{1}{3}\sum_i y_i/\hat{c}_i$, and against the physical upper bound $1.15\,G_{\mathrm{cs}}(t)$:

$$\rho^{G}(t) = \max\left(\frac{|G - \bar{G}^{\mathrm{pv}}|}{\hat{\sigma}_G}, \frac{\max(G - 1.15\, G_{\mathrm{cs}}, 0)}{\hat{\sigma}_G}\right). \tag{10}$$

Temporal and peer consistency. An exponentially weighted moving average $\hat{y}_t = \beta y_t + (1-\beta)\hat{y}_{t-1}$ with $\beta = 0.3$ gives the normalised innovation $\iota_t = (y_t - \hat{y}_{t-1})/(\hat{s}_0 + \hat{s}_1|\hat{y}_{t-1}|)$. The three PV arrays are compared with their per-unit median, $\pi_i = \big(p_i - \mathrm{med}(\mathbf{p})\big)/\big(\hat{s}_0 + \hat{s}_1\,\mathrm{med}(\mathbf{p})\big)$ with $p_i = y_i/P_i^{\mathrm{r}}$. A flatness statistic divides the 60 s rolling standard deviation by its commissioning median and flags frozen readings. Persistent small deviations are accumulated by a one-sided CUSUM [30] of the absolute residuals $|\rho_t|$ with reference value $k = 3$ and a cap of 100,

$$S_t = \min\big(100,\ \max(0,\ S_{t-1} + |\rho_t| - k)\big). \tag{11}$$

Multi-scale signed-bias aggregation. Noise averages out in a signed residual while a bias does not, so for each channel we keep the sign. Let $\bar{\rho}_w$ be the mean of the signed residual over the six steps of window w. The window features are $|\bar{\rho}_w|$ and the trailing ten-window mean

$$b_w^{(10)} = \left|\frac{1}{10}\sum_{j=0}^{9}\bar{\rho}_{w-j}\right|, \tag{12}$$

which spans five minutes and is causal. For independent residual noise the standard deviation of $b^{(10)}$ falls by up to $\sqrt{60}$ relative to a single sample, which is what lets a bias of a fraction of a noise standard deviation become visible. All residual families are aggregated per transport (maximum over the transport's channels of the window mean and window maximum) so that the feature dimension does not depend on which sensor the attacker chooses.

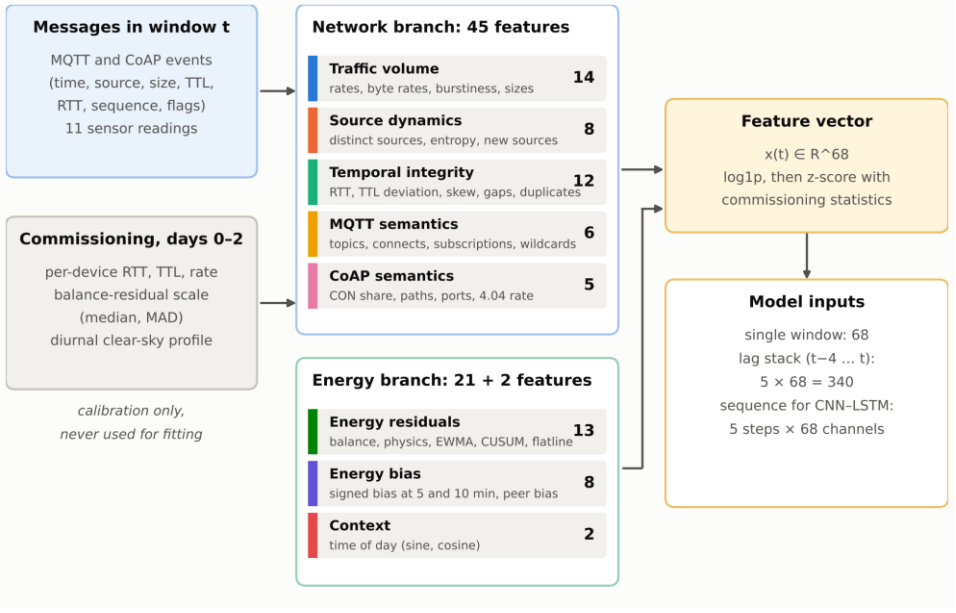


Fig. 4. Feature extraction. Messages of a 30 s window feed the network branch (45 features in five groups) and the control-centre readings feed the energy branch (21 features in two groups) plus two time-of-day features; both use baselines estimated on the commissioning days only. The resulting 68-dimensional vector is standardised and presented to the models as a single window, as a stack of five windows or as a sequence.

5.2 Classifiers

We compare six standard models and one temporal variant. *Logistic regression* (multinomial, L2) is the linear reference. *Random forest* [31] and *XGBoost* [32] and *LightGBM* [33] are the tree ensembles. A *multi-layer perceptron* (MLP) takes the standardised features after a $\log(1+x)$ transform. A *CNN–LSTM* [34] receives the last $T = 5$ windows (two minutes of context) as a sequence: two one-dimensional convolutions with kernel size 3 (64 and 128 filters at the tuned setting) feed an LSTM with 100 units, a 64-unit dense layer with dropout and a softmax output. *XGBoost (lag)* concatenates the features of the current window and the four preceding windows (340 inputs) so that a gradient-boosted model can use the same temporal context as the CNN–LSTM. The neural networks are trained with Adam [35] and early stopping on the

validation macro-F1; the boosted models use early stopping on the validation log-loss. Hyper-parameters are chosen by random search [36] on the validation block with 6–12 trials per model (Table 7); the test block is never consulted.

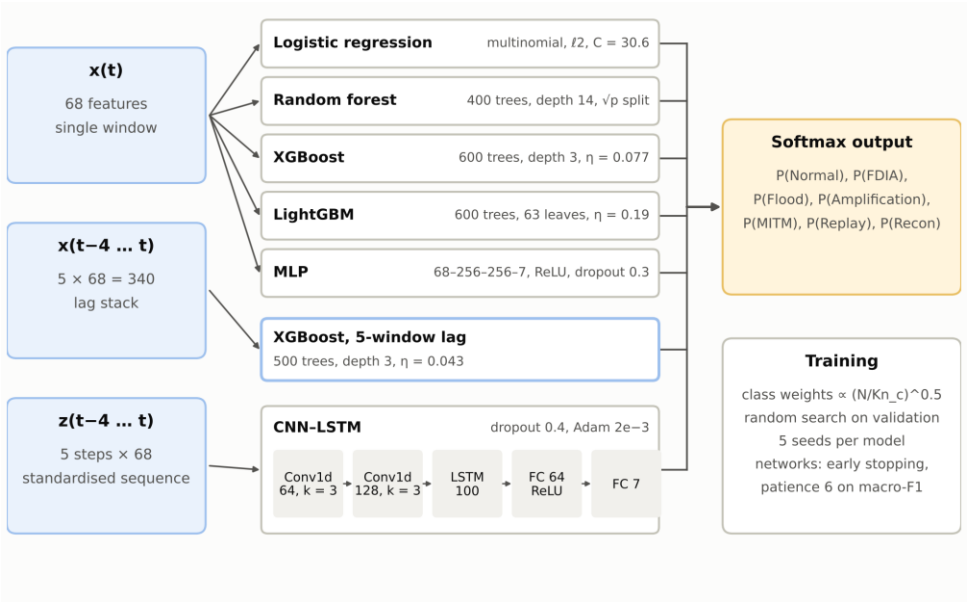


Fig. 5. Model zoo. Seven classifiers receive the single-window vector, a stack of the five latest windows or a sequence of five standardised windows, and output a probability vector over the seven classes; the tuned hyper-parameters are shown.

TABLE VII. RANDOM-SEARCH BUDGETS, SEARCH SPACES, SELECTED SETTINGS AND VALIDATION MACRO-F1 (DAYS 20–22).

Model	Trials	Search space	Selected	Val. F1
Logistic regression	6	$C \in [10^{-3}, 10^{2}]$ (log scale)	C=30.6	0.720
Random forest	8	trees {200,300,400}; depth {10,14,18,24,none}; leaf {1,2,4}; max features {√F, 0.3, 0.5}	n_estimators=400, max_depth=14, min_samples_leaf=2, max_features=sqrt	0.854
XGBoost	12	depth 3–9; learning rate 0.03–0.3; subsample 0.6–1; colsample 0.5–1; λ, γ, min child weight	max_depth=3, learning_rate=0.0773, subsample=0.696, colsample_bytree=0.519, reg_lambda=5.66, gamma=0.468, min_child_weight=3.53	0.859
LightGBM	10	leaves {15,31,63,127}; learning rate 0.03–0.3; subsample, colsample, λ; min child {10,20,50}	num_leaves=63, learning_rate=0.194, subsample=0.655, colsample_bytree=0.709, reg_lambda=4.27, min_child_samples=50	0.851
MLP	8	width {64,128,256}; depth {2,3}; dropout {0.1,0.3}; lr {1e-3,3e-3}; weight decay {1e-5,1e-4,1e-3}	hidden=256, depth=2, dropout=0.3, lr=0.003, wd=0.001	0.831
CNN–LSTM	6	filters {32,64}; LSTM {64,100}; dropout {0.2,0.4}; lr {1e-3,2e-3}; weight decay {1e-5,1e-4}	conv=64, lstm=100, dropout=0.4, lr=0.002, wd=0.0001	0.846
XGBoost (lag)	6	as XGBoost, depth 3–8, colsample 0.3–0.8	max_depth=3, learning_rate=0.0433, subsample=0.836, colsample_bytree=0.383, reg_lambda=2.27, gamma=0.0211, min_child_weight=2.04	0.864

TABLE VIII. TUNING AND TRAINING COST: NUMBER OF RANDOM-SEARCH TRIALS, BEST VALIDATION MACRO-F1, TEST MACRO-F1, THE DIFFERENCE BETWEEN THE TWO (TEST − VALIDATION) AND THE FIT TIME OF ONE MODEL (SEED 0, TWO VIRTUAL CPU CORES).

Model	Trials	Validation F1	Test F1	Test − val.	Fit time (s)
Logistic regression	6	0.720	0.747	+0.026	10.9
Random forest	8	0.854	0.896	+0.042	49.4
XGBoost	12	0.859	0.907	+0.048	5.6
LightGBM	10	0.851	0.906	+0.056	3.8
MLP	8	0.831	0.883	+0.051	9.3
CNN-LSTM	6	0.846	0.888	+0.042	48.3
XGBoost (5-window lag)	6	0.864	0.913	+0.050	40.6

5.3 Evaluation protocol

Chronological split. Days 0–2 form the commissioning interval, days 3–19 the training set, days 20–22 the validation set and days 23–29 the test set. A purge gap of 24 windows (12 min) before the validation and test boundaries removes windows whose context could overlap a burst that straddles the boundary. Because attack bursts are scheduled uniformly over days 3–29, all attack types appear in every partition, but no burst appears in more than one partition. As a contrast we also evaluate (i) a stratified random split of the windows of days 3–29 and (ii) a five-fold split grouped by attack burst, using for Normal windows one-hour blocks as groups (Section 7.4).

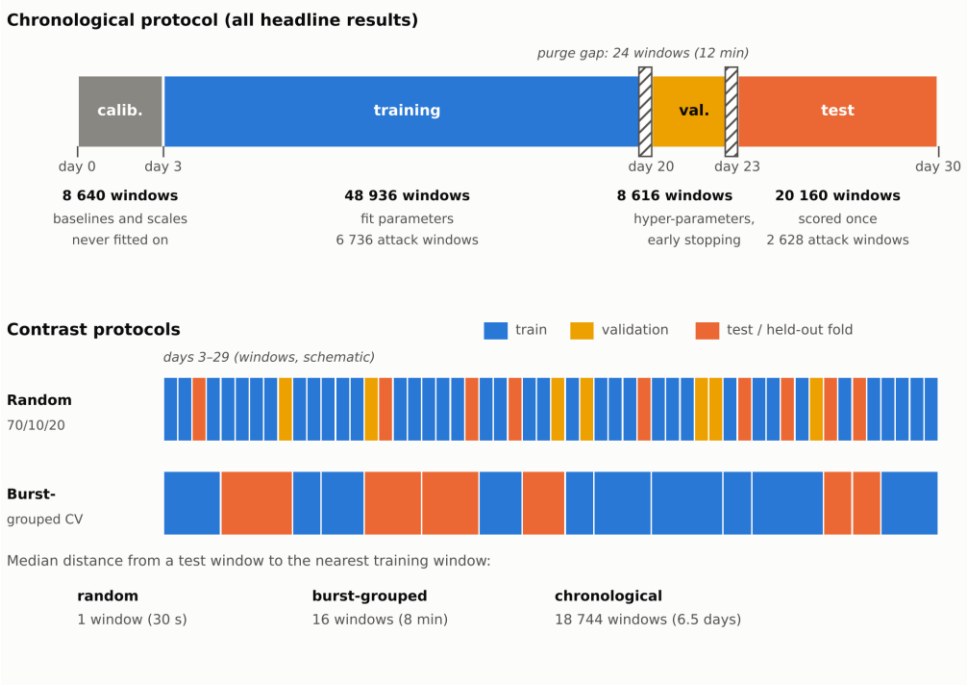


Fig. 6. Evaluation protocol. Top: the chronological split used for all headline results, with calibration, training, validation and test days, the purge gaps and the number of windows in each partition. Bottom: the random and burst-grouped contrast protocols, and the median distance between a test window and the nearest training window under each.

Metrics. The primary score is the macro-averaged F1, which weights the seven classes equally although Normal windows make up 87 % of the test block; it is complemented by the multiclass Matthews correlation coefficient [29], Cohen's kappa, balanced accuracy, one-vs-rest AUC and average precision, and by two calibration scores, the expected calibration error [37] and the Brier score (Table 9). Operational quantities are the detection delay in windows and the false-alarm rate on Normal windows.

TABLE IX. EVALUATION METRICS. K = 7 CLASSES, P_C AND R_C ARE PRECISION AND RECALL OF CLASS C, C IS THE NUMBER OF CORRECT PREDICTIONS, S THE NUMBER OF SAMPLES, P_K AND T_K THE PREDICTED AND TRUE COUNTS OF CLASS K.

Metric	Definition	Range	Role
Macro-F1	(1/K) Σ_c 2 P_c R_c / (P_c + R_c)	[0, 1]	Primary score; every class counts equally
MCC (multiclass)	(c·s − Σ_k p_k t_k) / √[(s² − Σ p_k²)(s² − Σ t_k²)]	[−1, 1]	Robust to the 87 % Normal share
Cohen's κ	(p_o − p_e) / (1 − p_e)	[−1, 1]	Chance-corrected agreement
Balanced accuracy	(1/K) Σ_c R_c	[0, 1]	Mean recall
Macro AUC / AP	one-vs-rest ROC-AUC and average precision, averaged over classes	[0, 1]	Threshold-free ranking quality
ECE	Σ_b (n_b / N) |acc_b − conf_b|, 15 equal-width bins, top label	[0, 1]	Calibration of the top probability
Brier score	(1/N) Σ_n Σ_c (p_nc − 1[y_n = c])²	[0, 2]	Proper score of the full vector
Detection delay	windows from burst onset to the first correct window	≥ 0	Timeliness (30 s resolution)
False-alarm rate	share of Normal windows classified as an attack	[0, 1]	Operator workload

Uncertainty. Windows within a burst are strongly dependent, so we resample whole 10-minute blocks of the test set (a moving-block bootstrap in the sense of Künsch [38]) with $B = 1000$ replications to obtain 95 % percentile intervals for the macro-F1, and $B = 2000$ replications of the paired difference to compare two models. Variability due to initialisation is reported over five seeds.

