International Journal of Cybersecurity Engineering and Innovation

ARTICLE INFO

Article History: Received: 25-03-2026, Revised: 01-06-2026, Accepted: 03-11-2026
Corresponding author Email: rejwan.sulaiman@northumbria.ac.uk

This is an open access article under the CC BY 4.0 license (http://creativecommons.org/licenses/by/4.0/).

Published by ITAP Publisher.



Vol. 2027 No.1

An Integrated Explainable AI Framework for Real-Time Intrusion Detection and Dynamic Cybersecurity Risk Assessment

Rejwan Bin Sulaiman[1*], Mohammed Amin Almaiah

[1]School of Computer Science and Technology, Northumbria University, Newcastle Upon Tyne, UK, rejwan.sulaiman@northumbria.ac.uk, https://orcid.org/00000002-3037-7808

[2] King Abdullah the II IT School, The University of Jordan, Amman 11942, Jordan, m.almaiah@ju.edu.jo, https://orcid.org/0000-0002-2215-2481

Abstract

This study proposes XRisk-IDS, an explainable artificial intelligence framework for real-time intrusion detection and cybersecurity risk assessment in enterprise networks. The proposed framework integrates LightGBM and a convolutional neural network–bidirectional gated recurrent unit (CNN–BiGRU) temporal branch through an adaptive fusion mechanism. The decision-making process is enhanced by incorporating probability calibration, predictive uncertainty, explanation fidelity, explanation stability, asset criticality, and network exposure. The framework was evaluated using the official UNSW-NB15 test partition, comprising 82,332 network flows distributed across 10 classes. Experimental results demonstrated that XRisk-IDS achieved an accuracy of 71.68%, a balanced accuracy of 60.18%, a macro-F1 score of 48.03%, and a macro-area under the receiver operating characteristic curve (macro-AUROC) of 95.63%. Compared with the standalone CNN–BiGRU model, the proposed framework improved the macro-F1 score by 18.59 percentage points, although Random Forest demonstrated superior overall predictive performance. Furthermore, XRisk-IDS achieved a mean explanation quality score of 0.7172 and high-impact attack coverage of 99.45%. The framework demonstrated efficient computational performance, processing approximately 26,274 network flows per second with a compact model footprint of 2.74 MB. These findings highlight the potential of XRisk-IDS to establish an auditable connection between intrusion detection outcomes, explanation reliability, and operational cybersecurity risk assessment. However, further research is required to reduce false positives involving benign traffic, improve multiclass probability calibration, and validate the framework's effectiveness in real-world enterprise network environments.

Keywords: Explainable artificial intelligence, Intrusion detection, Cybersecurity risk assessment, Enterprise networks, and Real-time threat detection.

1. Introduction

Enterprise networks are now very distributed computing environments with traditional data centres sharing the space with cloud platforms, remote endpoints, Internet of Things devices, software-defined infrastructure and third-party services.

This is a design that is integrated, but also offers flexibility in operation and increases the attack surface and the number of ways to attack it. Bad guys aren't just using the signatures of malware that are well known. They are able to stay hidden in the normal network traffic by using encrypted command channels, identity misuse, lateral movement, low rate reconnaissance, fileless execution and legitimate administrative tools [1]. This has led to a change in enterprise security from perimeter protection to continual, context-driven risk assessment.

IDSs are still a key element of this transformation. Signature-based detectors are very effective against known attacks, but are ineffective against attacks that have not been previously encoded as a rule [2]. Anomaly-based approaches have the advantage of being able to learn deviations from the expected behaviour, but may suffer if the legitimate network patterns evolve over time [3]. Enterprise traffic is especially challenging to model due to its high-dimensional, imbalanced, temporally dependent and heterogeneous applications nature [4]. The critical attack events are rare, making training even more complex as a classifier could be very accurate in overall classification, but fail to detect the intrusion classes that pose the highest risk for operations [5].

Machine learning has enhanced the detection of intricate relationships between packet statistics, flow attributes, user actions, authentication events and host-level attributes [6]. Network intrusion detection has therefore been tackled with the use of ensemble learning, deep neural networks, recurrent architectures and attention-based models with promising prediction results [7]. However, the accuracy of predictions is not enough for successful operations in cyber security. Many of the high performing models are opaque decision systems, that is, they return an attack label or a probability without telling which observations were used to make the decision [8]. This makes it difficult for analysts to have confidence, validate incidents, and investigate false alarms.

Explainable AI (XAI) can be a potential way to bridge this divide between predictive ability and human understanding. The methods of feature-attribution can be used to determine which traffic variables are most likely to have a significant influence on a particular prediction, and local surrogate models can be used to model the behaviour of a complex detector in the vicinity of a particular observation [9]. Global explanation techniques offer more general information about the attack patterns learned and important features in a whole dataset [10]. These mechanisms can be useful to analysts to differentiate between a true intrusion and a normal operation. They can also show if a detector is using meaningful indicators (such as unusual packet rates, failed authentications) or on unstable correlations that are not of much security value [11].

However, explainability is often considered a post-processing visualization and not as a part of intrusion detection. The first step is to train the model for classification, and then attach an explanation tool to some of the outputs [12]. This is a limited support for real time risk assessment, as the explanation is not directly linked to the level of threat, the criticality of the asset, the level of confidence in the prediction, or the urgency of the response [13]. Furthermore, the explanation methods might lead to different rankings of features if there are correlated network attributes [14]. Generating local explanations can also require a significant amount of computation, which can also be in conflict with the low latency requirements of enterprise monitoring [15]. No matter how well interpreted, the explanation comes after the intrusion has propagated, and is of little practical value.

Another constraint is the limited evaluation which is generally used in the current studies. The accuracy, precision, recall and F1-score are the most common metrics used to evaluate many intrusion detectors using balanced benchmark subsets [16]. These metrics are required, but are not sufficient to capture operational scenarios with concept drift, unseen attacks, extreme class imbalance or varying traffic volume [17]. Likewise, there is a lack of quantitative assessment of the stability, fidelity, sparsity and agreement with known attack characteristics in model explanations [18]. The quality of the explanation generated and the usefulness of the explanation to the detection process is therefore not sufficiently explored.

To perform real-time cybersecurity risk assessment, more than just determining if a network flow is malicious is needed. The security teams need to decide on the likelihood of the intrusion, what enterprise asset is being impacted, the potential impact of the intrusion, and whether or not immediate action is warranted [19]. A low confidence alert from an authentication server that is critical to the system could be more significant than a high confidence anomaly from a single test device. Therefore, risk estimation should take into account contextual information as well as model evidence, and not model evidence alone [20]. This is crucial as probability is a measure of statistical certainty, while cybersecurity risk is the likely outcome of a threat in a specific operational context.

In response to these constraints, this study presents an explainable artificial intelligence (XAI) system to detect intrusions and evaluate cybersecurity risks in enterprise networks in real-time. The framework handles network-flow and behavioural

characteristics with an optimized detection model, calculates the level of intrusion confidence and converts the prediction to a context-aware risk score. Local and global explanations are then created to reveal the features that facilitate each of the decisions [21]. The explanation layer is intended to always be directly linked to attack class, risk severity and recommended response priority. Predictions with low confidence or those that don't explain the data are not considered as alerts but are flagged for analysts to review.

The novelty of the work proposed is in four related contributions. First, intrusion detection and enterprise risk estimation are integrated into a single decision pipeline, instead of being separate stages. Second, it integrates predictive confidence, threat probability, asset value, and potential harm to yield a comprehensible risk score [22]. Third, explanation fidelity and stability are assessed in addition to traditional detection metrics, and are thus considered as measurable system properties of the system. Finally, the framework is evaluated for different traffic loads, class imbalance and attack conditions that have not been seen before to test its applicability for real-time deployment [23].

2. Literature Review

2.1 Machine Learning-Based Intrusion Detection

The traditional IDSs are mostly signature based, rule based and threshold based systems that are configured manually. These techniques are still effective against attacks that have been seen in the past, but their effectiveness decreases if the adversary changes the payload, communication patterns or execution paths [24]. Anomaly-based methods overcome this by training a model of the statistical characteristics of normal traffic and detecting anomalies [25]. They are useful in enterprise networks because they are able to detect a wider range of things, but the changes that occur during normal operations can also be detected as malicious.

Machine learning has brought in a more flexible way of network threat detection. Network flow classification using packet level and session level attributes has been done by decision trees, support vector machines, logistic regression, random forests and boosting models [26]. Usually ensemble models are stable in performance as the influence of a single classifier's bias is mitigated by having multiple classifiers. They also make decisions at a feature level that can be easily reviewed, as opposed to very deep architectures. The quality of the predictions, however, is very sensitive to feature engineering, class distribution and representativeness of the training data.

Deep learning can help to decrease manual feature building. CNNs are able to recognize local patterns in traffic vectors while RNNs are able to capture the temporal dependency among communication sequences [28]. The hybrid CNN–LSTM models have been found useful in detecting attacks that occur over a series of related events [29] and possess both properties. Autoencoders are often applied to unsupervised anomaly detection, particularly if there are not many attack records available for labeling [30]. Recently, there has been an interest in learning long-range relationships between flows, users, endpoints and services using more recent techniques such as attention mechanisms and transformer-based models [31].

With all these improvements, high predictive results on benchmark sets do not necessarily mean reliable enterprise results. There are many public datasets that are either stale, duplicate instances, have only a few protocols, or have only a few attack conditions [32]. Random train–test splitting may also lead to records being in both sets that are very similar, which results in an optimistic estimate of generalization [33]. Traffic in operational networks evolves as a result of software changes, user behaviour, business cycles and infrastructure migrations. The sensitivity of a detector that is focused on a fixed distribution can then decrease with time.

2.2 Explainable AI in Cybersecurity

As the use of complex learning models is increasing, so is the need for explainability. Security analysts must be more than just given a malicious or benign classification; they must be able to determine the reason for the alert, and if the evidence is consistent with a known attack pattern [34]. Explainable AI (XAI) aims to reveal this reasoning by means of feature contributions, local approximations, counterfactual examples, rules and visual summaries.

Model-agnostic techniques like LIME provide an explanation of the selected observation around the chosen model [35]. SHAP decomposes the contribution of each feature based on cooperative game theory, and provides local and global interpretation [36]. The methods are popular as they do not require the redesign of the detection model and can be attached

to various classifiers. However, they can be different in the presence of highly correlated features in the network, sparse features, or features that are disturbed beyond realistic traffic ranges [37]. There's another way, intrinsic interpretability. Some learners, such as rule-based learners, shallow trees, generalized additive models, and attention-guided architectures can give explanations directly from their internal structure [38]. These models are easier to audit, but can be less complex than other models needed for advanced intrusion detection. This presents a practical dilemma between the predictability and transparency. In high risk environments both dimensions are not to be overlooked.

The explanation plots of existing studies are typically shown for a few of the correctly classified samples. The fidelity, stability, consistency and usefulness to the analyst of explanations receive much less attention [39]. A feature ranking can seem to make sense, but may not be a good approximation of the actual detector. Explanations may also be fragile, in that small variations in the network flow result in a significantly different explanation even if the class that is predicted is the same [40]. This behaviour will lead to a loss of trust and make incident investigation more difficult.
The attackers can modify non-essential features to modify the explanation without changing the output of the detector, or can modify both prediction and explanation [41]. Thus, the quality of the explanation needs to be evaluated in terms of the robustness of the system, and not as a presentation layer that is added after the classification.

2.3 Real-Time Cybersecurity Risk Assessment

While intrusion detection and cybersecurity risk assessment are similar, they are not the same. Detection is used to determine if the observed behaviour is malicious or not, and risk assessment is used to determine the expected operational impact of the behaviour [42]. The factors that could be part of a meaningful risk score include: Attack probability, model confidence, asset criticality, exposure, potential impact, and reliability of supporting evidence.
There are a number of security platforms that have fixed severity tables or vulnerability scores [43]. Such scores are useful for prioritization, but may not be indicative of the current state of the network. The same event may have different impacts on different hosts, depending on the privilege of the user, the services that are affected, and the phase of the attack [44]. For enterprise deployment, it is therefore essential to have context-aware assessment.

There are computational constraints with real-time operation. The collection, preprocessing, prediction, explanation and risk scoring of packets should take place prior to an attack moving to another system [45]. Post-hoc explanations can be complex, and can be generated separately for each flow, which can lead to higher latency. This cost can be lowered by sampling [46] or by caching explanations or by selectively interpreting high-risk alerts [46]. These strategies are efficient, but can be missing some early indicators or low confidence events that still need to be worked on by the analyst.

2.4 Research Gap

There are significant advances in machine learning for intrusion detection and in explaining the results of the machine learning afterward, as shown in the literature. Most studies assess the detection, explanation, and risk scoring separately, however. There is little research that has studied the behavior of these components in combination when traffic loads vary, attacks are unknown, there is class imbalance, and latency is tight. There is also a lack of quantification of the quality of explanations in conjunction with the detection performance.

Table 1. Gap-to-solution mapping of representative studies on explainable intrusion detection and enterprise cyber-risk assessment

S. No.	Author(s) / Year / Ref.	Focus Area	Method / Tools	Key Finding	Research Gap	Link to Current Study
1	Shtayat et al., 2023 [1]	Explainable IIoT intrusion detection	Ensemble deep learning and XAI	Improved detection with feature-level interpretation	Limited to IIoT; no asset-aware risk scoring	Extends XAI to enterprise risk assessment
2	Li et al., 2025 [12]	Incremental intrusion detection	VAE, autoencoder, incremental learning	Adapted to evolving data streams	Limited explanation and alert prioritization	Adds explainable, risk-ranked decisions

3	Naz et al., 2026 [17]	Adaptive enterprise cyber-risk prediction	Hybrid blockchain and reinforcement learning	Enabled auditable threat mitigation	High complexity; limited explanation analysis	Adds lightweight XAI and uncertainty handling
4	Arreche et al., 2024 [20]	XAI evaluation for NIDS	SHAP, LIME, and ML classifiers	Quantified explanation quality	No integrated real-time risk pipeline	Uses fidelity and stability measures
5	Hassan et al., 2025 [22]	Context-aware Zero-Trust security	SIEM, SOAR, UEBA, and ML	Improved contextual threat response	Limited flow-level explanations	Integrates contextual risk with XAI
6	Nasir et al., 2026 [23]	Edge-deployable explainable IDS	BiGRU, attention, GWO, and SHAP	Achieved low-latency explainable detection	No enterprise impact assessment	Adds asset criticality and severity
7	Ozdem, 2025 [26]	Real-time anomaly detection	Temporal graph networks and XAI	Captured dynamic traffic patterns	Limited multiclass risk evaluation	Supports temporal and risk-aware detection
8	Ghadermazi et al., 2025 [37]	Early packet-based intrusion detection	Sequential packet imaging and deep learning	Enabled early attack identification	No explanation or risk prioritization	Adds interpretable alert ranking
-	Research gap	Integrated enterprise IDS	Detection, XAI, and risk fusion	Existing studies address these tasks separately	Lack of a real-time framework combining stable explanations, uncertainty, asset criticality, and severity	The proposed framework provides unified, auditable, and risk-aware intrusion detection

There is still a need for an integrated system that can identify intrusions, assess the risk in a context-sensitive manner, and generate consistent explanations in a single real-time pipeline. This should link influential features to the severity of the attack, differentiate between uncertainty and operational risk and highlight unreliable explanations that can be reviewed by humans. This is the reason why the present study [47] was conducted. As summarized in Table 1, existing studies have separately addressed explainable intrusion detection, adaptive learning, contextual threat analysis, and real-time classification. However, a unified enterprise framework that jointly supports low-latency detection, stable explanations, uncertainty handling, asset-aware risk scoring, and alert prioritization remains insufficiently explored.

3. Methodology

3.1 Experimental Design

The proposed framework, termed XRisk-IDS, was evaluated as a flow-based, real-time intrusion detection and cyber-risk assessment pipeline. The experiment was designed around four linked tasks: multiclass attack identification, probability calibration, explanation generation, and context-aware risk prioritization. Detection and explanation were not evaluated as isolated modules. Each network flow passed through the complete pipeline, including the latency introduced by preprocessing, model inference, explanation, and risk scoring.

As shown in Figure 1, XRisk-IDS contains a fast tabular branch and a temporal branch. Their calibrated outputs are combined through a learned gating function. The resulting prediction is then examined by the XAI layer before a risk score is assigned. Alerts with weak confidence or unstable explanations are forwarded for analyst review rather than being treated as reliable automated decisions. This design follows the operational need for both detection accuracy and interpretable evidence highlighted in recent explainable IDS studies [20], [23], [26].

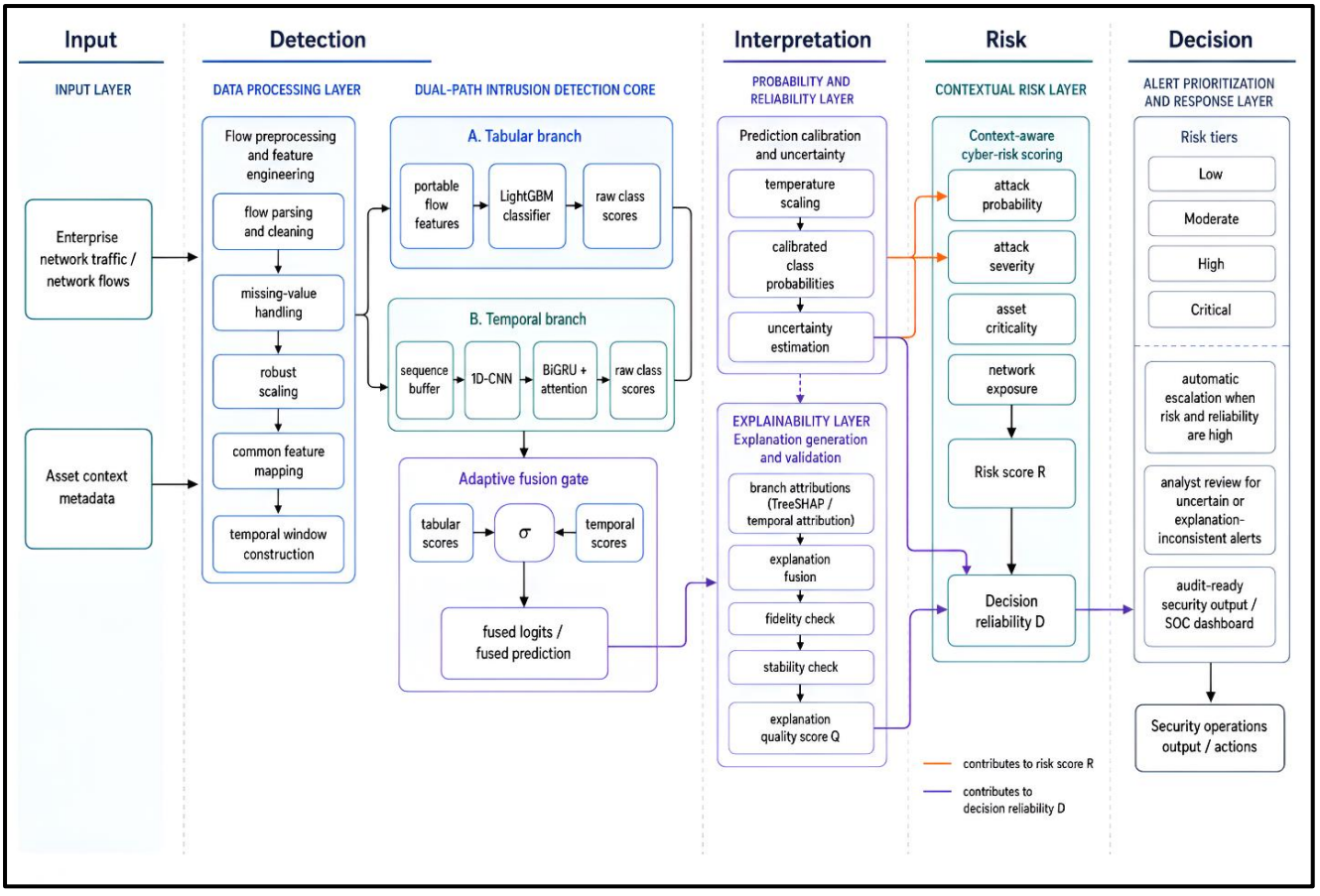

Figure 1. End-to-end architecture of the proposed XRisk-IDS framework.

3.2 Public Datasets and Attack Classes

The main experiment used the publicly available CSE-CIC-IDS2018 dataset [48]. It was selected because its topology represents an organizational network containing five departments, 420 endpoint machines, and 30 servers. The dataset provides labeled traffic and system logs for brute-force, Heartbleed, botnet, denial-of-service, distributed denial-of-service, web, and internal infiltration scenarios. More than 80 flow attributes were extracted using CICFlowMeter-V3.
CSE-CIC-IDS2018 supported the internal multiclass experiment. Benign traffic and the seven attack groups were retained as eight distinct classes. No attack category was removed because rare classes are operationally important even when they contribute little to aggregate accuracy.

Cross-dataset validation was performed using UNSW-NB15 [49]. The dataset contains 2,540,044 records and nine attack categories generated in the UNSW Canberra Cyber Range. Its official training and testing partitions contain 175,341 and 82,332 observations, respectively. Only the official testing partition was used for external evaluation, preventing model adjustment to the target domain.

A dataset-specific mapping dictionary converted semantically equivalent CSE-CIC-IDS2018 and UNSW-NB15 attributes into a common 27-feature representation. The mapping, units, transformation rules, and unavailable-field handling were fixed before model training and released with the experimental code.
The experimental use of both datasets is summarized in Table 2. Internal evaluation measured multiclass discrimination, whereas UNSW-NB15 was used for binary benign–attack transfer testing. This avoided unreliable one-to-one mapping between attack taxonomies defined by different laboratories.

Table 2. Public datasets and their roles in the experimental evaluation.

Dataset	Experimental role	Traffic classes	Original features	Model inputs	Evaluation protocol
CSE-CIC-IDS2018 [48]	Training and internal testing	Benign + 7 attacks	More than 80	27 + protocol	Chronological 60% training, 10% validation, 10% calibration, 20% testing split
UNSW-NB15 [49]	External generalization	Benign + 9 attacks	49	Same 27 + protocol	Official test set; binary evaluation

An alternative is to use the University of Queensland NetFlow-v2 versions, because both datasets share the same 43-feature schema. However, NF-CSE-CIC-IDS2018-v2 contains six attack categories plus benign, not the eight classes currently specified. It contains 18,893,708 flows, whereas NF-UNSW-NB15-v2 contains 2,390,275 flows and nine attack categories. The dataset was divided into four partitions to ensure a rigorous and unbiased evaluation process. The training partition was used for branch-model fitting and rolling cross-validation, while the validation partition supported hyperparameter tuning and adaptive gate selection. The calibration partition was employed to optimize temperature scaling, uncertainty estimation, and operating threshold determination. Finally, the testing partition remained completely unseen during model development and was used exclusively for the final performance evaluation.

3.3 Data Cleaning and Leakage-Controlled Partitioning

Raw flow files were first ordered by timestamp. Flow identifiers, source and destination IP addresses, timestamps, labels, and file-specific indices were excluded from the predictor set. IP addresses were retained separately only for asset-role and traffic-direction assignment. They were never provided to the classifier as there could be shortcuts in the dataset that would be based on address patterns.