Multiple comparisons. For k models evaluated on N test days we rank the models by per-day macro-F1, apply the Friedman test, and report the Nemenyi critical difference $\mathrm{CD} = q_\alpha\sqrt{k(k+1)/(6N)}$ with $q_{0.05} = 2.949$ for $k = 7$ [39]. Pairwise block-bootstrap p-values are corrected by the Holm procedure [40].

Further experiments. We evaluate (a) leave-one-group-out and group-only ablations over the eight feature groups; (b) five imbalance strategies (none, class weights, SMOTE [41], random undersampling, SMOTE with weights); (c) leave-one-variant-out generalisation across the three FDIA variants; (d) evasion by scaling attack intensity by $s \in \{1, 0.5, 0.25, 0.1, 0.05\}$ in the *problem space*, that is by regenerating the traffic and the measurements under the weakened attack rather than perturbing feature vectors [10]; and (e) transfer to an independently simulated site B with a different climate, ratings, link quality and benign-event rates, zero-shot and after adding $k \in \{1, 2, 4, 8\}$ days of site-B data to the training set. Evasion and transfer use models trained on site A only, with the site-A calibration.

5.4 Explanation methods

For an instance $\mathbf{x}$ with predicted class c, an additive attribution assigns ϕ_j to feature j. *TreeSHAP* computes exact Shapley values [11] of the tree ensemble's log-odds for class c,

$$\phi_j = \sum_{S \subseteq F \setminus j} \frac{|S|!\,(|F| - |S| - 1)!}{|F|!}\big(v(S \cup j) - v(S)\big), \tag{13}$$

where F is the set of features, $S \cup j$ is the subset S with feature j added and the value function v is the path-dependent conditional expectation of the model output. *KernelSHAP* [12] estimates the same quantity for any model by weighted regression over feature coalitions with a background sample; we use it to check TreeSHAP on a subset. *LIME* [13] fits, for each instance, a sparse weighted ridge surrogate

$$\hat{g} = \operatorname*{argmin}_{g \in \mathcal{G}} \sum_z \pi_{\mathbf{x}}(z)\big(f_c(z) - g(z)\big)^2 + \Omega(g), \qquad \pi_{\mathbf{x}}(z) = \exp\left(-\frac{\| z - \mathbf{x} \|_\sigma^2}{\ell^2}\right), \tag{14}$$

where z are 1000 Gaussian perturbations around $\mathbf{x}$ with the commissioning standard deviation as scale, $\|\cdot\|_\sigma$ is the standardised Euclidean norm, the kernel width is $\ell = 0.75\sqrt{|F|}$ and g keeps the eight largest coefficients. We implement this procedure directly, as the reference package could not be built in our environment; the instance-wise surrogate follows the original algorithm. *Integrated Gradients* [14] attributes the neural networks' outputs by integrating the gradient along a straight path from the baseline $\mathbf{x}'$ (the commissioning mean in standardised space) to the input,

$$\mathrm{IG}_j(\mathbf{x}) = \left(x_j - x'_j\right) \int_0^1 \frac{\partial f_c\left(\mathbf{x}' + a(\mathbf{x} - \mathbf{x}')\right)}{\partial x_j}\, da, \tag{15}$$

approximated with 32 steps; for the CNN–LSTM the attribution is summed over the five time steps.

We then *measure* the explanations instead of inspecting them.

Faithfulness. Features are ranked by attribution and either removed in that order (set to the commissioning median) while the predicted-class probability is tracked (*deletion*), or inserted into the baseline in that order (*insertion*). Lower deletion area and higher insertion area under the curve indicate a more faithful ranking [24]. *Comprehensiveness* $p_c(\mathbf{x}) - p_c(\mathbf{x}_{\setminus k})$ is the probability drop when the top-k features are removed, and *sufficiency* $p_c(\mathbf{x}) - p_c(\mathbf{x}_k)$ is the drop when only the top-k features are kept [25]; we report $k = 5$. A random ranking is the null reference. *Stability* is the top-5 Jaccard overlap and, for TreeSHAP, the relative ℓ_2 change of the attribution vector between an instance and a copy perturbed by Gaussian noise of 5 % of the commissioning standard deviation, restricted to perturbations that do not change the prediction (in the spirit of max-sensitivity [9]). *Agreement* between methods is the per-instance Kendall τ and the top-5 Jaccard index of the absolute attributions. *Group attribution* sums $|\phi_j|$ within each of the eight feature groups, which turns a 68-dimensional explanation into a statement an operator can act on.

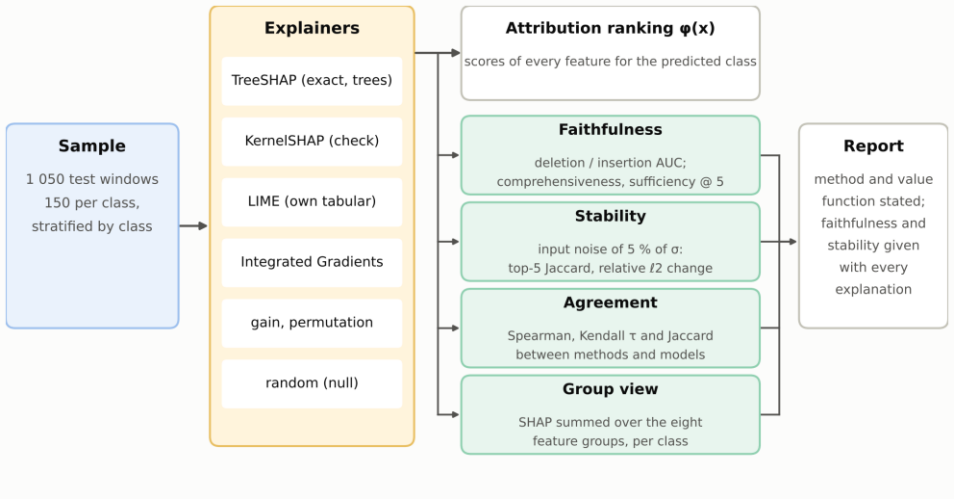


Fig. 7. Explanation audit. A class-stratified sample is explained by each method; the resulting rankings are scored for faithfulness, stability and agreement, and summed over the feature groups for interpretation.

TABLE X. MEASURES USED TO AUDIT THE EXPLANATIONS (P IS THE MODEL PROBABILITY OF THE PREDICTED CLASS).

Measure	Definition	Property
Deletion AUC (↓)	Area under the predicted-class probability as features are replaced by the commissioning median in order of decreasing attribution	Faithfulness
Insertion AUC (↑)	Same, inserting features into the median baseline	Faithfulness
Comprehensiveness@k (↑)	p(x) − p(x without its k top features)	Faithfulness
Sufficiency@k (↓)	p(x) − p(x with only its k top features)	Faithfulness
Top-5 Jaccard under noise (↑)	Overlap of the five largest attributions of a window and of a copy with Gaussian noise of 5 % of the commissioning σ	Stability
Relative ℓ₂ change (↓)	‖φ(x′) − φ(x)‖₂ / ‖φ(x)‖₂ for the same pair	Stability
Spearman ρ, Kendall τ, top-5 Jaccard	Rank agreement of two attribution vectors for the same window (or of two global rankings)	Agreement
Group share	Σ_{j in group} |φ_j| / Σ_j |φ_j|, averaged per predicted class	Interpretation

6. EXPERIMENTAL SET-UP

Data. Site A is simulated for 30 days at a 5 s plant resolution, which yields 86,400 windows of 30 s and about 0.52 million gateway messages on a typical day (day 25). Of the windows in the training, validation and test partitions, 10,605 (13.6 %) carry an attack label. Table 11 gives the class composition of each partition, and Fig. 8 shows the schedule of bursts. Site B (86,400 windows) differs in latitude, season, generation and storage ratings, link quality, device clock accuracy, the share of confirmable CoAP messages and the rates of benign events, and is used only for the transfer experiment.

TABLE XI. WINDOWS PER CLASS AND PARTITION AT SITE A (THE COMMISSIONING DAYS 0–2 AND THE PURGE WINDOWS ARE EXCLUDED).

Class	Train	Validation	Test	Total	Share (%)
Normal	42,200	7,375	17,532	67,107	86.35
FDIA	1,202	215	397	1,814	2.33
Flood	1,272	222	524	2,018	2.60
Amplification	975	237	389	1,601	2.06
MITM	1,279	177	422	1,878	2.42
Replay	938	181	398	1,517	1.95
Recon	1,070	209	498	1,777	2.29
Total	48,936	8,616	20,160	77,712	100

TABLE XII. SIMULATOR PARAMETERS OF THE TWO SITES (TAKEN FROM THE CONFIGURATION OF THE RELEASED CODE). SITE B DIFFERS IN EVERY ROW EXCEPT THE PLANT TOPOLOGY AND THE ATTACK SCHEDULE PARAMETERS.

Parameter	Site A	Site B
Latitude (°N), start day of year	33.3, 152	24.7, 240
PV ratings (kW)	40 / 40 / 25	30 / 30 / 30
Wind ratings (kW); Weibull scale c (m/s), shape k	30 / 30; 6.5, 2	40 / 20; 5.2, 2
Diesel rating / minimum load (kW)	80 / 25	70 / 20
Battery energy (kWh) / power (kW)	250 / 40	180 / 30
Feeder peak loads (kW); unmetered loss λ	45 / 28; 0.015	40 / 34; 0.025
Sensor noise: relative to rating / proportional	0.004 / 0.002	0.004 / 0.002
Median RTT MQTT / CoAP (ms); log-sd	45 / 120; 0.35 / 0.5	70 / 180; 0.45 / 0.6
Base packet loss MQTT / CoAP	0.003 / 0.015	0.006 / 0.03
Device clock-offset sd (s); CoAP CON share	0.15; 0.7	0.3; 0.55
Benign events per node and day: fault / outage / route / reconnect	0.25 / 0.35 / 0.08 / 0.4	0.4 / 0.6 / 0.15 / 0.7
Requested bursts per family and day; burst length (s)	6; 120–600	6; 120–600

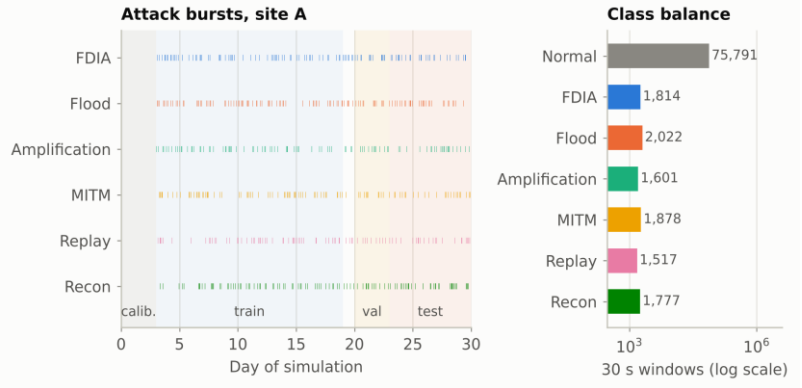


Fig. 8. Attack schedule at site A. Left: each tick is one attack burst, shaded by the calibration, training, validation and test intervals; the unshaded gap before day 20 is the purge interval. Right: number of 30 s windows per class on a logarithmic scale.

Implementation. The simulator, the feature extractor and all experiments are implemented in Python 3.13 with NumPy, SciPy and pandas; scikit-learn [42] provides logistic regression and random forests, XGBoost [32] and LightGBM [33] the boosted models, PyTorch [43] the neural networks, and the `shap` package the TreeSHAP and KernelSHAP computations. All random processes are seeded; the data set, the tuning log and every result file can be regenerated from the supplementary code. All experiments, including the latency measurements, ran on a container with 2 virtual CPU cores (Intel(R) Xeon(R) Processor @ 2.80GHz) and no GPU.

7. RESULTS

All numbers in this section are produced by the code in the supplementary archive; unless stated otherwise they refer to the chronological test block (days 23–29 of site A, 20 160 windows, of which 2 628 are attacks) and to the five-seed mean.

7.1 Overall performance

Table 13 compares the seven classifiers. The gradient-boosted model with a five-window lag is the best detector, with a macro-F1 of 0.913 (95 % block-bootstrap interval 0.893–0.927), an MCC of 0.907 and a macro-AUC of 0.974. Plain XGBoost (0.907), LightGBM (0.906) and the random forest (0.896) follow closely, whereas the two neural networks are lower (CNN–LSTM 0.888, MLP 0.883) and the linear reference is far behind (0.747). The weakness of logistic regression comes entirely from its precision on the minority class and on the Normal–attack boundary: its balanced accuracy of 0.900 is close to that of the trees, but its macro-precision is 0.695 because many Normal windows are flagged (the Normal recall is 0.825).

TABLE XIII. CHRONOLOGICAL HOLD-OUT TEST (DAYS 23–29, SITE A). ALL POINT ESTIMATES ARE MEANS OVER FIVE SEEDS (± STANDARD DEVIATION FOR MACRO-F1; LOGISTIC REGRESSION IS DETERMINISTIC). THE CONFIDENCE INTERVAL IS THE 10-MINUTE MOVING-BLOCK BOOTSTRAP (1 000 RESAMPLES) OF THE SEED-0 PREDICTIONS. BEST PER COLUMN IN BOLD.

Model	Macro-F1 (mean ± sd, 5 seeds)	95 % block-bootstrap CI	MCC	κ	Balanced acc.	Macro AUC	ECE ↓	Brier ↓
Logistic regression	0.747	[0.727, 0.764]	0.607	0.553	0.900	0.969	0.078	0.255
Random forest	0.896 ± 0.001	[0.880, 0.913]	0.879	0.878	0.910	0.969	0.244	0.156
XGBoost	0.907 ± 0.001	[0.886, 0.920]	0.898	0.898	0.910	0.971	0.043	0.048
LightGBM	0.906 ± 0.003	[0.884, 0.919]	0.902	0.902	0.898	0.969	**0.007**	**0.039**
MLP	0.883 ± 0.009	[0.869, 0.905]	0.875	0.875	0.891	0.970	0.025	0.058
CNN-LSTM	0.888 ± 0.006	[0.869, 0.905]	0.876	0.876	0.902	0.971	0.016	0.054
XGBoost (5-window lag)	**0.913 ± 0.001**	[0.893, 0.927]	**0.907**	**0.907**	**0.911**	**0.974**	0.037	0.042

TABLE XIV. FURTHER TEST METRICS (MEANS OVER FIVE SEEDS): ACCURACY, WEIGHTED F1, MACRO-AVERAGED PRECISION AND RECALL, F1 OF THE ATTACK-VERSUS-NORMAL DECISION, FALSE-ALARM RATE ON NORMAL WINDOWS (%) AND MACRO AVERAGE PRECISION. BEST PER COLUMN IN BOLD (LOWEST FOR THE FALSE-ALARM RATE).

Model	Accuracy	Weighted F1	Macro precision	Macro recall	Attack vs. Normal F1	False alarms (%)	Macro AP
Logistic regression	0.838	0.879	0.695	0.900	0.721	17.45	0.885
Random forest	0.970	0.971	0.886	0.910	0.882	2.06	0.920
XGBoost	0.975	0.975	0.905	0.910	0.894	1.42	0.926
LightGBM	0.977	0.976	**0.923**	0.898	0.893	**1.04**	0.921
MLP	0.970	0.969	0.881	0.891	0.866	1.78	0.898
CNN-LSTM	0.970	0.969	0.877	0.902	0.872	1.96	0.902
XGBoost (5-window lag)	**0.978**	**0.977**	0.919	**0.911**	**0.901**	1.15	**0.932**

The ordering is supported statistically, but only partly. The Friedman test over the seven test days rejects equal performance ($\chi^2(6) = 34.0$, $p = 6.8 \cdot 10^{-6}$), and the Nemenyi critical difference of 3.41 (Fig. 9) separates the lagged XGBoost model (average rank 1.29) from logistic regression (7.00), the MLP (5.57) and the CNN–LSTM (5.00). It does not separate it from the other three tree ensembles (ranks 2.86–3.29), which form one statistical group. The paired block-bootstrap comparison is more sensitive (Table 15): after Holm correction the lagged model improves on XGBoost by 0.7 points (95 % interval

0.2–1.2), on LightGBM by 0.9 points and on the random forest by 1.5 points, while XGBoost and LightGBM are indistinguishable (Holm-adjusted p = 1.00) and XGBoost and the random forest are not significantly different (p = 0.41). Two practical conclusions follow. Temporal context is worth a small but reliable gain, and, given tabular features that already encode the physics, the two neural networks have no advantage over boosting: the MLP and the CNN–LSTM are 2.4 and 1.9 points below XGBoost in the seed mean, with a larger seed variance (standard deviation 0.009 for the MLP against 0.001 for XGBoost). The per-day macro-F1 of the best model ranges from 0.845 (day 23) to 0.950 (day 26), which reflects how many attack bursts of each type fall on a given day and shows that a single aggregate hides a ten-point day-to-day swing.