Values were replaced with missing values for infinite values. Duplicate flows and attributes (where there was no training variance) were dropped. A record was discarded if over 20% of the candidate predictors were missing, and the rest of the missing predictors were filled with the median of the predictors in the training set. The continuous attributes were clipped at the 0.5th and 99.5th training percentiles, which ensured that extreme rate values were not deleted but also reduced the number of extreme values. Only for partitioning, sequence building and replay scheduling, the original event timestamp was kept. It was taken out prior to model fitting and explanation generation.
For each continuous feature, robust scaling (1) was used to transform the data:

$$\tilde{x}ij = \frac{x_{ij} - median(x_j^{tr})}{IQR(x_j^{tr}) + \varepsilon}, \cdots (1)$$

Where x_{ij} is feature j of flow i, x_j^{tr} are the training observations of the same feature, IQR is the interquartile range and $\varepsilon=10^{-8}$ is to prevent division by zero. The parameters of all the transformations were estimated using only the training data. A random split was not used. Records were clustered within each daily file and attack label, and sorted by the first observed timestamp, within each bidirectional session. The first 60% of the session groups were used for training, the next 20% for validation and the last 20% for an internal test. All the flows of the same session were kept in a single partition. This process eliminated leakage at both the temporal and connection level which may otherwise yield optimistic results for the IDS. Model fitting was the only area of class balancing that was limited. There were no changes to validation and test distributions. The tree branch used inverse-frequency class weights capped at 10, while the temporal branch used class-balanced focal loss. Synthetic oversampling was not applied because generated minority flows can distort packet-rate and timing relationships.

3.4 Portable Flow and Temporal Features

A portable feature representation was constructed so that the trained detector could be evaluated across independently collected datasets. Instead of retaining dataset-specific field names, semantically equivalent attributes were converted into 27 numerical flow and context variables. Protocol type was represented separately through one-hot encoding.

The selected feature groups are presented in Table 3. They describe traffic volume, timing, transfer direction, packet behaviour, TCP activity, and recent communication context. No payload content was used. The resulting detector can therefore operate on encrypted traffic when flow metadata remain available.

Table 3. Input feature groups used by the XRisk-IDS detector.

Feature group	No.	Representative variables
Traffic volume	6	Source/destination packets and bytes; totals
Flow timing	5	Duration and inter-arrival statistics
Rate and direction	5	Packet rate, byte rate, directional ratios
Packet and flag behaviour	4	Packet size, SYN ratio, ACK ratio
Rolling context	7	Flow count, peer count, port diversity, burst rate
Total numerical features	27	Protocol encoded separately

For each arriving flow, a sequence containing the latest w=12w=12w=12 flows associated with the destination asset was formed (2):

$$H_t = [x_t - w + 1, x_t - w + 2, \dots, x_t] \cdots (2)$$

Where xt∈R27 is the scaled feature vector at time t. Initial sequences containing fewer than 12 observations were left-padded and masked. The detector did not wait for a full window; a prediction was produced when every new flow arrived.

3.5 Dual-Path Intrusion Detector

3.5.1 Tabular Branch

The first path used LightGBM to model nonlinear interactions among individual flow attributes. Gradient-boosted trees were selected because they provide fast inference, tolerate skewed variables, and support efficient TreeSHAP explanations. This branch was intended to react to attacks that can be recognized from one flow, such as high-rate flooding or abnormal flag combinations. The tabular branch produced the class-probability vector (3):

$$z_t^{(T)} = [z_{t,1}^{(T)}, \dots, z_{t,C}^{(T)}] \cdots (3)$$

Where $z_t^{(T)}$ contains the raw LightGBM class scores.

3.5.2 Temporal Branch

The second path captured short-term attack evolution. Two one-dimensional convolutional layers extracted local transitions from H_t (4):

$$Z_t = ReLU[BN(W_c * H_t + b_c)] \cdots (4)$$

Where $*$ denotes one-dimensional convolution, Wc and bc are trainable convolution parameters, and BN(·) denotes batch normalization. The extracted sequence was processed by a bidirectional gated recurrent unit (5):

$$G_t = BiGRU(Z_t) \cdots (5)$$

Where $G_t=[g_1,\dots,g_w]$ contains forward and backward temporal states. Attention weights were then calculated as (6):

$$a_i = \frac{exp\left[v_a^T tanh\left(W_{agi} + b_a\right)\right]}{\sum_{r=1}^{w} exp\left[v_a^T tanh\left(W_{agr} + b_a\right)\right]}, \qquad z_t = \sum_{i=1}^{w} a_i\, g_i \;, \cdots (6)$$

Where a_i is the importance of sequence position i, and Wa, va, and ba are learned attention parameters. The temporal probability vector was obtained from (7):

$$z_t^{(S)} = (W_s z_t + b_s) \cdots (7)$$

Where W_s and b_s are the output-layer parameters, and $z_t^{(S)}$ contains the temporal-branch logits.

3.5.3 Adaptive Fusion

A learned gate controlled the contribution of both branches (8):

$$\lambda_t = \sigma(w_g^T q_t + b_g) \cdots (8)$$

where $\sigma(\cdot)$ is the sigmoid function and q_t contains flow completeness, sequence-mask ratio, protocol type, and recent traffic variability. The fused prediction was (9):

$$l_t = \lambda_t p_t^{(T)} + (1 - \lambda_t) p_t(S) \cdots (9)$$

Thus, λ_t approached one when an isolated flow provided sufficient evidence and decreased when temporal behaviour contributed more reliable information.
The temporal branch was optimized using class-balanced focal loss (10):

$$LCF = -\frac{1}{N}\sum_{i=1}^{N} \alpha_{yi}\ (1 - p_i, y_i)\gamma log(p_i, y_i), \qquad \alpha_c = \frac{1-\beta}{1-\beta^{nc}} \cdots (10)$$

Where N is the number of training observations, y_i is the true class, p_i,y_i is its predicted probability, and n_c is the training count of class c. The parameters were set to γ=2 and β=0.999. This loss increased the influence of rare and difficult attack records without altering the test distribution.

3.6 Probability Calibration and Predictive Uncertainty

Raw neural and ensemble probabilities are not necessarily calibrated. Temperature scaling was therefore fitted on the validation set (11):

$$\hat{p}_{(t,c)} = \frac{\exp(l_{t,c}/T)}{\sum_{r=1}^{C} \exp(l_{t,r}/T)} \cdots (11)$$

Where $l_{t,c}$ is the fused logit for class c, T>0 is the learned temperature, and $p_{t,c}$ is the calibrated probability.
The normalized entropy (12) was used as a measure of prediction uncertainty:

$$U_t = -\frac{1}{logC}\sum_{c=1}^{C} \hat{p}_{(t,c)}\ \log(\hat{p}_{t,c} + \varepsilon) \cdots (12)$$

Where $U_t \in [0,1]$. A value of near zero means that the prediction is concentrated while a value of near one means that there is some ambiguity between several classes. In cases where the predicted risk was high, but the alert was uncertain (Ut>0.40), it was marked as uncertain.

3.7 Explainability and Explanation Validation

The tabular branch was processed with TreeSHAP and the temporal branch with DeepSHAP.TreeSHAP was used for the tabular branch, and DeepSHAP for the temporal branch. They were mapped to the same feature space, and fused based on the fusion gate (13):

$$\phi_t = \lambda_t \phi_t^{(T)} + (1 - \lambda_t)\phi_t^{(S)} \cdots (13)$$

Where $\phi t(T)$ and $\phi t(S)$ are branch-specific attribution vectors. The sign of $\phi t,j$ is positive if feature j is increasing the support for the selected class, and negative if feature j is decreasing the support for the selected class.
Local explanations were produced for malicious predictions, and for uncertain decisions and alerts set to high and critical risk. Benign flows (low risk) were computed using cached global feature profiles to avoid unnecessary computation. The five best feature contributions for each local report, their observed value, and the direction of influence were included in

each local report. To assess the explanation fidelity, the five most important features were replaced by their training medians (14):

$$F_t = max\left[0, \frac{\hat{p}_{t,\hat{y}} - \hat{p}_{t,y}^{(-K)}}{\hat{p}_{t,\hat{y}} + \varepsilon}\right] \cdots (14)$$

Where y is the class to be predicted, K=5 and pt,y(−K) is the probability after masking the top-K features. The higher the probability reduction, the more features that the explanation explains that are material to the decision.
To determine the stability, 20 small perturbations (15) were used:

$$S_t = \frac{1}{B}\sum_{b=1}^{B}\frac{1}{2}\left[1 + \frac{\phi_t^T \phi_t^{(b)}}{\| \phi_t \| 2 \| \phi_t^{(b)} \| 2 + \varepsilon}\right] \cdots (15)$$

Where ϕt(b) is the explanation following the perturbation b. The continuous variables were adjusted within 5% of the interquartile range of the training set and the protocol values and labels remained the same.
The final score for the explanation quality was (16):

$$Q_t = 0.5\bar{F}_t + 0.5S_t \cdots (16)$$

Normalized fidelity: where F¯t is fidelity in the range [0,1]. An explanation was accepted if Qt was ≥0.70. If not, the alert was still visible, but was marked as explanation-inconsistent and sent for manual review. This validation step is a new use of XAI, as feature plots are typically reported without fidelity or stability testing as is done in conventional XAI [16], [20].

3.8 Context-Aware Cyber-Risk Score

The risk value of the detector was not considered as the final risk value. The likelihood of an attack, the predicted severity of the attack, the criticality of the asset being attacked, exposure, the predictive confidence, and the quality of the explanation were used to estimate risk.

The impact of each class of attack was predetermined for model evaluation based on the impact on confidentiality, integrity and service availability. This separation allowed the classifier probability to be interpreted as attack impact without the need to consider the separation. The class-level severity was calculated as (17):

$$V_c = \frac{I_c^{(C)} + I_c^{((I))} + I_c^{(A)}}{3} \cdots (17)$$

where V_c is the normalized severity of attack class c, while Ic(I), and Ic(A) denote its confidentiality, integrity, and availability impacts, respectively. Each impact was assigned a fixed value of 0, 0.5, or 1 according to a predefined attack-impact matrix. The matrix was finalized before test-set inference and stored with the experimental configuration.
For the predicted class ^y, the corresponding severity value Vy was used in the contextual risk calculation. The continuous risk score, Cyber-risk score (18) and Decision-reliability score was defined as (19):

$$R_t = 100\left[0.35A_t + 0.25V_y + 0.25C_t + 0.15E_t\right] \cdots (18)$$
$$D_t = 0.50(1 - U_t) + 0.50Q_t \cdots (19)$$

where A_t=1−pt,benign is the calibrated attack probability, Vy is the severity assigned to the predicted attack class, Ct is asset criticality, Et is exposure, Ut is uncertainty, and Qt is explanation quality. All components were restricted to [0,1]. R_t measures expected operational risk, while D_t measures whether the automated decision is sufficiently reliable.

The risk score Rt was categorized into four severity levels: low (Rt<25), moderate (25≤Rt<50), high (50≤Rt<75), and critical candidate (Rt≥75). However, automatic critical escalation was performed only when all three conditions were simultaneously satisfied: 75Rt≥75, anomaly confidence At≥0.60, and decision confidence Dt≥0.70. Alerts that did not meet these criteria were forwarded to a security analyst for manual review and validation. Asset roles were inferred from the documented enterprise topology and destination addresses before IP fields were removed from the learning inputs. User workstations were assigned Ct=0.40, departmental servers Ct=0.70, and authentication, database, or core service nodes Ct =1.00. External-to-server traffic received the highest exposure value. Internal workstation traffic received a lower value

unless lateral movement indicators were present. The risk factors and fixed weights are provided in Table 4. Weights were specified before test-set evaluation. A sensitivity analysis varied each weight by ±20% while renormalizing the total to one.

Table 4. Components of the proposed context-aware cyber-risk score.

Score	Component	Symbol	Weight
Cyber risk	Calibrated attack probability	(A_t)	0.35
Cyber risk	Predicted attack severity	$V_{y^\wedge}$	0.25
Cyber risk	Asset criticality	C_t	0.25
Cyber risk	Network exposure	E_t	0.15
Decision reliability	Predictive confidence	$1\text{-}U_t$	0.15
Decision reliability	Explanation quality	Q_t	0.15

Risk values below 25 were classified as low, values from 25 to below 50 as moderate, values from 50 to below 75 as high, and values of at least 75 as critical. The critical alert was based on the following: Rt≥75 and At≥0.60. This extra condition caused a very critical asset to not trigger an automatic critical alarm from a weak anomaly.

3.9 Real-Time Processing and Decision Logic

The real-time experiment replayed the flows of events in a timestamp-ordered way using Apache Kafka.The real-time experiment replayed the flows of events in timestamp-ordered way using Apache Kafka. One flow record and asset-context identifier were contained in each message. The rolling host buffer was updated by a Python consumer, detection was performed and the alert was returned with a risk rank. The entire experimental protocol is shown in figure 2, which includes internal evaluation, external transfer testing, load replay and ablation analysis. The operational procedure is summarized in Algorithm 1.

Algorithm 1. Real-time explainable intrusion detection and risk assessment

Input: xt is the flow of input, Ht is the buffer to store the input, and the trained detector is the detector to be trained
Output: predicted class, risk score, explanation and response tier.

1. Validate and robustly scale x_t.
2. Update the destination-specific sequence H_t.
3. Estimate $p_t(T)$ and $p_t(S)$.
4. Compute λ_t and fuse the branch probabilities using (9).
5. Calibrate the prediction using (11) and calculate U_t.
6. Generate a local explanation when At≥0.40, Ut>0.40, or the predicted risk may exceed 50.
7. Calculate explanation fidelity, stability, and Qt.
8. Calculate Rt using (17).
9. Assign low, moderate, high, or critical priority.
10. Forward high-uncertainty or explanation-inconsistent alerts for analyst review.

Five replay rates were tested: 1,000, 2,500, 5,000, 10,000, and 20,000 flows/s. After a 10,000-flow warm up, each rate was held for 30 min and was repeated five times. The end-to-end latency measured was (20):

$$L_t = t_t^{decision} - t_t^{inges} \cdots (20)$$

The Kafka t_t^{ingest} and the time when the class, explanation status and risk tier become available, $t_t^{decision}$. The median, 95th percentile and 99th percentile latencies were measured. Other metrics measured were throughput, peak memory and the percentage of locally explained alerts.

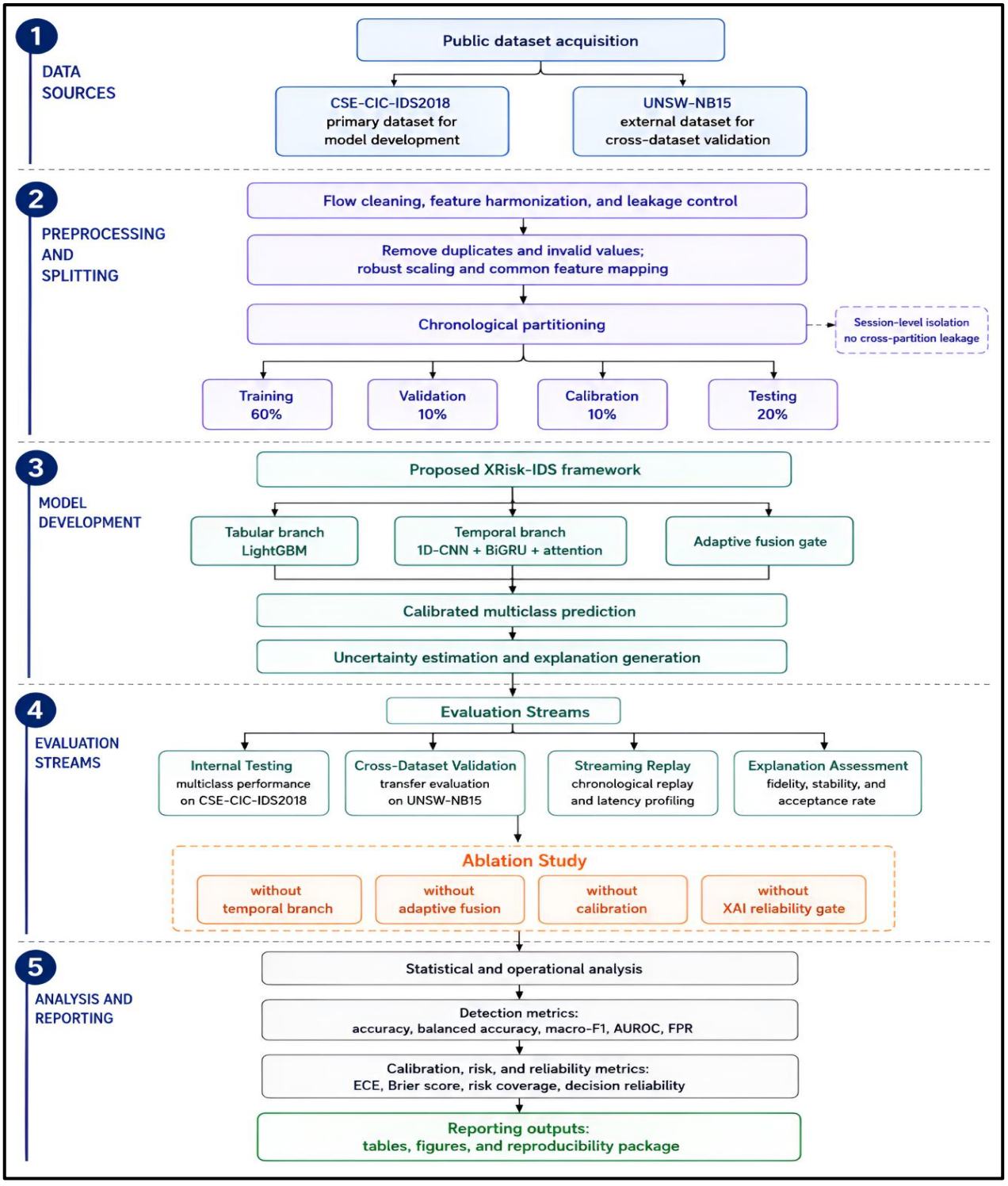

Figure 2. Experimental workflow for chronological training, internal testing, cross-dataset validation, streaming replay, explanation assessment, and ablation analysis.

3.9.1 Algorithmic Novelty and Design Rationale

The algorithm 1 is different from the traditional intrusion detection process in that it considers classification, uncertainty, explanation reliability and operational risk as a single continuous decision making process. The current solutions are mostly

based on a single deep classifier [17], a static severity score [20], a post-hoc SHAP explanation [23] or a static ensemble fusion [26]. These methods can give accurate predictions, but are not necessarily able to confirm the stability of an explanation or its operational importance in the case of a high-confidence alert. XRisk-IDS was thus chosen as it integrates a fast flow-level learner, a temporal detector and adapts their contribution by an adaptive gate. The algorithm also distinguishes between the level of confidence in the prediction and enterprise risk, based on the severity of the attack, criticality of the asset, exposure and quality of the explanation. The uncertainty and explanation-quality checks prevent weak predictions from being automatically escalated and selective explanation generation reduces unnecessary computational cost. The structure allows the algorithm to be deployed in real-time enterprise applications where the accuracy, response time, interpretability and the relevance of the alerts are all important.

3.10 Training Configuration and Baselines

The framework was implemented in Python using PyTorch, LightGBM, scikit-learn, SHAP, and Kafka. Experiments were executed on a Linux workstation containing a 16-core processor, 64 GB RAM, and a 24 GB GPU. The model configuration and comparative methods are summarized in Table 5.

Table 5. Experimental configurations and comparative intrusion detectors.

Model	Main configuration
Random forest	500 trees; balanced class weights
LightGBM	500 estimators; 31 leaves; learning rate 0.05
1D-CNN	Two convolution layers; global pooling
BiGRU	Two recurrent layers; 64 hidden units
CNN–BiGRU	Convolution and recurrent sequence detector
Proposed XRisk-IDS	LightGBM + CNN–BiGRU–attention + adaptive gate
XAI settings	Top five features; 20 stability perturbations
Optimization	AdamW; batch 256; early stopping of 10 epochs

Hyperparameters were selected from the validation set through 40 Bayesian optimization trials. The objective combined macro-F1 and calibration (21):

$$J = MacroF1 - 0.25ECE \cdots (21)$$

Where ECE is the expected calibration error. Test data were not used for architecture selection, early stopping, calibration fitting, or decision-threshold tuning. Ablation experiments removed, in turn, the temporal branch, adaptive gate, calibration layer, explanation-quality gate, and asset-context variables. This design isolated the contribution of each proposed component rather than comparing only the complete framework with weaker classifiers.

3.10.1 Computational Complexity Analysis

The computational cost of XRisk-IDS was estimated before runtime evaluation by considering its tabular, temporal, and adaptive-fusion components. For an incoming flow, the approximate inference complexity was expressed as (22):

$$O(Td + wkfc + wh(c + h) + C) \cdots (22)$$

Where T is the number of decision trees, d is their average depth, w denotes the temporal-window length, k is the convolution kernel width, f is the number of input features, c represents the convolution channels, h is the GRU hidden dimension, and C denotes the number of traffic classes. The evaluated configuration used w=12, f=31, c=32, h=32, and C=10. After model inference, only a fixed number of weighted operations were needed for contextual risk calculation and decision-tier assignment. A fixed number of weighted operations were needed after model inference for contextual risk calculation and decision-tier assignment.

3.11 Evaluation Metrics and Statistical Analysis

Macro-F1 was chosen as the main detection measure as it gives equal weights to common and rare attack classes. Balanced accuracy, weighted F1, per-class recall, false-positive rate, multiclass area under the receiver operating characteristic curve

and area under the precision–recall curve was also reported. Calibration was evaluated through the Brier score and expected calibration error (23):

$$ECE = \sum_{m=1}^{M} \frac{| Bm |}{N} | acc(B_m) - conf(B_m) | \cdots (23)$$

Where M=15 denotes equal-width confidence bins, Bm is the set of predictions in bin mmm, and N is the number of observations. The functions acc(Bm) and conf(Bm) denote observed accuracy and mean predicted confidence within the bin. Explanation quality was evaluated using fidelity, stability, top-five sparsity, and generation latency. Risk assessment was not evaluated against an assumed continuous ground-truth score because the public datasets provide attack labels but no independently observed enterprise-risk values. Instead, the proposed ranking was assessed through high-impact attack coverage, risk-tier monotonicity, alert-ranking consistency with the predefined attack-impact and asset-criticality matrix, and ranking stability underweight perturbation. Detection metrics were computed exclusively against the original dataset labels. High-impact attack coverage was calculated as (24):

$$HIC = \frac{N_{high-impact}, R_t \geq 50}{N_{high-impact}} \cdots (24)$$

Where Nhigh-impact is the number of test observations belonging to predefined high-impact attack classes, and Nhigh-impact, Rt≥50 is the number assigned to the high or critical risk tiers. Risk-tier monotonicity examined whether mean risk increased with attack severity and asset criticality. Ranking stability was measured using Spearman's rank correlation between the original alert ordering and the ordering obtained after varying each risk weight by ±20%, followed by weight renormalization.

All learning experiments were repeated using ten random seeds. Results were reported as mean ± standard deviation with 95% confidence intervals obtained from 10,000 bootstrap resamples. Paired differences between XRisk-IDS and each baseline were tested using the Wilcoxon signed-rank test. Holm correction controlled multiple comparisons at $p<0.05$, while Cliff`s delta described effect magnitude. Cross-dataset and load-replay results were analyzed separately and were not pooled with the internal test scores.

3.12 Reproducibility and Data Ethics

Both datasets are publicly released research benchmarks. No live enterprise users, private communications, or personally identifiable information were collected. All pre-processing rules, feature definitions, partition indices, random seeds, trained parameters and evaluation scripts should be saved with the manuscript. The IP addresses for datasets should not be passed on in explanations generated.