TABLE XV. PAIRED BLOCK-BOOTSTRAP DIFFERENCES IN MACRO-F1 (PAIRS INVOLVING XGBOOST OR ITS LAGGED VARIANT; ALL 21 PAIRS ARE IN THE SUPPLEMENTARY MATERIAL), WITH HOLM-ADJUSTED P-VALUES OVER ALL 21 COMPARISONS (2 000 BLOCK-BOOTSTRAP RESAMPLES OF THE SEED-0 PREDICTIONS). FRIEDMAN TEST ON THE PER-DAY MACRO-F1 OF THE SEVEN TEST DAYS: $X^2(6) = 34.0$, P = 6.8E-06; AVERAGE RANKS (1 = BEST): XGBOOST (5-WINDOW LAG) 1.29, LIGHTGBM 2.86, XGBOOST 3.00, RANDOM FOREST 3.29, CNN-LSTM 5.00, MLP 5.57, LOGISTIC REGRESSION 7.00; NEMENYI CRITICAL DIFFERENCE AT A = 0.05 IS 3.41.

Pair (A − B)	Δ macro-F1	95 % CI	p (raw)	p (Holm)
Logistic regression − XGBoost	-0.159	[-0.179, -0.139]	0.0005	0.0105
Logistic regression − XGBoost (5-window lag)	-0.166	[-0.187, -0.145]	0.0005	0.0105
XGBoost − CNN-LSTM	+0.016	[+0.007, +0.025]	0.0005	0.0105
MLP − XGBoost (5-window lag)	-0.023	[-0.034, -0.013]	0.0005	0.0105
CNN-LSTM − XGBoost (5-window lag)	-0.023	[-0.032, -0.015]	0.0005	0.0105
XGBoost − XGBoost (5-window lag)	-0.007	[-0.012, -0.002]	0.0020	0.0220
Random forest − XGBoost (5-window lag)	-0.015	[-0.025, -0.006]	0.0030	0.0300
XGBoost − MLP	+0.016	[+0.006, +0.027]	0.0040	0.0360
LightGBM − XGBoost (5-window lag)	-0.009	[-0.015, -0.002]	0.0060	0.0480
Random forest − XGBoost	-0.008	[-0.017, +0.001]	0.0690	0.4140
XGBoost − LightGBM	+0.001	[-0.006, +0.009]	0.7390	1.0000

TABLE XVI. PER-DAY BEHAVIOUR ON THE SEVEN TEST DAYS (SEED 0): AVERAGE FRIEDMAN RANK (1 = BEST; CRITICAL DIFFERENCE 3.41) AND THE MINIMUM, MEDIAN, MAXIMUM AND STANDARD DEVIATION OF THE DAILY MACRO-F1.

Model	Average rank	Min	Median	Max	SD
XGBoost (5-window lag)	1.29	0.845	0.909	0.950	0.035
LightGBM	2.86	0.842	0.904	0.937	0.036
XGBoost	3.00	0.843	0.888	0.945	0.035
Random forest	3.29	0.863	0.893	0.918	0.024
CNN-LSTM	5.00	0.832	0.890	0.914	0.028
MLP	5.57	0.821	0.873	0.901	0.027
Logistic regression	7.00	0.688	0.747	0.808	0.041

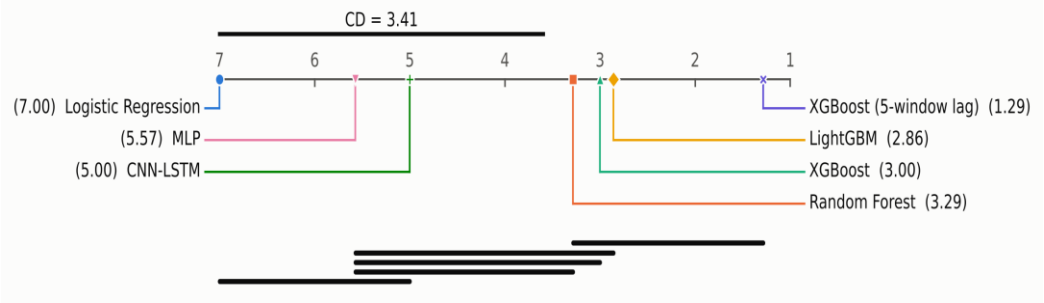


Fig. 9. Critical-difference diagram of the average rank of each model over the seven test days (rank 1 is best). Models joined by a bar are not significantly different at α = 0.05 (Nemenyi critical difference 3.41).

Training behaviour. Fig. 10 shows how the models were fitted. The validation macro-F1 of the networks peaks early and then fluctuates: the MLP reaches its best value (0.831) at epoch 2 and stops after epoch 8, whereas the CNN–LSTM peaks at epoch 14 (0.846) of 20 although its validation loss has been rising since epoch 4 (0.273 to 0.427) while the training loss keeps falling to 0.04. The loss and the F1 therefore disagree about when to stop, and early stopping on the macro-F1, the quantity of interest, restores the epoch that is best for detection even though it is no longer best for the cross-entropy. The random search separates the model families more clearly than it separates the settings within a family: the spread of the validation macro-F1 over the trials is 0.003 for logistic regression, 0.021 for XGBoost, 0.016 for the MLP and 0.065 for the random forest, whose shallow and wide settings are clearly worse. The validation scores of the selected models (0.72 to 0.86) are lower than their test scores (Table 8) because the three validation days contain fewer bursts of the hardest classes, which is another reason to report the test block with a block-bootstrap interval rather than to compare it with the validation score.

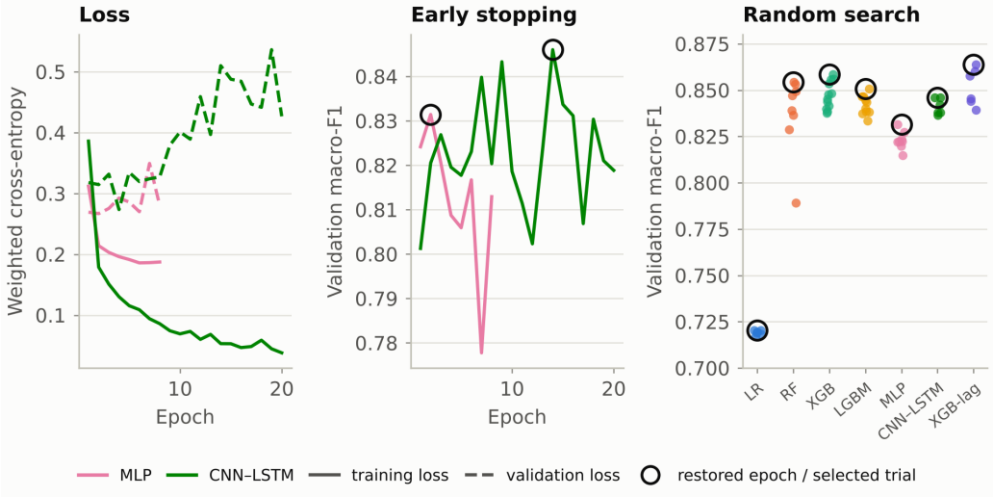


Fig. 10. Training diagnostics. Left: training (solid) and validation (dashed) loss of the two networks for seed 0. Centre: validation macro-F1 per epoch; the ringed epoch is restored by early stopping. Right: validation macro-F1 of the random-search trials of every model; the ringed trial is the selected one.

7.2 Per-Class Results and Error Anatomy

Table 17 breaks the result down by class. Five of the six attack classes are detected with an F1 between 0.929 and 0.978 by the best model: Flood 0.978, Recon 0.973, Amplification 0.964, Replay 0.953 and MITM 0.929. The remaining class, false-data injection, has an F1 of 0.609, and it is the reason why the macro-F1 is 0.91 whereas the accuracy is 0.978. All seven models rank the classes in the same way, so the difficulty is a property of the data rather than of a particular learner.

TABLE XVII. PER-CLASS F1 ON THE CHRONOLOGICAL TEST SET (SEED MEAN). BEST PER CLASS IN BOLD.

Model	Normal	FDIA	Flood	Amplification	MITM	Replay	Recon
Logistic regression	0.900	0.183	0.960	0.960	0.564	0.692	0.968
Random forest	0.983	0.523	0.974	**0.967**	0.923	0.929	**0.975**
XGBoost	0.986	0.585	0.973	0.963	0.919	0.951	0.972
LightGBM	**0.987**	0.581	0.975	0.965	0.912	0.952	0.973
MLP	0.983	0.552	0.963	0.948	0.851	0.913	0.970
CNN-LSTM	0.983	0.541	0.965	0.947	0.865	0.944	0.970
XGBoost (5-window lag)	0.987	**0.609**	**0.978**	0.964	**0.929**	**0.953**	0.973
Test windows	17,532	397	524	389	422	398	498

The confusion matrix (Fig. 11, Table 18) shows that the errors have a simple structure. Every one of the 183 missed FDIA windows of the seed-0 model is classified as Normal, none as another attack; conversely, 93 Normal windows are flagged as FDIA, which gives an FDIA precision of 0.686 and a recall of 0.539. MITM and Replay lose 8.5 % and 5.3 % of their windows to Normal, and almost nothing is confused between two different attacks. Flood, Amplification and Recon are

essentially error-free: their recall exceeds 99.6 % on both transports (Table 19), which confirms that a detector that sees both layers does not rely on the transport to identify the class.

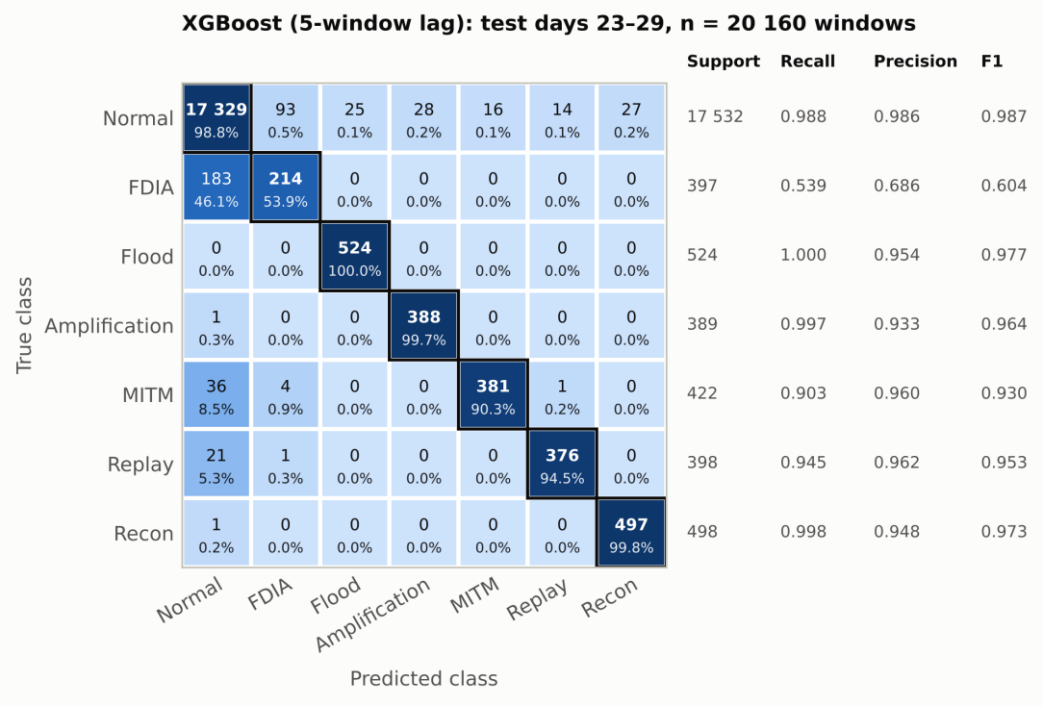


Fig. 11. Confusion matrix of the best model on the chronological test block (seed 0). Each cell gives the number of windows and, below it, the share of the true class; the colour encodes that share on a square-root scale, and the diagonal is outlined. The columns on the right give support, recall, precision and F1 of each class.

The logistic-regression reference makes errors of a different kind: it flags 17.5 % of the Normal windows (12.2 % as FDIA, 3.2 % as MITM and 1.6 % as Replay) while recalling 64.7 % of the FDIA windows, so that its higher FDIA recall is bought with a false-alarm rate that is fifteen times higher than that of the best model, and its FDIA precision is only 0.107.

TABLE XVIII. PER-CLASS TEST RESULTS OF THE BEST MODEL (XGBOOST (5-WINDOW LAG); MEAN OF FIVE SEEDS): SUPPORT, PRECISION, RECALL, F1, AVERAGE PRECISION (AP) AND ONE-VS-REST ROC-AUC.

Class	Windows	Precision	Recall	F1	AP	ROC-AUC
Normal	17,532	0.986	0.989	0.987	0.992	0.971
FDIA	397	0.693	0.544	0.609	0.585	0.855
Flood	524	0.957	1.000	0.978	0.998	1.000
Amplification	389	0.931	0.999	0.964	0.999	1.000
MITM	422	0.959	0.900	0.929	0.971	0.998
Replay	398	0.962	0.945	0.953	0.983	0.997
Recon	498	0.948	0.998	0.973	0.997	1.000
Macro mean	20,160	0.919	0.911	0.913	0.932	0.974

TABLE XIX. RECALL BY ATTACK AND TRANSPORT (BEST MODEL, TEST SET).

Attack	Transport	Windows	Recall (%)
FDIA	both	397	53.9
Flood	MQTT	255	100.0
Flood	COAP	269	100.0
Amplification	COAP	389	99.7
MITM	MQTT	205	89.3
MITM	COAP	217	91.2

Attack	Transport	Windows	Recall (%)
Replay	MQTT	210	95.2
Replay	COAP	188	93.6
Recon	MQTT	222	100.0
Recon	COAP	276	99.6

Which features separate the classes is visible in the raw distributions (Fig. 12). The balance bias is elevated for FDIA and for Replay, whose stale payloads disagree with the plant, but it is almost silent for the traffic attacks; the MQTT message rate separates Flood from every other class; the ratio of CoAP response to request bytes is large only during Amplification; the TTL deviation is non-zero for MITM and, to a lesser degree, for Replay; and wildcard subscriptions occur almost only in Recon windows. Each attack therefore has a primary observable that is absent from most other classes, and the overlap that remains, for example between the multi-scale bias of the normal background and of FDIA, is what the classifiers have to resolve.



Fig. 12. Distributions of six discriminating features by class (all windows of days 3–29; values on a signed log(1+x) scale, box = interquartile range with median, whiskers = 5th and 95th percentile). N Normal, F FDIA, Fl Flood, Am Amplification, MI MITM, Re Replay, Rc Recon.

The FDIA result is explained by the injection variant (Table 20, Fig. 13 (left panel)). The biased variant V1 is recognised in 80.7 % of its windows and the slowly growing ramp V2 in 64.1 %. The balanced variant V3, which preserves the energy balance exactly, is never recognised by the best model: all 105 test windows are labelled Normal (the single-window XGBoost and the random forest flag 4 % and 10 % of them, Section 7.7). Excluding V3 the recall on FDIA would be 73.3 %, and the model's FDIA performance is therefore bounded from above by the share of V3 in the data (26 % of the FDIA windows here) rather than by the learner. This is the theoretical behaviour of 7: the single-feature analysis in Table 21 shows that the balance-residual features that separate V1 (AUC 0.955) and V2 (AUC 0.828) from Normal are at chance level for V3 (AUC 0.51), and that the only weakly informative features for V3 are the peer-comparison and physical-consistency ones (best AUC 0.69), which are also what a sensor fault excites. A V3-aware detector therefore needs information that the 5 s control-centre stream does not carry, a point we return to in Section 8.

TABLE XX. FDIA RECALL BY INJECTION VARIANT (BEST MODEL, TEST SET). V3 SATISFIES THE BALANCE CONSTRAINT EXACTLY AND IS NOT SEPARABLE FROM THE PHYSICAL RESIDUALS.

Variant	Windows	Recall (%)	Predicted Normal (%)
V1-bias	161	80.7	19.3
V2-ramp	131	64.1	35.9
V3-balanced	105	0.0	100.0

TABLE XXI. BEST SINGLE-FEATURE SEPARABILITY (ROC AUC AGAINST NORMAL WINDOWS) FOR EACH FDIA VARIANT, WITH THE THREE STRONGEST FEATURES.

Variant	Windows	Best	Second	Third
V1-bias	818	e_bal_bias (0.955)	e_bal_mean (0.946)	e_bal_max (0.938)
V2-ramp	525	e_bal_bias (0.828)	e_bal_mean (0.823)	e_phys_m (0.810)
V3-balanced	471	e_peer_bias (0.691)	e_peer (0.672)	e_peer_bias10 (0.669)

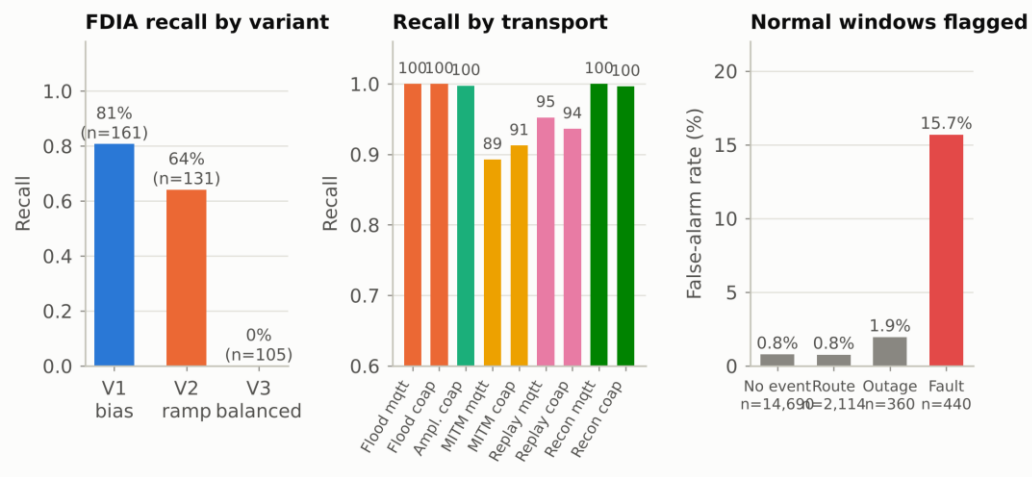


Fig. 13. Error anatomy of the best model. Left: FDIA recall by injection variant. Centre: recall of each attack by transport. Right: share of Normal windows flagged, by benign context (the number of windows is given with each label).

The false alarms are concentrated in one benign context (Table 22). On windows with no benign event the false-alarm rate is 0.77 %, with a link outage and burst flush it is 1.94 %, after a route change 0.76 %, and for the two rare events (client reconnects and administrator discovery, 13 and 9 windows) none was flagged. Windows that contain a sensor fault (a stuck value, a spike or a drift) are flagged 15.7 % of the time, twenty times the baseline rate, because a faulty sensor produces the same signature, a persistent residual in one channel, as the biased FDIA variants. The overall Normal false-alarm rate is 1.15 %, which at 120 windows per hour would be one alarm every 43 minutes if it were spread evenly; since the alarms are concentrated in fault episodes they are in practice bursty and can be deduplicated, but the figure is high enough to matter for an operator. At the burst level (Table 23) every Flood, Amplification, MITM, Replay and Recon burst of the test block is detected at least once, with a median delay of at most one 30 s window and a 90th percentile of at most 102 s (MITM); only 27 of the 38 FDIA bursts (71 %) are detected, with a median delay of 30 s and a 90th percentile of 174 s.

TABLE XXII. FALSE-ALARM RATE ON NORMAL WINDOWS BY BENIGN CONTEXT (BEST MODEL, TEST SET).

Benign context	Windows	False alarms (%)
No benign event	14,690	0.77
Link outage + flush	360	1.94
Route change	2,114	0.76
Client reconnect	13	0.00
Admin discovery	9	0.00
Sensor fault	440	15.68

TABLE XXIII. BURST-LEVEL DETECTION (BEST MODEL, TEST SET): FRACTION OF ATTACK BURSTS DETECTED AT LEAST ONCE, AND THE DELAY BETWEEN BURST ONSET AND THE FIRST ALARM (30 S WINDOWS). DELAY STATISTICS ARE COMPUTED OVER THE DETECTED BURSTS.

Attack	Bursts detected (%)	Median delay (s)	P90 delay (s)	Bursts	Missed
FDIA	71	30	174	38	11
Flood	100	0	30	40	0
Amplification	100	0	30	36	0
MITM	100	30	102	33	0
Replay	100	0	30	33	0
Recon	100	0	30	46	0

7.3 Calibration

Probabilities that are used to trigger an explanation or an escalation must also be calibrated. Fig. 14 (right panel) shows the reliability of the top-label probability of the best model, whose expected calibration error (ECE, 15 equal-width bins, five-seed mean) is 0.037 and whose multi-class Brier score is 0.042. LightGBM is the best calibrated model (ECE 0.007, Brier 0.039), XGBoost has an ECE of 0.043 and the neural networks 0.016–0.025. The class-weighted random forest is badly calibrated (ECE 0.244) because the class weights shift its vote fractions towards the minority classes, and logistic regression, which is also class-weighted, has an ECE of 0.078 and a Brier score of 0.255. The precision–recall curves in Fig. 14 (left panel) confirm the picture of the confusion matrix: five curves hug the top-right corner while the FDIA curve falls steeply, with an average precision of 0.59 against 0.97–1.00 for the other six classes.

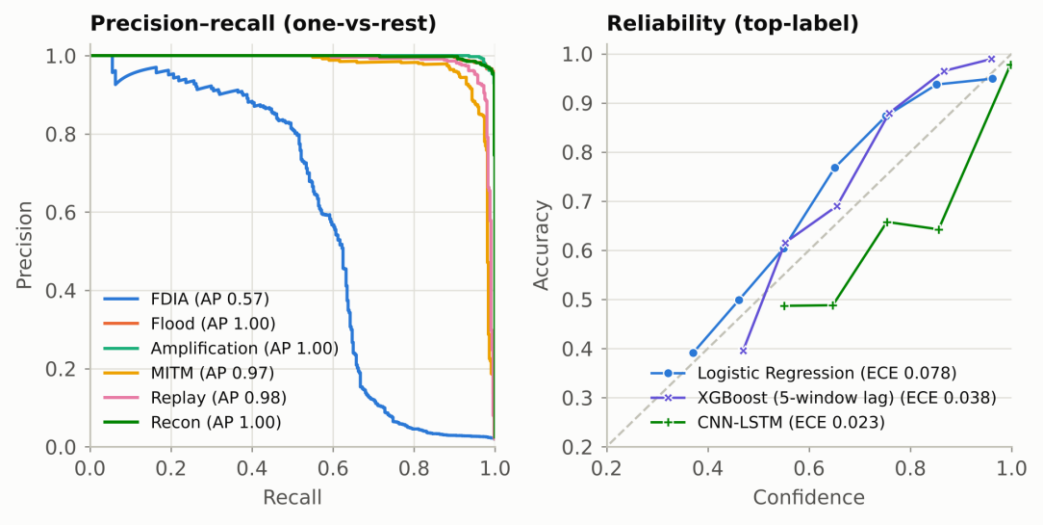


Fig. 14. Left: one-vs-rest precision–recall curves of the best model on the test block. Right: reliability diagram of the top-label probability (seed-0 models) for logistic regression, the lagged XGBoost model and the CNN–LSTM; the dashed line is perfect calibration.

7.4 Evaluation protocol and leakage

Table 24 and Fig. 15 repeat the experiment for the two strongest base learners under three protocols. With a stratified random split of the same windows the macro-F1 rises by 1.2 points for the random forest and by 2.7 points for XGBoost, and the minority-class F1 rises much more: from 0.583 to 0.716 for XGBoost and from 0.524 to 0.623 for the random forest. A random split therefore reports 13 additional points of FDIA F1 for the boosted model that the chronological protocol does not support. The reason is visible in the last column of the table. Under a random split the median test window has a training window as its immediate neighbour (the median temporal gap is one window, i.e. 30 s), so that a classifier can recognise a burst by interpolating between its own neighbours; under the chronological protocol the median gap is 6.5 days. The burst-grouped five-fold split is a harsher test of burst memorisation than a random split (the median gap is 16 windows, i.e. 8 min, because Normal windows are grouped by hour), and its results are within about one standard deviation of the chronological results (−0.7 points for the random forest and +0.4 points for XGBoost). We do not interpret the grouped folds as a replacement for the chronological split, because each fold mixes days and has a different class composition, but they show that the largest inflation comes from within-burst neighbours rather than from calendar effects. The inflation we measure is moderate because each burst lasts only 2–10 min (4–20 windows) and the traffic of every burst

is regenerated, which bounds how much a model can memorise; a data set with a few long captures would be exposed to far more near-duplicate windows.

TABLE XXIV. EVALUATION PROTOCOL AND OPTIMISM. Δ IS THE MACRO-F1 DIFFERENCE TO THE CHRONOLOGICAL PROTOCOL IN PERCENTAGE POINTS; THE LAST COLUMN IS THE MEDIAN TEMPORAL GAP BETWEEN A TEST WINDOW AND THE NEAREST TRAINING WINDOW (30 S WINDOWS), A DIRECT MEASURE OF HOW CLOSE TEST SAMPLES LIE TO TRAINING SAMPLES WITHIN THE SAME BURST.

Model	Protocol	Macro-F1	MCC	FDIA F1	Δ (pp)	Median gap to nearest training window
Random forest	Chronological (proposed)	0.896 ± 0.001	0.879	0.524	+0.0	18,744 windows (6.5 days)
Random forest	Burst-grouped K-fold	0.889 ± 0.009	0.860	0.567	-0.7	16 windows (8 min)
Random forest	Random stratified split	0.908 ± 0.004	0.883	0.623	+1.2	1 window (30 s)
XGBoost	Chronological (proposed)	0.907 ± 0.002	0.898	0.583	+0.0	18,744 windows (6.5 days)
XGBoost	Burst-grouped K-fold	0.911 ± 0.010	0.895	0.645	+0.4	16 windows (8 min)
XGBoost	Random stratified split	0.933 ± 0.001	0.920	0.716	+2.7	1 window (30 s)

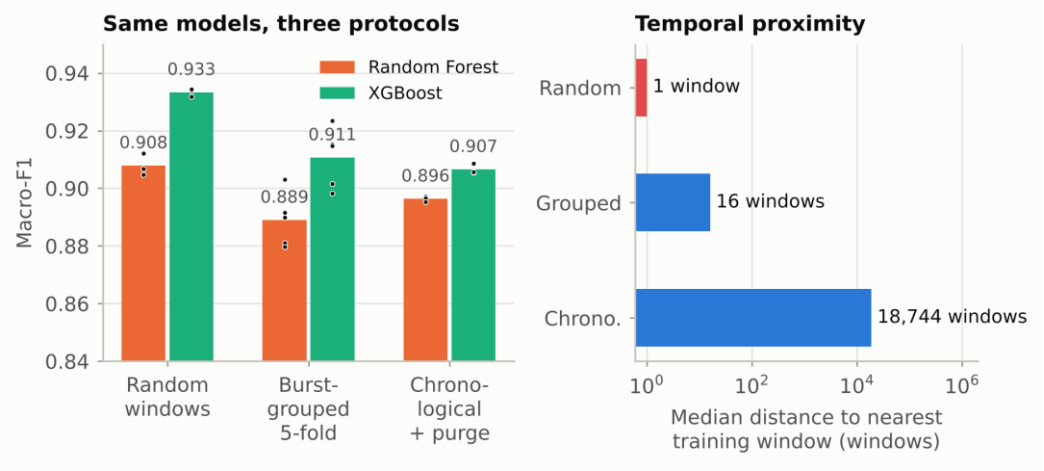


Fig. 15. Left: macro-F1 of the random forest and XGBoost under the chronological, burst-grouped and random protocols (mean and standard deviation over seeds or folds). Right: median temporal distance of a test window to the nearest training window, on a logarithmic axis.

7.5 Feature-group ablation

Table 25 and Fig. 16 quantify what each of the eight feature groups contributes. Removing a single network group changes the macro-F1 by at most 0.1 points, with one exception: removing the temporal-integrity group (RTT, TTL, duplicate identifiers, sequence gaps, clock skew) costs 5.6 points, and it is the only group whose loss the remaining features cannot compensate. The loss is concentrated in the two classes that are defined by temporal anomalies: the F1 of MITM falls by 28.6 points and that of Replay by 11.3 points (Fig. 16 (left panel)). Removing the multi-scale bias group costs 6.1 F1 points on FDIA, and removing the residual group 2.6 points on FDIA and 3.1 points on MITM. The other network groups are mutually redundant (traffic volume and source dynamics each carry about 0.66 macro-F1 on their own, CoAP semantics 0.52), which is why none of them is individually indispensable. The energy features matter for a different reason. The network-only model reaches 0.817 and has an FDIA F1 of exactly 0, whereas the energy-only model reaches 0.338 overall but has an FDIA F1 of 0.514 and, interestingly, a MITM F1 of 0.598 because payload tampering is visible in the residuals. Adding the two groups of energy features to the network features therefore raises the macro-F1 by 9.0 points almost entirely through the FDIA class, and removing only the multi-scale bias group costs a further 0.9 points (FDIA F1 from 0.583 to 0.523). The two time-of-day features are not useful in this data set: removing them changes the macro-F1 by +0.2 points, which is within seed noise. We keep them in the feature set because they are cheap and because the commissioning calibration is done per time of day, but the result shows that the classifier does not exploit the daily schedule of the generated attacks.

TABLE XXV. FEATURE-GROUP ABLATION WITH XGBOOST (MACRO-F1, THREE SEEDS). "LEAVE-OUT" REMOVES THE GROUP; "ONLY" TRAINS ON THE GROUP ALONE. Δ IN PERCENTAGE POINTS RELATIVE TO THE FULL MODEL.