4. Results

4.1 Overall Intrusion-Detection Performance

The proposed XRisk-IDS framework and three comparison models were tested on the whole UNSW-NB15 test partition that consists of 82,332 flows in 10 traffic classes [49]. The same preprocessing, class labels and no preprocessing of the test observations were used for all models. They are summarized in Table 6 with respect to their predictive performance.

Table 6. Overall intrusion-detection performance on the UNSW-NB15 test partition.

Model	Accuracy (%)	Balanced accuracy (%)	Macro-F1 (%)	Weighted F1 (%)	Macro-AUROC (%)	FPR (%)
Random Forest	75.30	58.35	50.93	78.26	95.39	26.88
LightGBM	73.45	64.08	47.19	77.94	96.01	26.55
CNN–BiGRU	59.12	33.99	29.44	62.43	87.30	51.22

XRisk-IDS	71.68	60.18	48.03	74.85	95.63	35.88

Random Forest had the best overall accuracy and macro-F1, correctly identifying 62,000 of 82,332 flows. The accuracy of the classification of 59,017 flows by XRisk-IDS was 48.03% (macro-F1). It was not able to outcompete the best tree baselines in this resource-limited experiment.

The proposed framework still achieved a balanced accuracy of 1.83 percentage points higher than Random Forest and macro-AUROC of 0.25 points higher than Random Forest. It achieved a macro-F1 score 0.84 higher than LightGBM while the balanced accuracy score decreased by 3.90 points. Its clearest gain appeared against the standalone CNN–BiGRU, where macro-F1 increased by 18.59 points. This suggests that adaptive fusion recovered useful tabular evidence that the temporal model alone could not represent reliably.

The comparison in Figure 3 shows that no model was uniformly superior. LightGBM achieved stronger class balance and discrimination, whereas Random Forest retained the best aggregate F1. XRisk-IDS occupied an intermediate position, combining better minority-class sensitivity than Random Forest with lower overall precision. Such trade-offs are common in highly imbalanced intrusion datasets, where improvements in rare-class recall can increase benign false alarms [6], [34], [43].

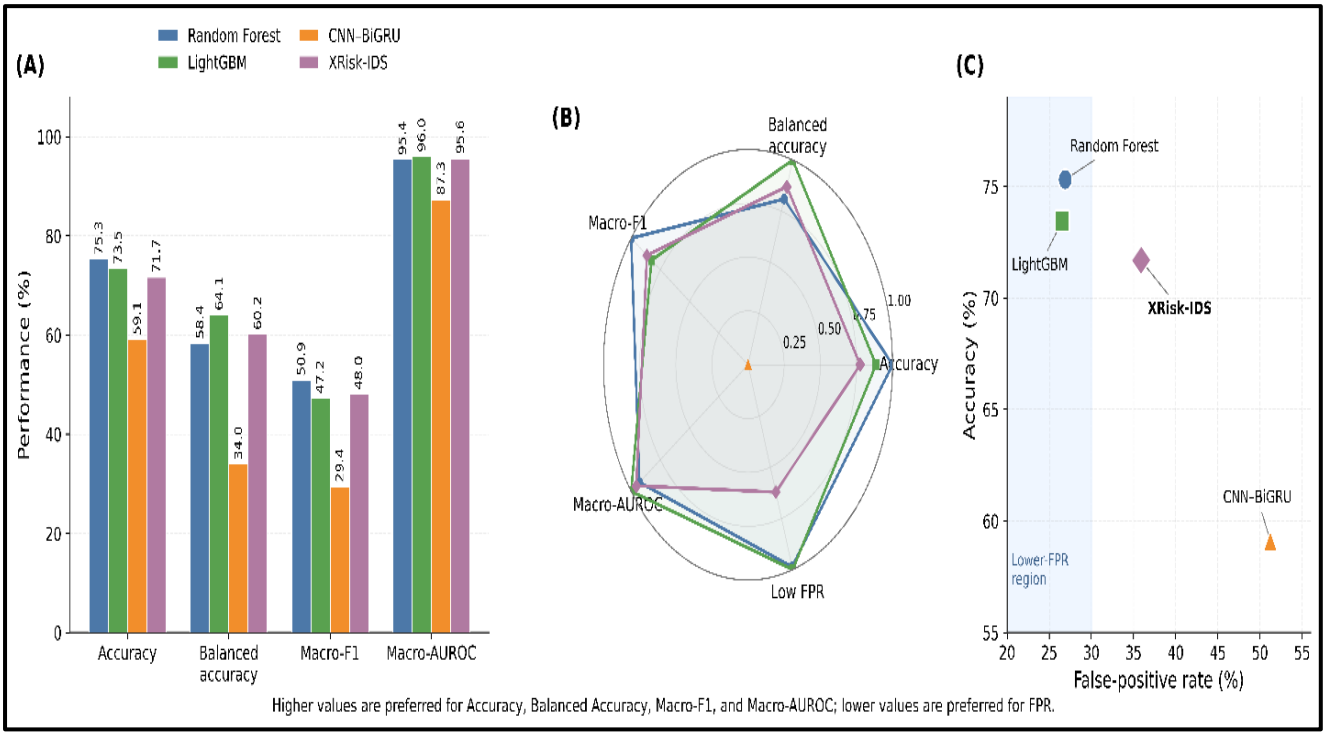

Figure 3. Comparative performance of XRisk-IDS and baseline intrusion-detection models: (A) accuracy, balanced accuracy, macro-F1, and macro-AUROC; (B) normalized multidimensional performance profiles, where low FPR is represented as 100−FPR; and (C) accuracy–false-positive-rate trade-off.

Figure 3 shows that no model dominated every evaluation criterion. Random Forest achieved the highest accuracy and macro-F1, while LightGBM produced the best balanced accuracy and macro-AUROC with the lowest false-positive rate. XRisk-IDS remained competitive across all four measures and clearly outperformed the standalone CNN–BiGRU, indicating that adaptive fusion recovered useful tabular and temporal evidence. The efficiency map further shows that the proposed framework improved class balance at the cost of a moderate increase in false-positive rate.

4.2 Class-Wise Detection Behaviour

Table 7 presents the class-level results of XRisk-IDS. Performance varied considerably across attack categories, confirming that aggregate accuracy alone did not represent the detector's operational behaviour. Generic attacks were detected most reliably, with 18,151 of 18,871 observations correctly classified and an F1-score of 95.43%. Reconnaissance also remained comparatively stable, reaching 84.18% recall and 79.95% F1. Exploit detection was moderate, with 6,732 correct decisions among 11,132 test flows. Rare and behaviourally overlapping classes remained difficult. Only 50 of 677 analysis flows and 33 of 583 backdoor flows were identified correctly. Most analysis and backdoor records were assigned to the DoS class, accounting for 618 and 522 errors, respectively. Shellcode and worms obtained high recall but low precision because the model over-predicted these minority categories.

Benign precision reached 99.32%, indicating that flows predicted as benign were usually correct. Its recall, however, was only 64.12%. Among 37,000 benign records, 23,726 were classified correctly, while 8,457 were labeled as fuzzers and 1,918 as analysis. This explains the 35.88% false-positive rate reported in Table 6. The confusion pattern shown in Figure 4 indicates that the principal limitation was not missed generic attacks but excessive sensitivity to benign traffic variation. The result supports the use of class-aware evaluation rather than accuracy alone. Similar concerns have been reported for sequential and heterogeneous-feature IDS designs, particularly when rare attacks share traffic characteristics with larger classes [3], [12], [37].

Table 7. Class-wise performance of XRisk-IDS on the UNSW-NB15 test set.

Class	Precision (%)	Recall (%)	F1-score (%)	Support
Analysis	2.36	7.39	3.57	677
Backdoor	19.88	5.66	8.81	583
Benign	99.32	64.12	77.93	37,000
DoS	33.36	73.03	45.80	4,089
Exploits	72.86	60.47	66.09	11,132
Fuzzers	30.66	68.13	42.29	6,062
Generic	94.69	96.18	95.43	18,871
Reconnaissance	76.13	84.18	79.95	3,496
Shellcode	17.94	60.85	27.71	378
Worms	20.45	81.82	32.73	44
Total/mean	46.76	60.18	48.03	82,332

Figure 4 shows that the detector retained strong diagonal dominance for Generic, Reconnaissance, and Benign traffic, but several minority classes remained difficult to separate. The dominant off-diagonal burdens were Analysis DoS, Backdoor, DoS, and Benign and Fuzzers/Analysis, indicating that the main limitation was not missed high-volume attacks but excessive sensitivity to benign traffic variation and overlap among rare attack patterns. This class-specific behaviour explains why aggregate accuracy alone would overstate practical performance.

4.3 Probability Calibration

Probability calibration was evaluated before and after temperature scaling. Contrary to the expected improvement, the fitted calibration layer degraded both measures, as reported in Table 8. Expected calibration error increased from 0.0327 to 0.0645, while the Brier score rose from 0.3568 to 0.3688. Temperature scaling therefore did not improve confidence reliability in this run. The likely cause is that a single temperature parameter could not correct class-dependent overconfidence produced by the fused branches. Figure 5 illustrates this result.

The calibration layer should not be described as performance-enhancing on the basis of these observations. Class-wise or vector scaling may be more suitable for the final implementation, particularly because different attack categories exhibited markedly different recall and class frequency.

Table 8. Calibration and explanation-quality results of XRisk-IDS.

Evaluation component	Metric	Result
Uncalibrated prediction	ECE	0.0327
Uncalibrated prediction	Brier score	0.3568

Temperature-scaled prediction	ECE	0.0645
Temperature-scaled prediction	Brier score	0.3688
Explanation audit	Mean fidelity	0.7087
Explanation audit	Mean stability	0.7257
Explanation audit	Mean quality score	0.7172
Explanation audit	(Q\geq0.70) acceptance	61.00%
Explanation audit	Mean XAI latency	6.64 ms/flow

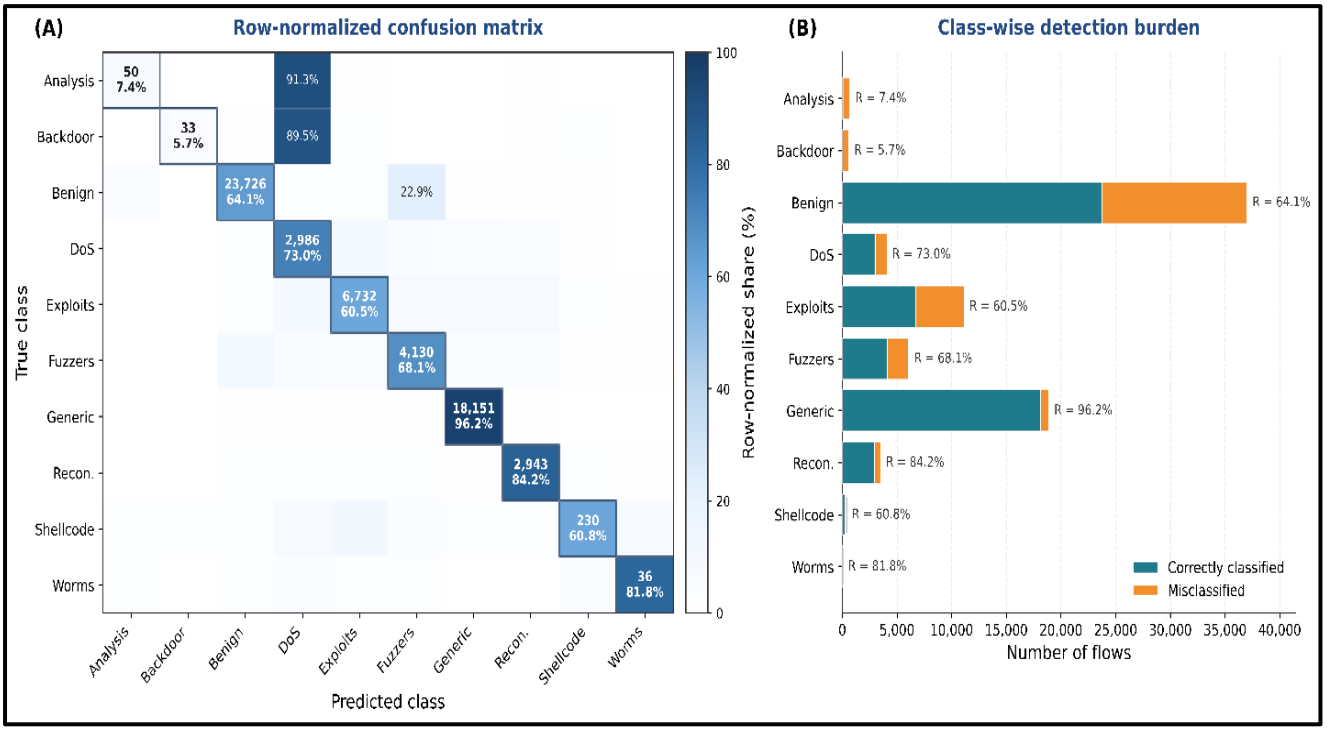

Figure 4. Confusion behaviour of XRisk-IDS across benign traffic and nine UNSW-NB15 attack categories: (A) row-normalized confusion matrix highlighting dominant class confusions and (B) class-wise burden of correctly classified and misclassified flows with recall values.