Feature group	Features	Leave-out	Δ (pp)	Only
All 68 features	68	0.907	—	—
Traffic volume	14	0.905	-0.1	0.657
Source dynamics	8	0.906	0.0	0.659
Temporal integrity	12	0.851	-5.6	0.478
MQTT semantics	6	0.905	-0.1	0.384
CoAP semantics	5	0.906	0.0	0.519
Energy residuals	13	0.900	-0.7	0.310
Energy bias (multi-scale)	8	0.897	-0.9	0.261
Context	2	0.909	+0.2	0.133
Network features only		0.817	-9.0	—
Energy features only		0.338	-56.8	—
Without time-of-day		0.909	+0.2	—
Without multi-scale bias		0.897	-0.9	—

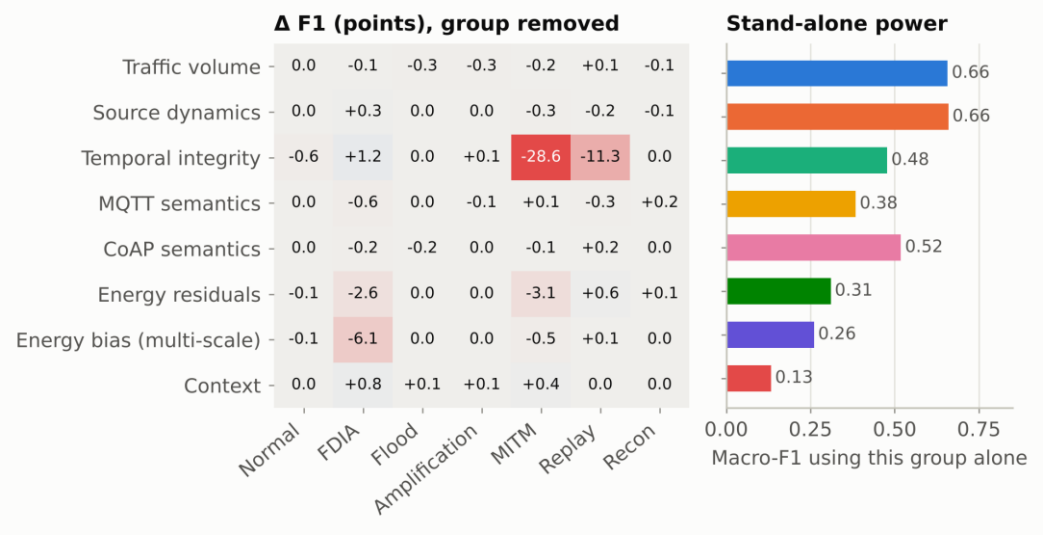


Fig. 16. Left: change in per-class F1 when one feature group is removed from the XGBoost model. Right: macro-F1 of XGBoost trained with only that group.

7.6 Class-Imbalance Handling

The Normal class has 17 532 test windows and each attack class 389–524, so imbalance is a first-order issue (Table 26, Fig. 17). For XGBoost, sample weights proportional to the inverse square root of the class frequency raise the macro-F1 from 0.897 to 0.908, and SMOTE raises it to 0.911, but SMOTE increases the training set from 48 936 to 295 400 windows and the fit time from 6 s to 116 s, a twenty-fold cost relative to weighting for 0.3 additional points. Random undersampling to the size of the smallest class is clearly harmful (0.830): the FDIA F1 falls to 0.263 because the precision collapses to 0.166. Weights and SMOTE trade precision for recall on FDIA differently: weighting raises the FDIA recall from 0.43 to 0.55 and lowers the precision from 0.82 to 0.64, whereas SMOTE keeps the precision at 0.81 and raises the recall only to 0.47. For the random forest no strategy helps (class weights 0.896 and SMOTE 0.897 against 0.900 with no handling), and undersampling lowers the macro-F1 to 0.848. The combination of SMOTE and weights was not better than SMOTE alone for XGBoost. We therefore use inverse-square-root weights in all main experiments, since they deliver nearly all the benefit at no computational cost, and treat the choice between weights and SMOTE as a precision–recall operating-point decision rather than as an accuracy decision.

TABLE XXVI. CLASS-IMBALANCE HANDLING (CHRONOLOGICAL TEST SET). NUMBER OF TRAINING WINDOWS AFTER RESAMPLING; MACRO-F1 OVER FIVE SEEDS; FDIA METRICS ARE THE MINORITY-CLASS F1, RECALL AND PRECISION. SMOTE WITH CLASS WEIGHTS WAS RUN FOR XGBOOST ONLY, AND THE RANDOM FOREST IS LIMITED TO 200 TREES IN THIS EXPERIMENT.

Model	Strategy	Train windows	Macro-F1	FDIA F1	FDIA recall	FDIA precision	Fit (s)
XGBoost	No handling	48,936	0.897 ± 0.001	0.565	0.432	0.817	3
XGBoost	Class weights	48,936	0.908 ± 0.002	0.592	0.552	0.639	6
XGBoost	SMOTE	295,400	0.911 ± 0.001	0.597	0.474	0.806	116
XGBoost	Random undersampling	6,566	0.830 ± 0.005	0.263	0.646	0.166	3
XGBoost	SMOTE + class weights	295,400	0.911 ± 0.001	0.597	0.474	0.806	114
Random forest	No handling	48,936	0.900 ± 0.002	0.606	0.469	0.858	26
Random forest	Class weights	48,936	0.896 ± 0.000	0.520	0.591	0.464	28
Random forest	SMOTE	295,400	0.897 ± 0.000	0.512	0.604	0.445	228
Random forest	Random undersampling	6,566	0.848 ± 0.008	0.329	0.631	0.224	2

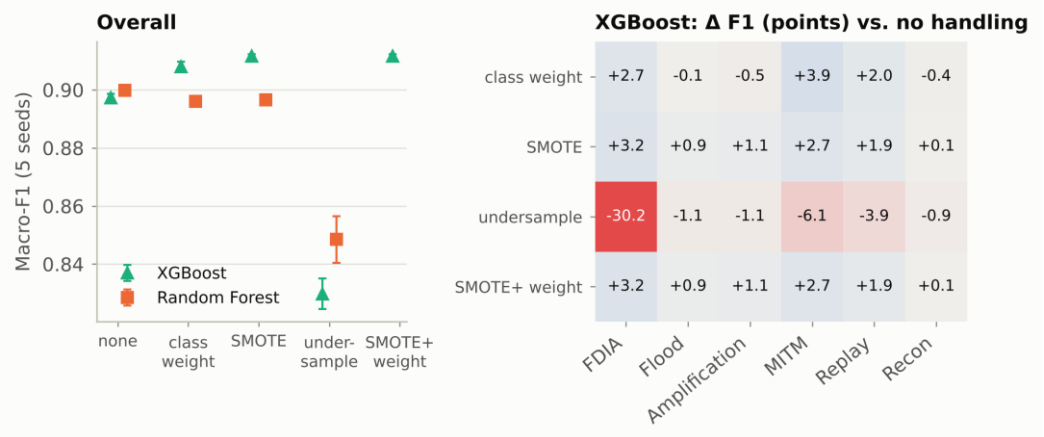


Fig. 17. Left: macro-F1 of XGBoost and the random forest under five imbalance strategies (mean ± standard deviation of five seeds). Right: change in per-class F1 of XGBoost relative to training without any handling.

7.7 Generalisation to unseen FDIA variants

A detector trained on the injections an operator can imagine will meet others. Table 27 and Fig. 19 (left panel) remove one FDIA variant at a time from the training set (the other two stay in) and measure the recall on the variant that was removed. The loss is moderate. For XGBoost, V2 keeps 94 % of its recall (64.6 % when seen, 61.1 % when held out) and V1 keeps 82 % (81.0 % to 66.7 %); for the random forest the retained shares are 92 % and 96 %. The overall macro-F1 changes by at most 1.1 points. The two unbalanced variants share the same physical signature, a persistent signed residual in the balance and in the resource-consistency checks, which is why a model trained on a bias generalises to a ramp and the reverse, but the 14-point loss of XGBoost on V1 shows that the learned decision boundary is closer to the training variants than the physics requires. The balanced variant V3 is detected at 4.1 % (XGBoost) and 10.2 % (random forest) even when it is in the training data, and at 3.2 % and 1.3 % when it is held out, so that for this variant training data make no practical difference, which is consistent with the single-feature analysis in Section 7.2.

Table XXVII. Leave-one-variant-out FDIA generalisation (chronological test block, three seeds). “Seen” is the recall of the variant when all three variants are in the training data; “held out” when that variant is removed from the training set (the other two remain). Retained is the held-out recall as a percentage of the seen recall; the last two columns give the overall macro-F1.

Model	Variant	Seen recall (%)	Held-out recall (%)	Δ (pp)	Retained (%)	Macro-F1 seen	Macro-F1 held out
XGBoost	V1-bias	81.0	66.7	-14.3	82	0.907	0.902
XGBoost	V2-ramp	64.6	61.1	-3.6	94	0.907	0.904
XGBoost	V3-balanced	4.1	3.2	-1.0	—	0.907	0.908
Random forest	V1-bias	82.2	78.9	-3.3	96	0.896	0.885

Model	Variant	Seen recall (%)	Held-out recall (%)	Δ (pp)	Retained (%)	Macro-F1 seen	Macro-F1 held out
Random forest	V2-ramp	70.0	64.4	-5.6	92	0.896	0.896
Random forest	V3-balanced	10.2	1.3	-8.9	—	0.896	0.902

7.8 Evasion by attack-intensity scaling

An adaptive adversary can weaken an attack until it falls below the detector's operating point. We emulate this in the problem space [10]: for each scale $s \in \{1,0.5,0.25,0.1,0.05\}$ the whole simulation is regenerated with the message rate of Flood, Amplification and Recon, the injected bias of FDIA, the added delay and the tampering amplitude of MITM, and the share of tampered (MITM) or replayed (Replay) messages multiplied by s, the schedule of bursts and the benign events being identical (the random draws are unaffected by the scale), and the models trained at full intensity on site A are evaluated on days 23–29 (Table 28, Fig. 18). At $s = 1$ the regenerated test set reproduces the main benchmark (XGBoost 0.905 against 0.907), which confirms that the regeneration pipeline is consistent.

The degradation is smooth and steep. Halving the intensity costs XGBoost 4.8 points of macro-F1 (0.857), a quarter costs 15.4 points (0.751), and at one tenth the model is at 0.600. The Normal recall stays between 98.5 % and 99.0 % at every scale, so the loss comes entirely from missed attack windows and not from additional false alarms. The classes degrade at very different speeds (Fig. 18 (right panel)). FDIA recall falls from 55 % to 35 % at half intensity and to 1 % at one tenth, which is expected as a bias of a few percent of the rating disappears into the noise of the residual. Replay is the most fragile network attack (recall 94 %, 76 %, 47 %, 20 %, 7 %) because the fraction of replayed messages directly sets how many of the 30 s window's samples are inconsistent, whereas Flood and Amplification, whose rate is multiplied by s, are still detected at 85 % at a quarter and 52 % at one twentieth of the original rate, and Recon falls from 85 % to 36 % between a quarter and one twentieth. The ranking of the models also changes: the boosted trees are best down to $s = 0.25$, but at $s \leq 0.1$ the CNN–LSTM (0.611 and 0.551) is as good as or better than XGBoost (0.600 and 0.510) and the random forest is the worst (0.561 and 0.474). A model that is a sharper detector at full intensity is therefore not necessarily the more robust one when the attack is weakened. We note that scaling is not free evasion for the attacker: a flood at a tenth of the rate or a bias at a tenth of the amplitude also has a tenth of the effect, so the figures describe a trade-off between impact and detectability, and give the operator a quantitative picture of the intensity below which the detector can no longer be relied upon. If a recall of 85 % is the requirement, Flood, Amplification and Recon meet it down to a quarter of the training intensity, MITM (90 % at full intensity, 80 % at half) and Replay (94 % and 76 %) fall below it between full and half intensity, and FDIA never reaches it.

TABLE XXVIII. EVASION BY ATTACK-INTENSITY SCALING (PROBLEM-SPACE REGENERATION; MODELS TRAINED AT FULL INTENSITY ON SITE A). THE SCALE MULTIPLIES THE MESSAGE RATE OF FLOOD, AMPLIFICATION AND RECON, THE BIAS OF FDIA, THE DELAY AND TAMPERING AMPLITUDE OF MITM, AND THE SHARE OF TAMPERED OR REPLAYED MESSAGES. LEFT: MACRO-F1 OVER DAYS 23–29; RIGHT: RECALL (%) OF XGBOOST FOR SELECTED CLASSES (FLOOD AND AMPLIFICATION, MITM AND REPLAY AVERAGED).

Scale	XGBoost	Random forest	CNN-LSTM	Logistic reg.	FDIA recall	Flood/Amplif. recall	MITM/Replay recall	Recon recall
1	0.905	0.897	0.889	0.747	55	100	92	100
0.5	0.857	0.833	0.832	0.720	35	98	78	97
0.25	0.751	0.710	0.734	0.668	15	85	57	85
0.1	0.600	0.561	0.611	0.596	1	66	34	57
0.05	0.510	0.474	0.551	0.542	1	52	27	36

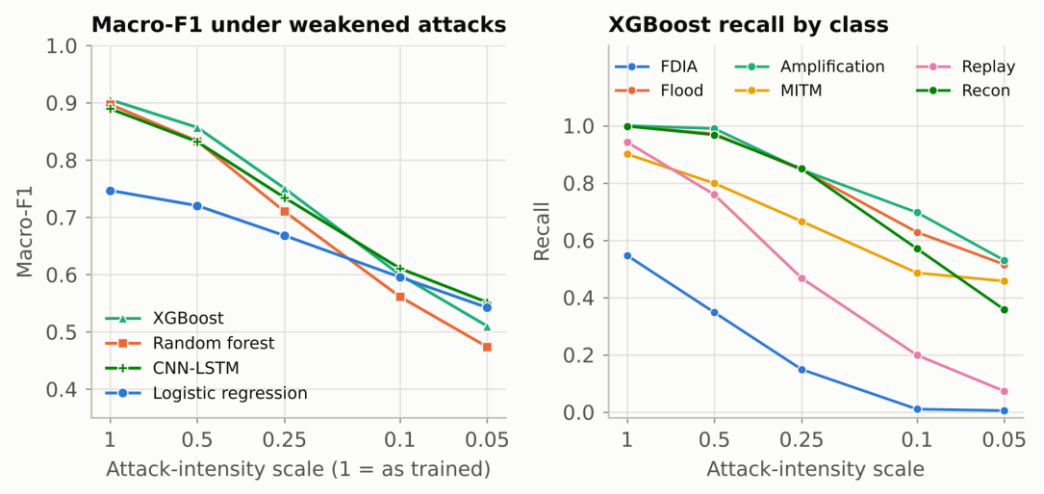


Fig. 18. Evasion by attack-intensity scaling. Left: macro-F1 of four models trained at full intensity, as the attack intensity is scaled down in the regenerated traffic. Right: recall of XGBoost per attack class.

7.9 Cross-site transfer and few-shot adaptation

Site B differs from site A in climate, ratings, link quality, clock accuracy and benign-event rates (Section 6). Applying the site-A models unchanged, with site-A calibration, lowers the macro-F1 of XGBoost from 0.907 to 0.785 and that of the random forest from 0.896 to 0.746 (Table 29, Fig. 19 (centre panel)); the CNN–LSTM loses less (0.861) and logistic regression, which started lower, loses little (0.744). The three volumetric and reconnaissance classes transfer almost perfectly (F1 0.96–0.98 for the tree models), the replay class slightly less (0.93–0.95), and the two classes that rely on temporal-integrity and energy-consistency features fail: the FDIA F1 of XGBoost and the random forest falls from 0.58 and 0.52 to 0.50 and 0.21, and the MITM F1 collapses to 0.23 and 0.25, with a MITM recall of 100 % and a precision of 0.13.

TABLE XXIX. ZERO-SHOT TRANSFER: MODELS TRAINED ON SITE A, TESTED ON DAYS 23–29 OF SITE B (PER-CLASS F1).

Model	Macro-F1	Normal	FDIA	Flood	Amplification	MITM	Replay	Recon
XGBoost	0.785	0.90	0.50	0.97	0.98	0.23	0.95	0.97
Random forest	0.746	0.92	0.21	0.97	0.97	0.25	0.93	0.97
CNN-LSTM	0.861	0.97	0.52	0.96	0.96	0.76	0.88	0.97
Logistic regression	0.744	0.90	0.21	0.92	0.81	0.70	0.70	0.96

The MITM collapse has a single, identifiable cause. We analysed the XGBoost predictions on the 17 231 Normal windows of site B. Of these 17.6 % are flagged, but only 1.6 % of the 13 313 windows without any benign event; the 2 791 windows with a route change are flagged in 93.0 % of the cases, 92.3 % of them as MITM, whereas the same model flags 0.7 % of the route-change windows of site A as MITM. A route change at site B is therefore indistinguishable from a MITM for a model that has learned the site-A thresholds, and the reason is visible in the features: the median RTT of the MQTT and CoAP devices at site B is 1.7 times that at site A (0.079 s against 0.046 s and 0.228 s against 0.135 s), the median clock skew is about twice as large, and route changes are twice as frequent (0.15 against 0.08 per device and day). A route change raises the RTT by 10–40 % and shifts the TTL by one; on top of the larger baseline this moves the window into the region that the site-A model reserves for the MITM delay. The model has learned absolute thresholds of the temporal-integrity features rather than deviations from the local baseline. Sensor-fault windows are also flagged more often at site B (43 %, 23 % of them as MITM) and link outages in 15 % of the cases.

Adding a few days of site-B data is effective and cheap (Table 30, Fig. 19 (right panel)). One day (about 2 800 windows) lifts the macro-F1 of XGBoost from 0.785 to 0.844 and the MITM F1 from 0.23 to 0.55; eight days bring it to 0.900 (MITM F1 0.86), within 0.7 points of the site-A result, whereas the random forest reaches 0.854 (MITM F1 0.64). The FDIA F1 improves only from 0.50 to 0.58 because the limit there is the variant V3, not the site. The transfer experiment thus supports a deployment recipe, to recalibrate the energy features on a short commissioning interval *and* to re-train or fine-tune the temporal-integrity thresholds on a few days of local benign traffic, and it shows what happens otherwise: a detector that was accurate on its own site generates an alarm on one Normal window out of six on another.

TABLE XXX. FEW-SHOT ADAPTATION: SITE-B DAYS ADDED TO THE SITE-A TRAINING SET (MACRO-F1, MITM F1 AND FDIA F1 ON SITE-B DAYS 23–29).

Site-B days	XGB macro-F1	XGB MITM F1	XGB FDIA F1	RF macro-F1	RF MITM F1	RF FDIA F1
0	0.785	0.23	0.50	0.746	0.26	0.20
1	0.844	0.55	0.53	0.827	0.49	0.50
2	0.849	0.59	0.51	0.832	0.50	0.51
4	0.868	0.68	0.56	0.836	0.56	0.49
8	0.900	0.86	0.58	0.854	0.64	0.53

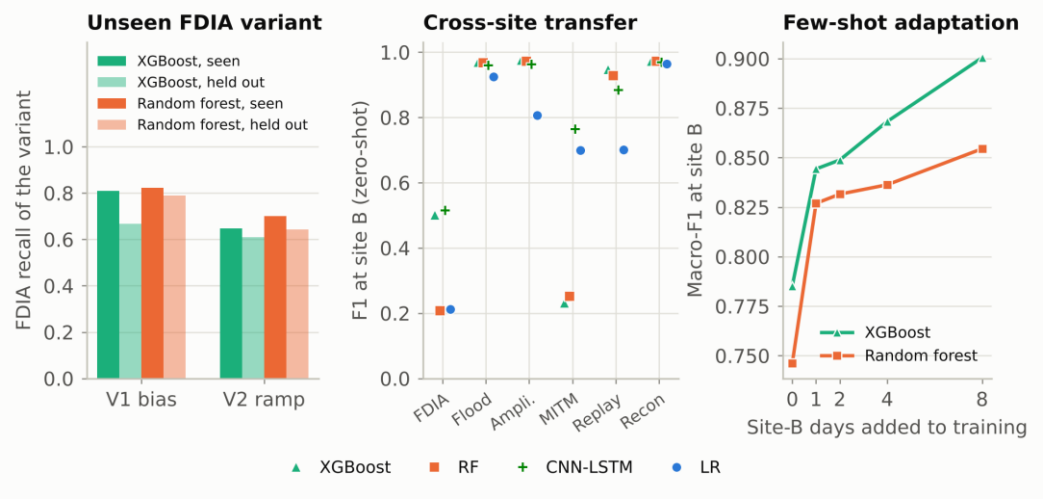


Fig. 19. Left: recall of an FDIA variant when it is seen in training and when it is held out. Centre: F1 per attack class at site B without adaptation (the Normal class is omitted). Right: macro-F1 at site B as days of site-B data are added to the site-A training set.

7.10 Explanations

Explanations are computed for a class-stratified sample of 1 050 test windows (150 per class), on which XGBoost, the tuned single-window model that TreeSHAP explains exactly, has an accuracy of 0.902. The additivity of TreeSHAP holds to numerical precision (the maximum absolute error between the sum of the attributions plus the base value and the model's log-odds is 4.0×10^{-6}).

What the model uses. Fig. 20 (left panel) lists the fifteen features with the largest mean absolute SHAP value for the predicted class. All of them are network features: the number of CoAP source addresses, the TTL deviation of both transports, the number of CoAP source ports, the MQTT subscribe rate, the duplicate rate and the message rate. This reflects the composition of the sample, in which four of the seven classes are visible in the traffic, and it is not evidence that the energy features are unimportant; the class-conditional view in Fig. 20 (right panel) shows the division of labour. Flood is explained by traffic volume (66 % of the absolute attribution) and CoAP semantics (22 %), Amplification by source dynamics (67 %) and traffic volume (23 %), Recon by MQTT and CoAP semantics (38 % and 28 %) and traffic volume (18 %), and MITM and Replay by the temporal-integrity group (70 % each), with the energy residuals contributing a further 18 % to MITM. For FDIA the energy features carry 78 % of the attribution (multi-scale bias 57 %, residuals 21 %), and for Normal windows the attribution is spread over residuals (31 %), traffic (24 %), temporal integrity (15 %) and bias (14 %). Each of these assignments agrees with the mechanism by which the attack was generated, which is what a domain expert would hope to see, but it is also a check on the detector: a model that separated floods by the time of day or MITM by the message size would have been exposed here.

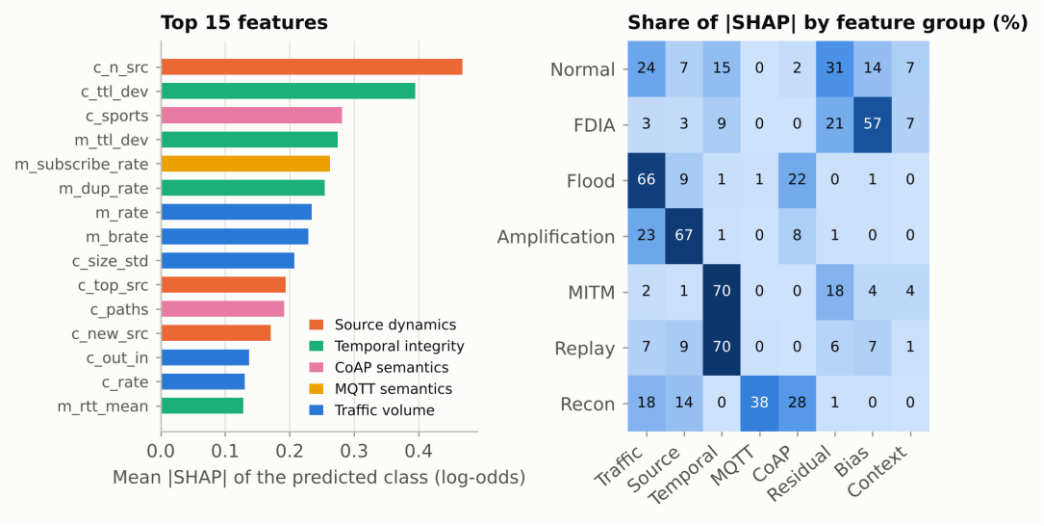


Fig. 20. Global attribution of the XGBoost model on 1 050 test windows (150 per class). Left: fifteen features with the largest mean absolute SHAP value of the predicted class, coloured by feature group. Right: share of the absolute SHAP value that falls on each feature group, by predicted class.

The four case studies in Fig. 21 show the same thing for single windows. An FDIA window with a bias injection (predicted with probability 0.97) is explained by the multi-scale bias of the measurement channels, the peer-comparison bias and the balance-residual bias; a MITM window on MQTT by the TTL deviation (+2.9 log-odds) and the round-trip time (+1.9); a CoAP reconnaissance window by the number of distinct source ports (+3.2) and the dominant-source share (+1.7); and an MQTT flood by the message rate and byte rate (about +2.2 and +2.1). An operator reading these four panels would be directed to the correct equipment and protocol layer each time.

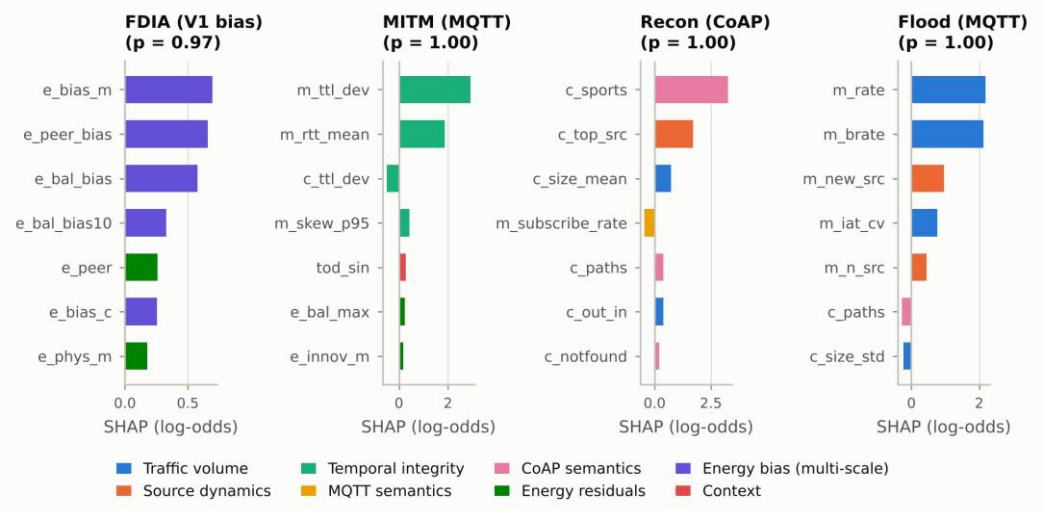


Fig. 21. SHAP explanations of four correctly classified test windows (the highest-confidence correctly classified window of each type in the sample): bars show the seven largest contributions in log-odds to the predicted class, coloured by feature group.

Are the explanations faithful? Table 31 and Fig. 22 report the deletion and insertion tests. For XGBoost, removing the five features that TreeSHAP ranks highest lowers the predicted-class probability by 0.73 (from an average of 0.95), whereas removing five random features lowers it by 0.05; keeping only these five features restores all but 0.10 of the probability, against all but 0.71 for random features. The deletion AUC is 0.22 against 0.51 for a random order. The other rankings are clearly less faithful: global gain importance (deletion AUC 0.32, comprehensiveness 0.23), permutation importance (0.41, 0.16) and our LIME implementation (0.44, 0.09) are closer to the random order than to TreeSHAP, and LIME's local surrogate explains on average only 42 % of the variance of the model's output in the neighbourhood of an instance (the weighted R^2 of the eight-feature ridge model), which is the likely reason. Two cautions apply to this comparison: LIME's sparse surrogate gives non-zero scores to only eight features, so that the ranking beyond them is arbitrary and the AUCs over 68 features favour dense methods, and the two global rankings are by construction class-agnostic whereas the test is

class-specific; the five-feature comprehensiveness and sufficiency are the fairer indicators. The same pattern holds for the random forest (TreeSHAP deletion AUC 0.21 against 0.44 for random) and for the neural networks, where Integrated Gradients is highly faithful (deletion AUC 0.19 for the MLP and 0.12 for the CNN–LSTM against 0.57 and 0.64 for random), which also shows that the lowest-deletion ranking does not have to come from a tree model. The faithfulness of an attribution refers to the model that it explains, so these results do not imply that the models rely on the same features.

TABLE XXXI. FAITHFULNESS OF ATTRIBUTIONS (PREDICTED-CLASS PROBABILITY; MEAN OVER THE EXPLAINED WINDOWS). DELETION AUC (↓) AND INSERTION AUC (↑) ARE AREAS UNDER THE CURVES OVER ALL 68 FEATURES (X-AXIS NORMALISED TO 1); COMPREHENSIVENESS AT 5 FEATURES (↑) IS THE PROBABILITY DROP WHEN THE FIVE TOP-RANKED FEATURES ARE REPLACED BY THE COMMISSIONING MEDIAN, AND SUFFICIENCY AT 5 (↓) THE DROP WHEN ONLY THESE FIVE ARE KEPT. RANDOM RANKING IS THE NULL REFERENCE. THE NETWORKS USE A STANDARDISED-SPACE BASELINE (COMMISSIONING MEAN).

Model	Ranking	Deletion AUC ↓	Insertion AUC ↑	Comp.@5 ↑	Suff.@5 ↓	Windows
XGBoost	TreeSHAP	0.219	0.935	0.728	0.103	1,050
XGBoost	LIME (own implementation)	0.437	0.632	0.094	0.703	1,050
XGBoost	Gain importance (global)	0.316	0.768	0.225	0.673	1,050
XGBoost	Permutation importance (global)	0.412	0.654	0.163	0.656	1,050
XGBoost	Random ranking	0.510	0.528	0.047	0.709	1,050
Random forest	TreeSHAP	0.205	0.823	0.579	0.323	1,050
Random forest	Random ranking	0.436	0.440	0.087	0.646	1,050
MLP	Integrated Gradients	0.187	0.961	0.697	0.074	350
MLP	Random ranking	0.566	0.578	0.038	0.691	350
CNN-LSTM	Integrated Gradients	0.123	0.968	0.732	0.034	350
CNN-LSTM	Random ranking	0.643	0.654	0.027	0.684	350
XGBoost (same windows as the networks)	TreeSHAP	0.249	0.930	0.685	0.092	350

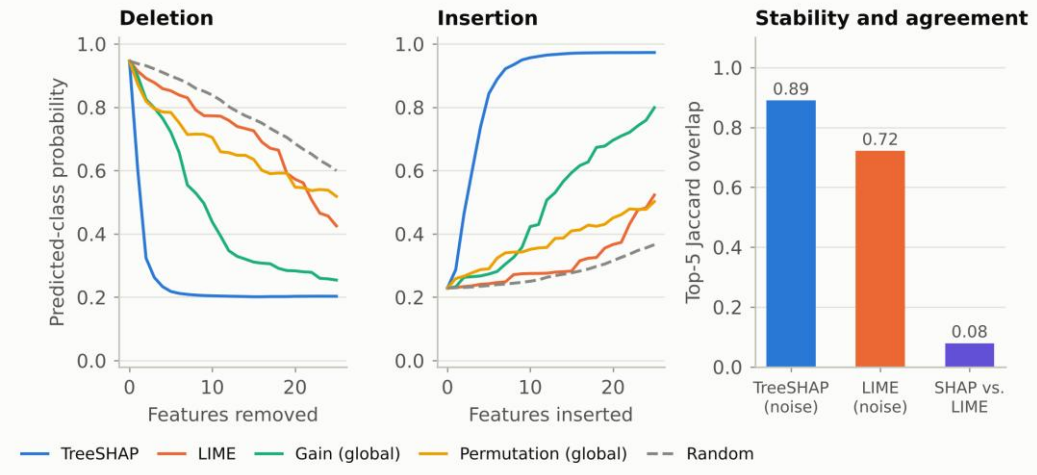


Fig. 22. Faithfulness, stability and agreement of the explanations of XGBoost. Left: mean predicted-class probability when the features are removed in the order of the ranking (first 25 features). Centre: the same when features are inserted into the commissioning-median baseline. Right: top-5 Jaccard overlap between the explanations of a window and of a noisy copy (TreeSHAP and LIME) and between TreeSHAP and LIME on the same window.