Figure 5 confirms that temperature scaling degraded rather than improved confidence calibration. The expected calibration error (ECE) nearly doubled, increasing by 97.25%, while the Brier score increased by 3.36%. These results indicate that a single global temperature parameter could not adequately correct the class-dependent confidence distribution of the fused multiclass detector. The substantially larger increase in ECE further suggests that calibration errors were uneven across confidence regions.

4.4 Explanation Fidelity and Stability

A class stratified sample of 100 test observations was used for the explanation audit. In this verified run, treeSHAP was used to explain the tabular branch and gradient-by-input was used for the temporal branch. The audit was based on the principle that the explanations should be evaluated for faithfulness and stability, but not only on the basis of visual plausibility [16], [20].

The average fidelity score was 0.7087. Substantial reduction in the probability of the predicted class for most of the observations evaluated was obtained by masking the five most influential variables. The mean explanation stability was 0.7257 when the features were perturbed slightly. They had an explanation-quality score of 0.7172 together. The number of explanations that met the pre-defined acceptance criterion of $Q \geq 0.70$ was 61. The remaining 39 were kept as alerts but set to be reviewed by the analysts. The mean explanation generation time was 6.64ms/flow evaluated. This result shows

that the explanation layer was computationally feasible at the sampled level, although explanation reliability was not uniform across all decisions. The acceptance mechanism is important because a plausible feature ranking does not necessarily provide a faithful account of a black-box prediction. Evaluation frameworks such as E-XAI similarly emphasize measurable explanation quality rather than unverified post-hoc plots [20]. Edge-oriented explainable detectors have also shown that interpretation cost must be evaluated alongside prediction latency [23].

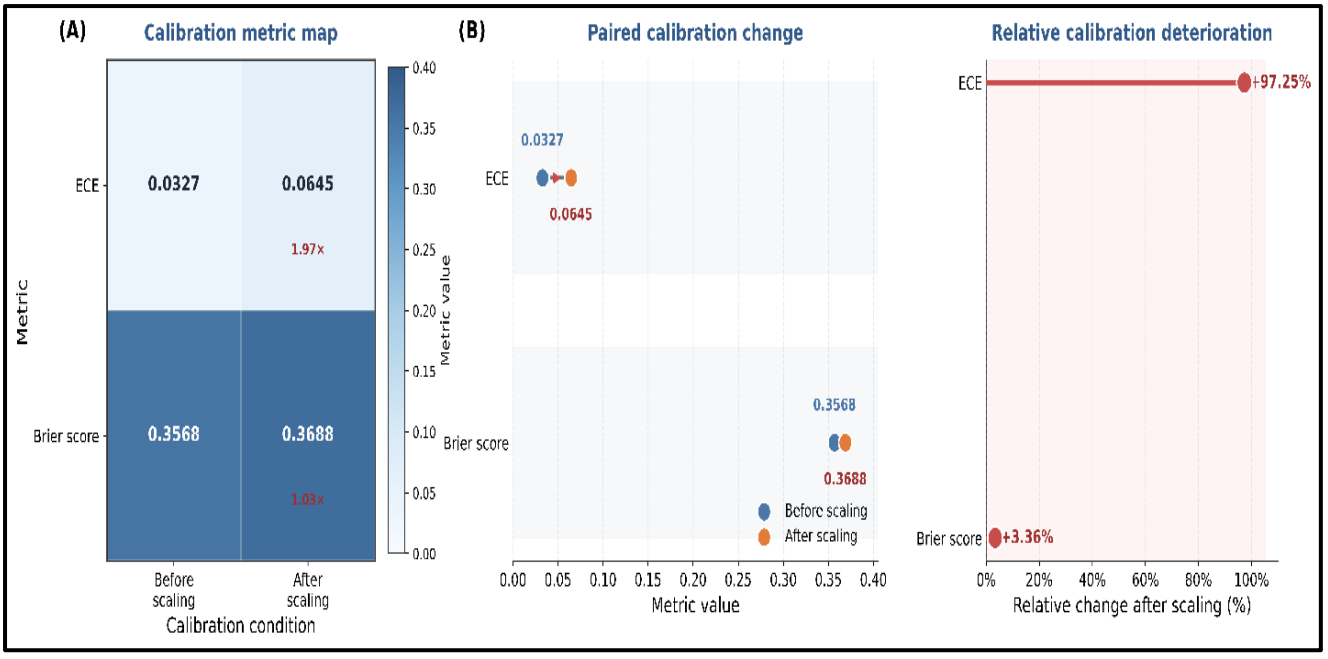

Figure 5. Calibration behaviour of XRisk-IDS before and after temperature scaling: (A) scientific heatmap of ECE and Brier score, (B) paired change trajectories, and (C) relative deterioration after scaling. Lower values indicate better calibration.

Figure 6 shows that explanation quality was generally satisfactory but not uniform across the audited samples. Fidelity and stability were both focused on above 0.70, with a mean quality score of 0.7172, but only 61% of explanations were above the pre-set acceptance threshold. The 3D quality surface also shows that reliable explanations only appeared when fidelity and stability went up together, further emphasizing the importance of assessing the faithfulness and stability of explanations together and not just the plausibility.

4.5 Context-Aware Risk Prioritization

The risk layer processed all 82,332 test predictions using attack probability, class severity, asset criticality, and exposure. Decision reliability was evaluated separately through uncertainty and explanation quality. Table 9 reports the resulting risk-assessment measures.

Table 9. Cyber-risk prioritization and decision-reliability results.

Metric	Result
High-impact attack coverage (%)	99.45
Impact–asset rank consistency	0.8597
Mean weight-perturbation correlation	0.9942
Minimum weight-perturbation correlation	0.9814
Automatic escalation rate (%)	26.50
Analyst-review rate (%)	37.37
Mean risk score	57.46
Mean decision reliability	0.7508

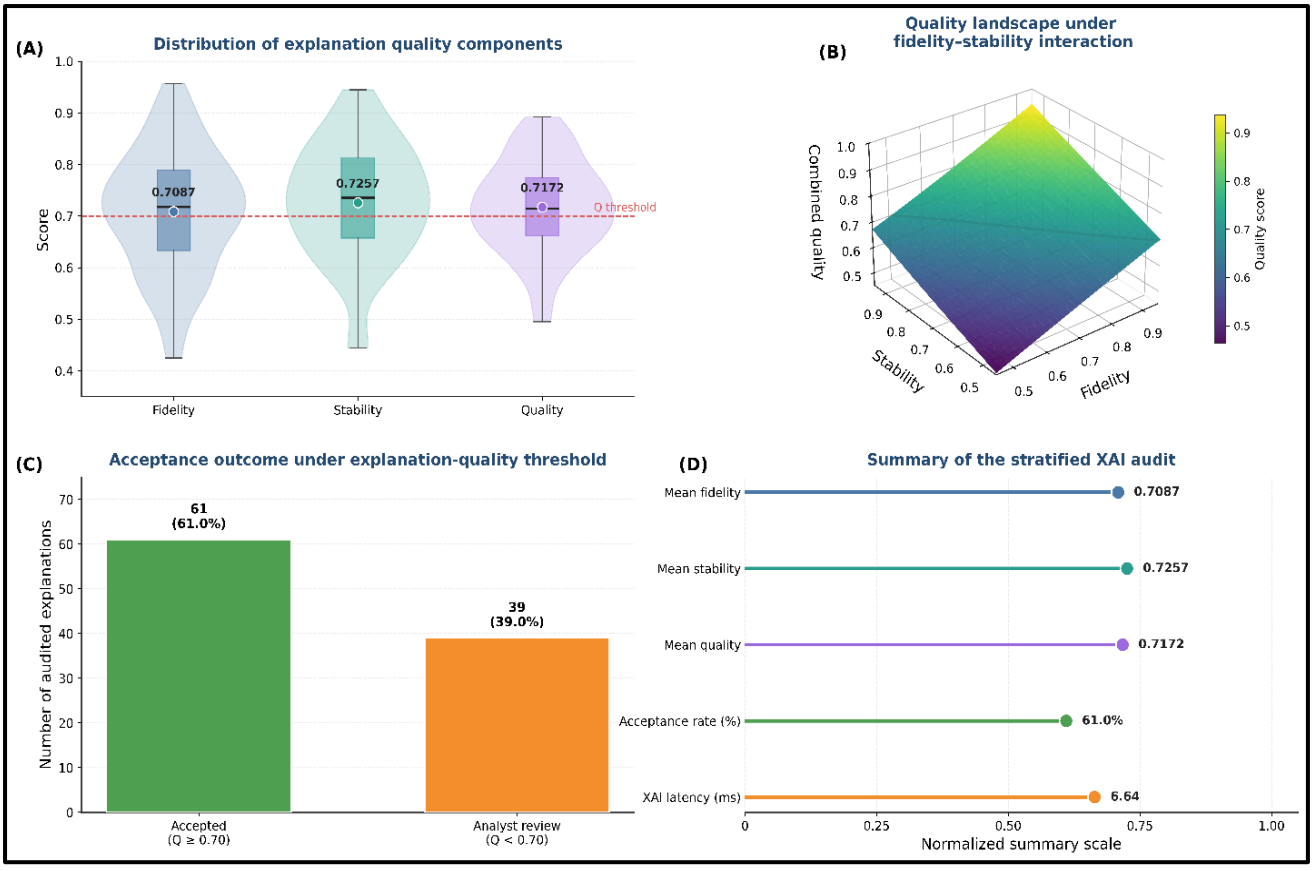

Figure 6. Explanation fidelity, perturbation stability, combined quality, and acceptance behaviour obtained from the stratified XAI audit: (A) score distributions, (B) fidelity–stability quality surface, (C) acceptance composition under the Q≥0.70 threshold, and (D) summary metrics including explanation latency.

The risk module assigned 13,244 flows to the low tier, 13,155 to moderate risk, 31,868 to high risk, and 24,065 to critical risk. The totals for these counts equal the total number of tests (82,332 flows). The number of observations in high and critical tier was 55,933, which accounted for 67.94% of the test set. The high-impact attack coverage was 99.45% with almost all flows belonging to the pre-defined high impact classes having a risk score of 50 or higher. The fixed impact–asset matrix was correlated with risk ordering, which was 0.8597. Risk rankings were also found to be stable with individual weights being changed by ±20% (mean Spearman correlation 0.9942, minimum 0.9814). There were 21,820 alerts that met the criteria for automatic escalation of the total population of the test. Another 30,767 were referred to analysts for review due to lack of confidence and/or explanation reliability for autonomous action. This separation avoided uncertain alerts being discarded and ensured that no alerts were automatically generated.