Are they stable and consistent? Under input noise of 5 % of the commissioning standard deviation, which does not change the predicted class, the TreeSHAP attribution vector changes by 20 % in relative ℓ_2 norm and its top five features keep a Jaccard overlap of 0.89; for LIME the overlap is 0.72 (Table 32), because each LIME explanation is a fresh random sample. The agreement between methods is poor. For the same window, the top-5 sets of TreeSHAP and LIME have a Jaccard overlap of only 0.08 and the rank correlation of the absolute attributions is $\tau = 0.17 \pm 0.12$, and the agreement is lowest

for Amplification and Recon (Jaccard 0.003 and 0.05), the classes in which several correlated traffic features carry similar information, which we expect to split the attribution differently in the two methods. The global rankings agree more where the definition is the same: TreeSHAP of XGBoost and of the random forest have a Spearman correlation of 0.82 and TreeSHAP and gain importance 0.88, but TreeSHAP and permutation importance only 0.38 and TreeSHAP and LIME 0.09. Integrated Gradients of the two networks and TreeSHAP of XGBoost, computed for the same windows, are only weakly correlated (Spearman 0.38 for the MLP and 0.34 for the CNN–LSTM). Even two definitions of the Shapley value disagree on a model with correlated inputs: interventional TreeSHAP and KernelSHAP, which estimate the same quantity with the same background sample, have a mean absolute difference of 8.7 % of the attribution mass and a rank correlation of 0.65, and path-dependent and interventional TreeSHAP of the same model correlate by 0.72 (Table 32). The practical conclusion is that an explanation is not a property of a prediction but of a pair of a method and a model, and that a deployed system should fix one method, state its value function, and report its faithfulness and stability together with the explanation, as we do here.

TABLE XXXII. STABILITY AND CROSS-METHOD AGREEMENT OF THE EXPLANATIONS. STABILITY COMPARES THE EXPLANATION OF A WINDOW WITH THAT OF A COPY PERTURBED BY GAUSSIAN NOISE OF 5 % OF THE COMMISSIONING STANDARD DEVIATION, OVER PERTURBATIONS THAT DO NOT CHANGE THE PREDICTED CLASS.

Quantity	Value	Based on
TreeSHAP: top-5 Jaccard under 5 % noise	0.89	1,246 perturbed windows
TreeSHAP: relative ℓ_2 change of attribution	0.20	same
LIME: top-5 Jaccard under 5 % noise	0.72	same
TreeSHAP vs. LIME: Kendall τ (per window)	0.17 ± 0.12	1,050 windows
TreeSHAP vs. LIME: top-5 Jaccard (per window)	0.08	1,050 windows
Global mean absolute SHAP (XGBoost) vs. permutation importance: Spearman ρ	0.38	68 features
Global mean absolute SHAP (XGBoost) vs. gain importance: Spearman ρ	0.88	68 features
Global mean absolute SHAP (XGBoost) vs. LIME: Spearman ρ	0.09	68 features
Global mean absolute SHAP (XGBoost) vs. mean absolute SHAP (random forest): Spearman ρ	0.82	68 features
KernelSHAP vs. interventional TreeSHAP: Spearman ρ (per window)	0.65	70 windows
Path-dependent vs. interventional TreeSHAP: Spearman ρ (per window)	0.72	70 windows
Integrated Gradients (MLP) vs. TreeSHAP: Spearman ρ (per window)	0.38	350 windows
Integrated Gradients (CNN-LSTM) vs. TreeSHAP: Spearman ρ (per window)	0.34	350 windows

7.11 Latency and cost

Table 33 and Fig. 23 give the measured cost per 30 s window on two virtual CPU cores of a cloud server (no GPU). The best detector, the lagged XGBoost model, needs a median of 0.42 ms and a 95th percentile of 0.61 ms per window including the assembly of its 340-dimensional input, and it sustains about 109 000 windows per second in batches. Plain XGBoost, LightGBM, the MLP, the CNN–LSTM and logistic regression are all below 1 ms at the 95th percentile (0.49, 0.67, 0.17, 0.91 and 0.16 ms); the random forest is the exception with 110 ms because every tree of the 400-tree ensemble is evaluated in turn by a single-window call. The feature extraction costs 0.09 ms per window, 0.06 ms for the network features and 0.03 ms for the energy features, when a whole day is processed in one pass. The explanation is the dominant cost of the pipeline: TreeSHAP for one XGBoost prediction takes 1.1 ms (95th percentile 1.2 ms), against 25 ms for our LIME implementation, 0.35 s for KernelSHAP and 83 ms for TreeSHAP of the random forest; Integrated Gradients takes 0.4 ms (MLP) and 3.1 ms (CNN–LSTM) per window in batches. The complete chain of feature extraction, prediction and TreeSHAP explanation of XGBoost has a 95th percentile of 1.8 ms (the sum of the three 95th percentiles), that is 0.006 % of the 30 s window; even on a gateway that is a hundred times slower, the budget would be below 1 % of the window. This figure excludes the parsing of the 0.52 million messages that arrive per day (about six per second), and it is a single-process measurement on a server CPU, so it is a lower bound for edge hardware; but it settles the question of principle that a tree-based detector with an exact explanation is far from the computational limit of the monitoring task, and that the reasons for preferring or avoiding a model in this setting should be accuracy, robustness and the quality of its explanation rather than speed.

TABLE XXXIII. LATENCY PER 30 S WINDOW ON 2 VIRTUAL CPU CORES, SINGLE PROCESS. INFERENCE AND EXPLANATION LATENCIES ARE FOR ONE WINDOW AT A TIME (600 RANDOMLY CHOSEN TEST WINDOWS; 300 FOR TREESHAP OF XGBOOST AND 100 FOR THE RANDOM FOREST); THROUGHPUT IS FOR A BATCH OF 5 000 WINDOWS. FEATURE EXTRACTION IS THE MEAN COST PER WINDOW OF PROCESSING ONE FULL DAY.

Component	p50 (ms)	p95 (ms)	p99 (ms)	Batch throughput (windows/s)
Logistic regression	0.11	0.16	0.19	580,371
Random forest	87.53	109.80	120.51	25,435
XGBoost	0.33	0.49	1.12	256,258
LightGBM	0.53	0.67	0.97	109,582
MLP	0.09	0.17	0.22	596,871
CNN-LSTM	0.64	0.91	1.22	59,086
XGBoost (5-window lag)	0.42	0.61	0.88	108,629
TreeSHAP, XGBoost (explanation)	1.09	1.25	1.72	—
TreeSHAP, random forest (explanation)	83.07	102.49	112.43	—
Feature extraction, network (per window, batch-amortised)	0.060	—	—	—
Feature extraction, energy (per window, batch-amortised)	0.033	—	—	—

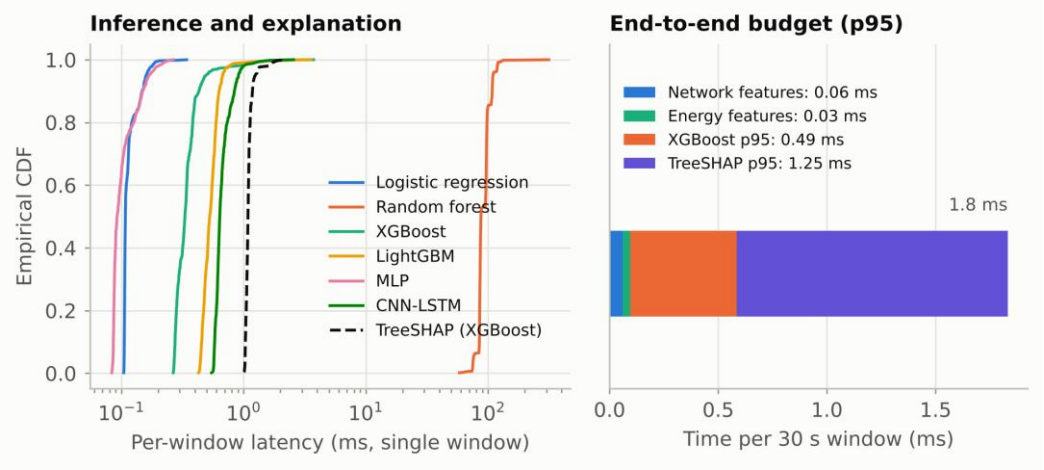


Fig. 23. Left: empirical distribution of the latency of one window for each model and for TreeSHAP of XGBoost (logarithmic axis). Right: the 95th-percentile budget of the pipeline of feature extraction, XGBoost prediction and TreeSHAP explanation.

8. DISCUSSION

8.1 Principal findings

Detection. Under a chronological protocol with a purge gap, the best of seven classifiers, an XGBoost model that sees the five latest windows, reaches a macro-F1 of 0.913 (95 % interval 0.893–0.927) and an MCC of 0.907 on seven unseen days. Five of the six attack classes are detected with an F1 between 0.93 and 0.98 and every burst of those five classes is found, usually in the first window. Gradient-boosted trees are at least as good as a CNN–LSTM and an MLP on the 68 engineered features and are cheaper to train (6 s for XGBoost against 48 s for the CNN–LSTM); the advantage of the lagged model over its single-window counterpart is small (0.7 points) but statistically reliable.

Evaluation. A conventional random split of the same windows would have inflated the macro-F1 by 1.2 to 2.7 points and the minority-class FDIA F1 by 10 to 13 points, because the median test window then has a training neighbour 30 s away. The inflation is moderate in our setting, where bursts are short, and it would be larger for data sets that consist of a few long captures. The practical message is that a protocol should be judged by the temporal distance between training and test samples as well as by the split ratio.

The limit of the stream. The false-data injection class is the hardest one because of its structure, not because of the learner. The unbalanced variants are detected through the signed balance residual in 81 % (bias) and 65 % (ramp) of their windows;

the balanced variant, which satisfies the same consistency relation as the true measurement, is flagged in at most 10 % of its windows by the models we examined (0 % by the best one), and removing the energy-balance features from the input removes the FDIA class altogether (F1 of 0 with network features only). Between these two extremes, the energy-consistency features add 9 points of macro-F1 to the network features, and the temporal-integrity features (round-trip time, TTL, duplicates, sequence gaps, clock skew) are the only network group whose loss costs more than a tenth of a point.

Robustness. Detection quality degrades gracefully but quickly when an attacker weakens the attack: half of the training intensity costs about 5 points of macro-F1 and a quarter costs 15. Replay and FDIA are the most sensitive classes. A model trained on one site transfers to another with a 12-point loss, and the loss is explained by a single mechanism, the confusion of route changes with man-in-the-middle delay, which a few days of local data repair.

TABLE XXXIV. SUMMARY OF THE STRESS TESTS. REFERENCE IS THE CHRONOLOGICAL, IN-DISTRIBUTION RESULT OF THE SAME MODEL; THE LAST COLUMN IS THE CHANGE CAUSED BY THE STRESS (THE RANDOM SPLIT IS OPTIMISTIC, THE OTHER STRESSES ARE PESSIMISTIC).

Stress	Condition	Metric	Reference	Stressed	Change
Evaluation protocol	XGBoost: random vs. chronological split	Macro-F1	0.907	0.933	+2.7 pp
Evaluation protocol	Random forest: random vs. chronological split	Macro-F1	0.896	0.908	+1.2 pp
Unseen FDIA variant	XGBoost: V1-bias held out	Recall of the variant	81.0 %	66.7 %	-14.3 pp
Unseen FDIA variant	XGBoost: V2-ramp held out	Recall of the variant	64.6 %	61.1 %	-3.6 pp
Attack intensity	XGBoost: scale 0.5	Macro-F1	0.905	0.857	-4.9 pp
Attack intensity	XGBoost: scale 0.25	Macro-F1	0.905	0.751	-15.5 pp
Attack intensity	XGBoost: scale 0.1	Macro-F1	0.905	0.600	-30.6 pp
Cross-site	XGBoost: site B, no adaptation	Macro-F1	0.907	0.785	-12.2 pp
Cross-site	XGBoost: site B, 8 adaptation days	Macro-F1	0.907	0.900	-0.6 pp

Explanations. TreeSHAP attributions are exact, faithful (the five top-ranked features account for most of the probability of the predicted class), reasonably stable under noise (top-5 overlap 0.89) and consistent with the generating mechanisms, so they pass the checks that a domain expert would apply. LIME and the global importance rankings do not: they are less faithful, less stable, and different from each other. The agreement between methods and between model families is low, which means that the reported attribution depends on the choice of method and cannot be read as a property of the data.

TABLE XXXV. COMPARISON OF THE EXPLANATION METHODS ON THE STRATIFIED SAMPLE: DELETION AUC (LOWER IS MORE FAITHFUL), COMPREHENSIVENESS AT FIVE FEATURES (HIGHER IS MORE FAITHFUL), TOP-5 STABILITY UNDER INPUT NOISE, AND COST PER EXPLANATION. THE FAITHFULNESS COLUMNS REFER TO XGBOOST EXCEPT FOR INTEGRATED GRADIENTS, WHICH EXPLAINS THE NETWORK NAMED IN THE FIRST COLUMN.

Method	Applies to	Deletion AUC ↓	Comp.@5 ↑	Stability (Jaccard) ↑	Cost per window (ms)	Remark
TreeSHAP	Tree ensembles (exact)	0.22	0.73	0.89	0.45	Exact; additive to 4×10^{-6}
KernelSHAP	Any model	—	—	—	345	$\rho = 0.65$ with interventional TreeSHAP
LIME (own)	Any model	0.44	0.09	0.72	24.8	Surrogate $R^2 = 0.42$
Integrated Gradients (MLP)	Differentiable	0.19	0.70	—	0.44	$\rho = 0.38$ with TreeSHAP
Integrated Gradients (CNN–LSTM)	Differentiable	0.12	0.73	—	3.10	$\rho = 0.34$ with TreeSHAP
Gain importance	Trees, global	0.32	0.23	—	≈ 0	Class-agnostic
Permutation importance	Any model, global	0.41	0.16	—	—	Class-agnostic; needs many re-predictions
Random ranking	—	0.51	0.05	—	—	Null reference

Cost. A full pass of feature extraction, prediction and exact explanation takes less than 2 ms per window on a server CPU, so computational cost is not the argument for or against a particular detector.

8.2 Positioning against prior work

The data-set and benchmark studies reviewed in Section 2 establish what traffic-based detectors can achieve on captured attacks, and our results on the five network-visible classes (F1 0.93–0.98) are of the same order as the high scores they report. The comparison is nevertheless only qualitative, because those scores come from other traffic, other labels and, in most cases, random splits; Table 36 therefore compares practice rather than numbers. Three differences matter for the interpretation of our results. First, the evaluation protocol: a chronological split with a purge gap and interval estimates gives a lower and more honest estimate than the 99 % accuracies that are typical of random splits, and it makes the difference between the two measurable. Second, the physical process: only when the data contain a measurement process with a consistency relation can the false-data injection that an operator cares most about be separated from benign faults, and only then does the balance-preserving variant appear as the limit of what any stream-based detector can do. Third, the explanation: reporting an attribution without a faithfulness and stability check, as is usual, would have hidden the fact that three of the four methods we examined disagree on which features matter.

TABLE XXXVI. PRACTICE IN THE DATA-SET AND BENCHMARK PAPERS DISCUSSED ABOVE COMPARED WITH THE PROTOCOL OF THIS STUDY.