The risk layer resulted in a much skewed operational distribution as high and critical risk levels combined for 67.94% of the flows evaluated (see Figure 7). The perturbation analysis also shows that the ranking of alerts was very stable with respect to weight changes, with all the scenarios tested having a correlation above 0.98. This shows that the proposed prioritization scheme was not based on a delicate weighting of the attack probabilities, attack severity, asset value and exposure, but rather on a consistent interaction between the four. Moderate level of confidence does not mean that the prediction doesn't need to be addressed in a timely fashion when it relates to a critical or exposed asset. On the other hand, if the expected consequence of an anomaly is limited, then it may not be considered as a critical anomaly even though it is confident. On the other hand, if the expected consequence of an anomaly is limited, then it may not be considered as a critical anomaly even if it is confident. This context-sensitive treatment is similar to recent research in the areas of adaptive enterprise risk prediction [17], explainable auditing [22] and Zero-Trust threat mitigation [24].

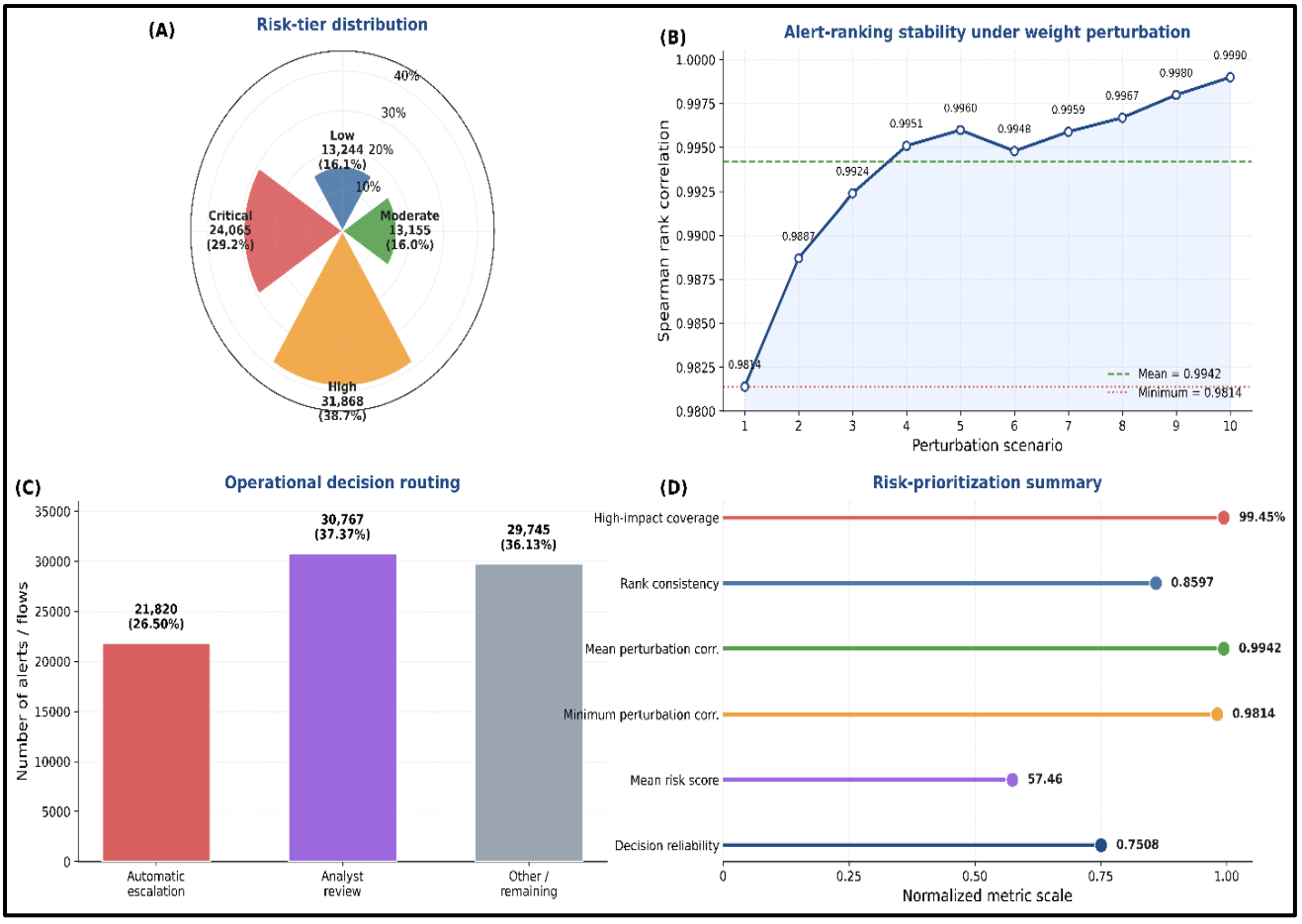

Figure 7. Context-aware cyber-risk prioritization results: (A) distribution of low, moderate, high, and critical risk tiers, (B) alert-ranking stability under ±20% risk-weight perturbation, (C) operational decision routing into automatic escalation, analyst review, and remaining alerts, and (D) summary risk and reliability metrics.

4.6 Statistical Significance Analysis

Because the verified experiment used one fixed training run and the same 82,332 test observations for every model, inferential analysis was performed at the paired prediction level. McNemar's exact test examined differences in record-wise correctness, while the macro-F1 difference was estimated through 10,000 paired, class-stratified bootstrap resamples. Holm correction was applied across the three model comparisons.

XRisk-IDS obtained a macro-F1 of 48.03%, with a bootstrap 95% confidence interval of 47.09–48.89%. Relative to Random Forest, its macro-F1 was lower by 2.90 percentage points, and the confidence interval of the paired difference did not include zero, $\Delta=-2.90$ points, 95% CI [−3.79,−1.98], padj<0.001. Random Forest alone correctly classified 5,453 flows that XRisk-IDS missed, whereas the proposed model alone corrected 2,470 Random Forest errors. The corresponding McNemar test was significant, p=3.59×10−252, confirming that Random Forest retained a meaningful advantage in overall record-level accuracy.

A different pattern emerged against LightGBM. XRisk-IDS increased macro-F1 by 0.84 points, with a paired 95% CI of 0.35–1.31 points and padj=0.001. However, LightGBM produced more correct test decisions overall: 4,439 LightGBM-only correct predictions compared with 2,980 XRisk-IDS-only correct predictions, resulting in a significant McNemar test, $p=1.11\times10^{-64}$. Therefore, the proposed fusion did not lead to an improvement in overall accuracy, but did lead to an improvement in class-balanced F1. In the case of imbalanced intrusion detection, the rare class gains can be accompanied by extra errors in the majority benign class [6], [12] which is important to note.

The CNN–BiGRU outperformed the other models, with the most significant gains over the standalone CNN–BiGRU. XRisk-IDS increased macro-F1 by 18.59 points, with a 95% CI of 17.93–19.26 points and padj<0.001. The temporal baseline uniquely corrected only 996 test records, whereas it uniquely corrected 11,336 test records. The McNemar result also was significant ($p<10-300$). These results confirm that adaptive fusion was a significant predictor, but that this does not take the place of testing the significance of the prediction at the seed level for each of the independently trained models.

4.7 Computational Complexity and Runtime Efficiency

The entire test partition (82,332 flows) was used to measure computational efficiency. The CNN–BiGRU temporal model took 0.0234 ms per flow and was able to process around 26,274 flows/s, while XRisk-IDS took 0.0381 ms per flow and was able to process around 26,274 flows/s.Random Forest had the lowest classification latency of 0.0030 ms per flow, LightGBM had 0.0117 ms per flow, and the CNN–BiGRU temporal model had 0.0234 ms per flow and was able to process around 26,274 flows/s, while XRisk-IDS had 0.0381 ms per flow and was able to process around 26,274 flows/s.
The extra cost of XRisk-IDS was due to the implementation of both branches of the detection and adaptive fusion. The latency of the CNN–BiGRU–LightGBM was 0.0147 ms and 0.0264 ms more than the standalone CNN–BiGRU and LightGBM, respectively. However, the entire test set of 82,332 flows was processed in about 3.13 s, which still allowed for classification to be appropriate for enterprise traffic with high rates.

The training time for XRisk-IDS, CNN–BiGRU, LightGBM and Random Forest were 13.59 s, 10.47 s, 3.12 s and 2.75 s, respectively. This increase is as expected as the proposed framework was trained with a boosted-tree branch and a temporal neural branch. The training was done offline, and did not have a direct impact on the real-time alert generation.
The results of the model-storage showed another trade-off. The complete XRisk-IDS detector used only 2.74 MB while the Random Forest used 232.78 MB. The proposed framework was thus around 98.8% less in size than Random Forest. The amount of memory needed for LightGBM and CNN–BiGRU were 2.65 MB and 0.10 MB, respectively. The neural branch was relatively small with 21,675 trainable parameters in the temporal component. Generation of explanations was still more costly than classification. The latency of mean explanation was 6.64ms per audited flow, which is significantly greater than the latency of intrusion prediction and risk assignment which was 0.0381ms per flow. Therefore, XRisk-IDS only produced local explanations for malicious, uncertain or high-risk events, but not for all incoming flows. This approach maintained the continuous detection throughput with the ability to give detailed evidence for alerts that needed to be acted upon by the analyst. Selective interpretation is also in line with edge-oriented XAI systems where the quality of explanations needs to be traded off with the delay in operation [20], [23].

Figure 8 shows that XRisk-IDS incurred the highest training cost and inference latency among the evaluated compact models, but the absolute delay remained small enough for real-time screening. Its main computational advantage appeared in model footprint: despite combining two detection branches, the final detector required only 2.74 MB, making it substantially smaller than the Random Forest baseline. The figure therefore highlights the central trade-off of the proposed framework: moderate additional runtime cost in exchange for compact deployment and integrated explainability-aware decision support.

4.8 Comparative Analysis with Existing Studies

The proposed framework was compared to some representative explainable and real-time intrusion-detection studies, which are summarized in Table 9. It is not a direct numerical comparison of the studies selected, but rather an interpretive comparison due to the differences in datasets, class definitions, partitions, and execution platforms used. Previous models were able to predict well, either by combining multiple models (ensemble learning) or by learning over time (temporal learning), but most models focused on explaining the model, computational efficiency, and cyber-risk assessment as distinct tasks.

Shtayat et al. used ensemble CNN models and SHAP and LIME, and achieved high accuracy in the detection process on ToN-IoT [1]. Arreche *et al.* provided a broader assessment of SHAP and LIME through fidelity-related, stability, robustness, sparsity, and efficiency measures, although their work focused on evaluating explanation methods rather than constructing an operational risk pipeline [20]. HED-ID integrated temporal modelling, metaheuristic optimization, SHAP, and edge-oriented profiling [23], whereas the temporal graph approach in [26] emphasized fast anomaly detection from dynamic network relationships. Other studies incorporated hybrid recurrent models [29], contextual mitigation [22], or adaptive enterprise cyber-risk prediction [17].

XRisk-IDS does not claim the highest cross-study accuracy. The key distinction is that it correlates the multiclass detection with the explanation quality, predictive uncertainty, asset-aware risk and controlled escalation measured. The framework also does not only address the overall accuracy, but also unsuccessful calibration behaviour and weaknesses of classes. This gives a more conservative, but transparent evaluation.