Aspect	Practice in the papers reviewed	This study
Data split	Random or stratified split of windows or flows (e.g. 80/20 in [19])	Chronological split with a 24-window purge gap; random and burst-grouped splits kept as contrast
Optimism	Rarely measured	Random split adds 2.7 pp (XGBoost) and 1.2 pp (RF) macro-F1 and up to 13 pp FDIA F1
Uncertainty	Single point estimate	Block bootstrap, Friedman–Nemenyi test with Holm correction, five seeds
Protocol confounding	Attack classes often generated on one transport	Attacks on both transports where meaningful; recall reported per transport
Benign background	Idle or scripted normal traffic	Five benign look-alike classes; sensor faults alone cause 15.7 % false alarms
Physical process	None	Exact power balance; three FDIA variants including a balance-preserving one
Explanations	Qualitative plots	Deletion/insertion, stability and agreement of four methods; latency
Robustness	Not tested	Unseen variants, attack-intensity scaling, cross-site transfer with few-shot adaptation

8.3 Implications for deployment

Three observations carry over to a field deployment, independently of the exact figures. First, the control centre should not treat the detector as a single alarm. The network attacks are detected with a recall above 89 % and a delay of at most a few windows, and they can be acted upon automatically (rate limiting, source blocking, session reset), whereas the FDIA class is detected only when the injection unbalances the energy residual, so an FDIA alarm should open an investigation of the affected channels rather than trigger a control action. Second, the false-alarm budget is dominated by one benign mechanism. A sensor fault produces exactly the signature of a biased injection (a persistent residual on one channel), and no feature in this study separates the two; the operational remedy is a fault-and-tamper triage that combines the detector with device health data (self-diagnostics, signed firmware state, redundant sensors) rather than a better threshold. Third, the balanced FDIA is out of reach of any detector that uses only the stream observed by the control centre, because by construction every consistency relation that the centre can evaluate is satisfied. Detection of this variant requires an independent measurement (a calibrated reference meter on the feeder, a secure out-of-band channel, or message authentication), and it is a measure of the limits of the data rather than of the models that no amount of tuning on this data would remove. The consequence for the usual claims of near-perfect FDIA detection is straightforward: such figures are obtained when the attack is either unbalanced or tested on the training distribution, and they do not apply to an adversary who knows the plant.

8.4 Limitations and threats to validity

Simulation. All data are synthetic. The plant, the traffic and the attacks are modelled at message level; we do not simulate packet-level effects such as TCP congestion control, TLS record structure, queueing in the gateway or broker behaviour under overload, and the parameters of the attacks (intensity, duration, rate) were chosen by us. The absolute values reported here, for example a macro-F1 of 0.91, are therefore properties of this simulator, and they should not be read as field

performance. What we expect to transfer are the relative findings: the ordering of the model families, the size of the leakage effect, the dependence of the FDIA result on the injection variant, the contribution of the temporal-integrity and energy-consistency features, and the behaviour of the explanation methods. We have not validated the simulator against captured traffic from a physical site.

Labels and windows. A window is labelled by the burst that overlaps it by at least half, so windows at burst boundaries are ambiguous and some of the errors at the onset and the end of a burst are label noise rather than detector errors. The detection-delay figures are given in units of 30 s windows and cannot resolve sub-window latency.

Test size. The test block spans seven days and between 389 and 524 windows per attack class, and windows within a burst are strongly dependent. Confidence intervals and tests therefore use a block bootstrap, but the effective number of independent test events is the number of bursts (33–46 per class), which limits the precision of per-class estimates and of the burst-level detection rates.

Selection. Hyper-parameters were chosen on the validation block with a small random search (6–12 trials), and the 'best' model is the one with the highest test macro-F1 among seven candidates, so its score carries a small selection effect. Its margin over the runner-up (0.6 points) is smaller than the width of either model's own bootstrap interval, although the paired test, which cancels the variability shared by the two models, still finds it significant.

Explanations. Attribution methods were evaluated on a stratified sample of test windows with feature-removal tests that replace values by a nominal operating point. Removing a feature in this way produces off-manifold inputs, especially for strongly correlated features such as the three PV channels, and deletion and insertion scores are therefore comparative indicators of faithfulness and not absolute measures of it. The faithfulness of an attribution to the model also says nothing about whether the model has learned the physically right relation; the case studies are consistent with the simulated mechanisms, which is encouraging but is not a proof.

Generalisation across sites. Transfer is evaluated on one additional simulated site, which differs in climate, ratings, link quality and benign-event rates, but which shares the simulator's structure and the attack implementation. Two sites are not enough to estimate how far a detector would transfer to a real, structurally different plant.

9. CONCLUSION

We built a reproducible message-level co-simulation of a hybrid PV–wind–diesel–battery microgrid with MQTT and CoAP telemetry, injected six attack families and five classes of benign hard negatives, and used it to evaluate seven classifiers and four explanation methods under a protocol that controls temporal leakage. The study supports four conclusions. First, with a chronological split and uncertainty estimated by a block bootstrap, a boosted-tree detector with 68 network and energy-consistency features reaches a macro-F1 of 0.913 on unseen days, which is lower than the figures usually reported but is not inflated by near-duplicate windows; a random split would have added up to 2.7 points overall and 13 points on the FDIA class. Second, the network attacks are detected quickly and reliably, whereas FDIA detection depends on the injection: about 81 % of biased windows and 65 % of ramp windows are found and at most 10 % of the windows of a balance-preserving injection (none for the best model), a limit that no tuning of the models on this data can remove. Third, accuracy is fragile outside the training distribution: halving the attack intensity costs 5 points of macro-F1 and quartering it 15 points, and moving to another site costs 12 points (and 69 points on the MITM class, whose delay signature is confused with route changes), while eight days of local data recover almost all of the loss. Fourth, explanations must be measured, not displayed: TreeSHAP is faithful, stable and mechanism-consistent, LIME and global rankings are not, and the methods disagree with one another, while the computation of an exact explanation costs about a millisecond.

Several directions follow. The balanced FDIA calls for an independent source of truth, such as a calibrated reference meter, a secure out-of-band channel or message authentication, and its detection should be studied as a sensing problem rather than a learning problem. The loss of transfer suggests normalising the temporal-integrity features by a locally estimated baseline and testing domain-adaptation methods on more than two sites. A fault-and-tamper triage that combines the detector with device health data should reduce the false alarms that sensor faults cause. Finally, the conclusions on relative performance, leakage and explanation quality should be validated on traffic and measurements captured from a physical site, which we could not do here, and the simulator, the code and all result files are provided to allow others to repeat and extend our experiments.

Acknowledgment

Not applicable.

Funding

This work was supported by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia (Grant No. KFU266341).

CRediT Authorship Contribution Statement

Ahmed Fakhri Abdullah contributed to conceptualization, methodology, validation, formal analysis, writing – original draft, writing – review & editing, and project administration; Omar Ahmed Razooqi contributed to conceptualization, software, investigation, and writing – original draft; Oras Fadhil Khalaf contributed to methodology, formal analysis, investigation, and data curation; Ayman AbuBaker contributed to software, data curation, and visualization; Amer Alqutaish contributed to validation, investigation, and writing – review & editing; Saidah Khan contributed to supervision, validation, and writing – review & editing; and all authors have read and approved the final manuscript..

Conflicts of Interest

The authors declare no conflicts of interest.

Ethical Approval

This study did not involve human or animal subjects, and ethical approval was not required.

Data Availability Statement

The data supporting the findings of this study are available from the corresponding author upon reasonable request.

References

[1] I. Vaccari, G. Chiola, M. Aiello, M. Mongelli, and E. Cambiaso, “MQTTset, a New Dataset for Machine Learning Techniques on MQTT,” Sensors, vol. 20, no. 22, p. 6578, Nov. 2020, doi: 10.3390/s20226578.

[2] Z. Shelby, K. Hartke, and C. Bormann, “The Constrained Application Protocol (CoAP),” IETF RFC 7252, Jun. 2014.

[3] H. Zeghida, M. Boulaiche, R. Chikh, A. Patel, A. L. B. Barros, and A. M. Bamhdi, “XMID-MQTT: Explaining machine learning-based intrusion detection system for MQTT protocol in IoT environment,” International Journal of Information Security, vol. 24, Art. no. 128, 2025, doi: 10.1007/s10207-025-01036-w.

[4] Almaiah, Mohammed Almaayah, and Mohamed Chahine Ghanem. "Deep Learning-Based Adaptive Intrusion Detection for Next-Generation Network Security." Journal of Information and Cybersecurity 2026, no. 1 (2026): 16-31.

[5] A. Al Hanif and M. Ilyas, “Effective feature engineering framework for securing MQTT protocol in IoT environments,” Sensors, vol. 24, no. 6, Art. no. 1782, 2024, doi: 10.3390/s24061782.

[6] F. Pendlebury, F. Pierazzi, R. Jordaney, J. Kinder, and L. Cavallaro, “TESSERACT: Eliminating experimental bias in malware classification across space and time,” in *Proc. 28th USENIX Security Symp.*, Santa Clara, CA, USA, 2019, pp. 729–746.

[7] I. E. Kumar, S. Venkatasubramanian, C. Scheidegger, and S. A. Friedler, “Problems with Shapley-value-based explanations as feature importance measures,” in *Proc. 37th Int. Conf. Machine Learning (ICML)*, 2020, pp. 5491–5500.

[8] D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju, “Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods,” in *Proc. AAAI/ACM Conf. AI, Ethics, and Society (AIES)*, New York, NY, USA, 2020, pp. 180–186.

[9] D. Alvarez-Melis and T. S. Jaakkola, “On the robustness of interpretability methods,” arXiv preprint arXiv:1806.08049, 2018. [Online]. Available: https://arxiv.org/abs/1806.08049.

[10] I. Vaccari, M. Aiello, and E. Cambiaso, “SlowITe, a novel denial of service attack affecting MQTT,” Sensors, vol. 20, no. 10, Art. no. 2932, 2020, doi: 10.3390/s20102932.

[11] S. M. Lundberg *et al.*, "From local explanations to global understanding with explainable AI for trees," *Nature Machine Intelligence*, vol. 2, no. 1, pp. 56–67, Jan. 2020.

[12] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017, pp. 4765–4774.

[13] D. Garreau and U. von Luxburg, "Explaining the explainer: A first theoretical analysis of LIME," in Proc. 23rd International Conference on Artificial Intelligence and Statistics, vol. 108, 2020, pp. 1287–1296.

[14] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proc. 34th Int. Conf. Machine Learning (ICML)*, Sydney, Australia, 2017, pp. 3319–3328.

[15] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Information Systems Security and Privacy (ICISSP)*, 2018, pp. 108–116.

[16] Alshabibi, Moza. "A systematic review of cyber security threats in Cloud Computing." International Journal of Cybersecurity Engineering and Innovation 2025, no. 1 (2025).

[17] H. Hindy, E. Bayne, M. Bures, R. Atkinson, C. Tachtatzis, and X. Bellekens, "Machine learning based IoT intrusion detection system: An MQTT case study (MQTT-IoT-IDS2020 dataset)," arXiv:2006.15340, 2020.

[18] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning," *IEEE Access*, vol. 10, pp. 40281–40306, 2022.

[19] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment," *Sensors*, vol. 23, no. 13, art. 5941, Jun. 2023.

[20] A. Sayghe, Y. Hu, I. Zografopoulos, X. Liu, R. G. Dutta, Y. Jin, and C. Konstantinou, "Survey of machine learning methods for detecting false data injection attacks in power systems," *IET Smart Grid*, vol. 3, no. 5, pp. 581–595, Oct. 2020.

[21] C. Rossow, "Amplification hell: Revisiting network protocols for DDoS abuse," in *Proc. Network and Distributed System Security Symp. (NDSS)*, San Diego, CA, USA, 2014.

[22] S. Neupane, J. Ables, W. Anderson, S. Mittal, S. Rahimi, I. Banicescu, and M. Seale, "Explainable intrusion detection systems (X-IDS): A survey of current methods, challenges, and opportunities," arXiv:2207.06236, 2022.

[23] T.-T.-H. Le, H. Kim, H. Kang, and H. Kim, "Classification and explanation for intrusion detection system based on ensemble trees and SHAP method," Sensors, vol. 22, no. 3, Art. no. 1154, 2022, doi: 10.3390/s22031154.

[24] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller, "Evaluating the visualization of what a deep neural network has learned," *IEEE Trans. Neural Networks and Learning Systems*, vol. 28, no. 11, pp. 2660–2673, Nov. 2017.

[25] S. M. Lundberg et al., "From local explanations to global understanding with explainable AI for trees," Nature Machine Intelligence, vol. 2, pp. 56–67, 2020, doi: 10.1038/s42256-019-0138-9.

[26] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, no. 5, pp. 206–215, May 2019.

[27] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," arXiv:1702.08608, 2017.

[28] P. Ineichen and R. Perez, "A new airmass independent formulation for the Linke turbidity coefficient," Solar Energy, vol. 73, no. 3, pp. 151–157, 2002, doi: 10.1016/S0038-092X(02)00045-2.

[29] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, art. 6, Jan. 2020.

[30] F. Charmet et al., "Explainable artificial intelligence for cybersecurity: A literature survey," Annals of Telecommunications, vol. 77, nos. 11–12, pp. 789–812, 2022, doi: 10.1007/s12243-022-00926-7.

[31] G. Biau and E. Scornet, "A random forest guided tour," TEST, vol. 25, no. 2, pp. 197–227, 2016, doi: 10.1007/s11749-016-0481-7.

[32] L. Cui, Y. Qu, L. Gao, G. Xie, and S. Yu, "Detecting false data attacks using machine learning techniques in smart grid: A survey," Journal of Network and Computer Applications, vol. 170, Art. no. 102808, 2020, doi: 10.1016/j.jnca.2020.102808.

[33] G. Ke *et al.*, "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017, pp. 3146–3154.

[34] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[35] S. Aoufi, A. Derhab, and M. Guerroumi, “Survey of false data injection in smart power grid: Attacks, countermeasures and challenges,” Journal of Information Security and Applications, vol. 54, Art. no. 102518, 2020, doi: 10.1016/j.jisa.2020.102518.

[36] B. Bischl et al., “Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges,” WIREs Data Mining and Knowledge Discovery, vol. 13, no. 2, Art. no. e1484, 2023, doi: 10.1002/widm.1484.

[37] T. D. Caleb, S. Shao, and N. Kaabouch, “False data injection attacks on smart grids: Attack models, challenges and future directions,” International Journal of Information Security, vol. 25, Art. no. 94, 2026, doi: 10.1007/s10207-026-01262-w.

[38] P. Vigneshwaran, S. Thuseethan, B. Shanmugam, and S. Thennadil, “Cyber attack detection in smart grids: A survey of methods, challenges and future directions,” Computer Science Review, vol. 60, Art. no. 100915, 2026, doi: 10.1016/j.cosrev.2026.100915.

[39] S. García and F. Herrera, “An extension on ‘Statistical comparisons of classifiers over multiple data sets’ for all pairwise comparisons,” Journal of Machine Learning Research, vol. 9, pp. 2677–2694, 2008.

[40] S. Holm, “A simple sequentially rejective multiple test procedure,” *Scandinavian J. Statistics*, vol. 6, no. 2, pp. 65–70, 1979.

[41] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” *J. Artificial Intelligence Research*, vol. 16, pp. 321–357, Jun. 2002.

[42] H. Ishfaq, N. Mushtaq, M. A. Saleem, Q. Huang, and W. Amin, “A comprehensive survey of false data injection attacks modeling and ML/DL-based detection in smart grids,” ICT Express, early access, Aug. 8, 2026, doi: 10.1016/j.icte.2026.07.027.

[43] A. Paszke *et al.*, “PyTorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019.