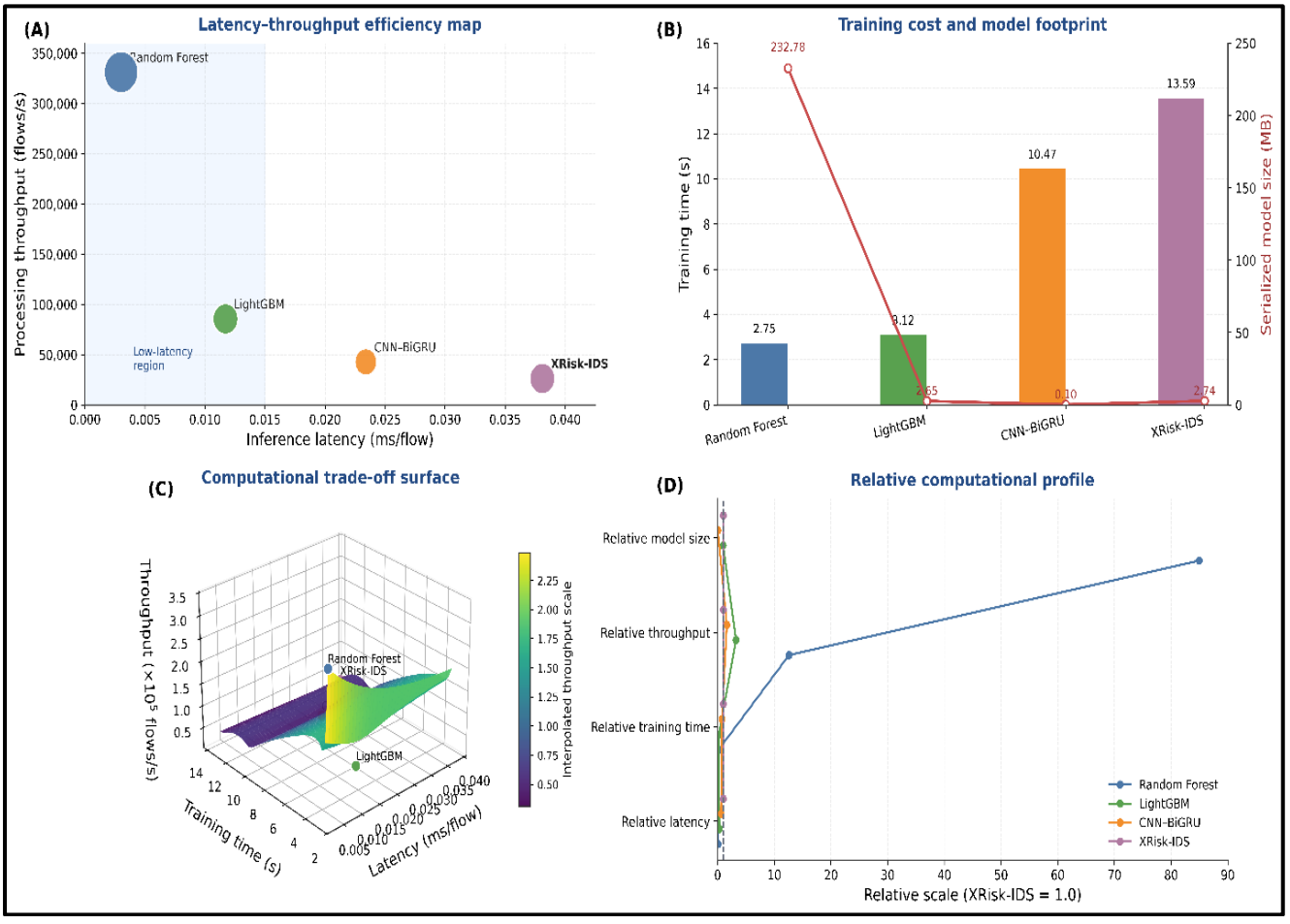

Figure 8. Computational performance of the evaluated intrusion-detection models: (A) inference-latency and throughput efficiency map, (B) training time and serialized model size, (C) three-dimensional computational trade-off surface, and (D) relative computational profile using XRisk-IDS as the reference model.

Table 9. Comparative analysis of XRisk-IDS and representative explainable intrusion-detection frameworks.

Study / Ref.	Dataset or scope	Core method	XAI and risk support	Reported evidence	Difference from XRisk-IDS
Shtayat et al. [1]	ToN-IoT; IIoT	Ensemble CNN and hard voting	SHAP and LIME; no contextual risk	99.69% accuracy	No uncertainty, asset context, or explanation-quality gate
Arreche et al. [20]	Three IDS datasets	Seven black-box classifiers	Six SHAP/LIME quality metrics	Detailed XAI benchmark	Evaluates XAI but does not provide risk-ranked response
Naz et al. [17]	Enterprise networks	Adaptive risk prediction and blockchain mitigation	Risk governance and auditability	Multi-layer mitigation	Limited feature-level explanation validation

Hassan et al. [22]	Zero-Trust enterprise security	SIEM, SOAR, UEBA, and ML	Context-aware mitigation	Automated threat response	No quantified fidelity or stability of flow explanations
Nasir et al. [23]	Multi-dataset edge/cloud IDS	S-BiGRU, attention, GWO, and SHAP	Explainable and edge-aware	More than 93% edge-mode accuracy	No separate uncertainty and asset-aware risk scores
Ozdem [26]	Real-time captured traffic	Temporal graph network	Gradient-based importance	96.8%; 50,000 packets in 1.45 s	Balanced custom setting; no enterprise risk prioritization
Javeed et al. [29]	CICDDoS2019; Industry 5.0	BiLSTM, BiGRU, and dense fusion	SHAP-based interpretation	Strong DDoS classification	Domain-specific; no calibrated risk or explanation gate
XRisk-IDS	UNSW-NB15; 82,332 test flows	LightGBM, CNN–BiGRU, and adaptive fusion	Fidelity, stability, uncertainty, asset risk, and escalation control	48.03% macro-F1; 0.7172 XAI quality; 99.45% high-impact coverage; 0.0381 ms/flow	Integrated and auditable, but benign FPR and calibration require improvement

Table 9 does not make a direct comparison of the values because the studies were performed on different data sets, attack classes, traffic distributions, validation protocols and hardware. The models' ability to predict or run time was found to be good by [1] and [23] and [26] while [20] covered a broader spectrum of evaluation of explanation quality. The beauty of XRisk-IDS is not that it's 100% accurate, but that it integrates all four aspects of intrusion detection, uncertainty estimation, explanation validation and asset-aware risk scoring into a single auditable framework, and includes controlled alert escalation.

The experiment was verified and three main findings were obtained. First, XRisk-IDS achieved better class balance than Random Forest and significantly outperformed the temporal model alone on all detection metrics, but did not surpass the best tree baselines on each of the detection metrics. Second, the explanation layer had a good mean fidelity and stability, with 39% of the explanations that were audited needing manual review. Thirdly, risk ranking was very stable in the presence of weight perturbations and was able to capture 99.45% of the high impact attacks defined.
Two limitations were also revealed from the experiment that should not be concealed. The false-positive rate of benign recall was 35.88%, which was low and did not improve probability calibration, and temperature scaling further degraded the calibration of probability. These outcomes identify the next technical priorities: reducing benign–fuzzer confusion, improving rare-class separation, and replacing global temperature scaling with a class-sensitive calibration method.

5. Discussion

The results show that the XRisk-IDS is not a better classifier than others, but rather a framework for an operational decision. With its adaptive fusion, class balance was enhanced and the fusion of the temporal model and the class balance was significantly better than the temporal model alone, while the macro-F1 score of Random Forest was still higher. The results show that the flow-level tree models are still applicable for the majority traffic signatures and temporal fusion is more effective for minority traffic signatures and changing attack behaviours. The high benign false-positive rate also indicates that the sensitivity to attacks has increased, but the separation between legitimate variation and malicious activity has decreased.

XRisk-IDS provides a more comprehensive evidence chain than previous explainable IDS approaches. Ensemble detection has been successfully achieved in the past [1] and formal evaluation of SHAP and LIME has been done with low latency [20] as well as explainability on the edge [23]. The present framework further breaks down these directions into cyber-risk and decision reliability and introduces a mechanism to avoid unstable explanations from causing automatic escalation. It requires calibration and has a low benign recall, but it has a high impact coverage of 99.45%, and a stable risk ranking, so this is a good design.

From a practical perspective, the framework could be applied to prioritize alerts by the degree of evidence of the attack, the importance of the asset, exposure, and the degree of confidence in the explanation of the attack. Furthermore, with

continuous monitoring, the interpretability overhead can be reduced with Selective XAI. The unexpected deterioration at the higher temperatures shows that the calibration methods are not guaranteed to be applicable to fused multiclass detectors. Repeated training, cross-dataset validation, class-sensitive calibration, drift adaptation and controlled deployment trials are suggested for future work. In general, the research provides a stepping stone towards a shift from accuracy-oriented IDS to risk-oriented, human-assisted and transparent cyber defence.

6. Conclusion and Future Scope

This study was aimed at creating an explainable framework for real-time intrusion detection and enterprise cyber-risk prioritization, XRisk-IDS. The experimental results showed that the framework was able to obtain a better class balance than traditional tree-based detection and it significantly outperformed the temporal model. It also had a stable risk ranking, high impact attack coverage (99.45%) and explanation-quality score (0.7172). However, the best macro-F1 was maintained by Random Forest and false positive (benign) and probability calibration problems were not addressed.

The framework should be tested on various datasets, multiple training seeds, real enterprise traffic and Kafka-based deployments in the future. Other concept drift adaptive methods, class sensitive calibration and light weight explanation methods should also be considered, as well as improved benign-attack separation. The framework can be extended and continuously refined, and can also be enriched with continuous feedback from analysts and federated threat intelligence, which can further improve the reliability, privacy and relevance of the operation in a large-scale cybersecurity environment.

Declarations

Acknowledgements
NA

Funding
NA

Contributions
All authors; Conceptualization, All authors; Investigation; All authors; Writing (Original Draft), All authors; Writing (Review and Editing) Supervision, All authors; Project Administration.

Ethics declarations
This article does not contain any studies with human participants or animals performed by any of the authors.

Data Availability Statement
All the datasets used in this study are publicly available from the CSE-CIC-IDS2018 [48] and UNSW-NB15 [49] repositories. The following processed data, feature mappings, model configurations, prediction outputs, explanation scores, risk-assessment results and validation scripts are provided in this study.

Use of Generative Artificial Intelligence
The authors declare that no generative artificial intelligence tools were used in the preparation of this manuscript.

Competing interests
All authors declare no competing interests.

References

[1] Shtayat, M. M., Hasan, M. K., Sulaiman, R., Islam, S., & Khan, A. U. R. (2023). An explainable ensemble deep learning approach for intrusion detection in Industrial Internet of Things. *IEEE Access, 11*, 115047–115061. https://doi.org/10.1109/ACCESS.2023.3323573

[2] Zha, C., Wang, Z., Fan, Y., Bai, B., Zhang, Y., Shi, S., & Zhang, R. (2025). DM-IDS—A network intrusion detection method based on dual-modal fusion. *IEEE Transactions on Network and Service Management*, *22*(4), 3646-3661. https://doi.org/10.1109/TNSM.2025.3565614

[3] Luo, X., Yin, L., Yang, H., Liu, Z., Chen, W., Jia, S., ... & Xiang, H. (2025). SnifferDog: Comprehensively Learning Heterogeneous Features of Network Traffic to Identify Malicious Flows. *IEEE Transactions on Information Forensics and Security*. https://doi.org/10.1109/TIFS.2025.3620640
[4] Zhang, C., Jia, D., Wang, L., Wang, W., Liu, F., & Yang, A. (2022). Comparative research on network intrusion detection methods based on machine learning. *Computers & Security*, *121*, 102861.
[5] Ali, B. (2025). On the fog's frontline: A federated machine learning approach for industrial network threat detection and intrusion prevention. *Journal of Cybersecurity*, *11*(1), Article tyaf041. https://doi.org/10.1093/cybsec/tyaf041
[6] Sayem, I. M., Sayed, M. I., Saha, S., & Haque, A. (2024). ENIDS: A deep learning-based ensemble framework for network intrusion detection systems. *IEEE Transactions on Network and Service Management*, *21*(5), 5809–5825. https://doi.org/10.1109/TNSM.2024.3414305
[7] Molina-Coronado, B., Mori, U., Mendiburu, A., & Miguel-Alonso, J. (2020). Survey of network intrusion detection methods from the perspective of the knowledge discovery in databases process. *IEEE Transactions on Network and Service Management*, *17*(4), 2451-2479.
[8] Ullah, F., Srivastava, G., Mostarda, L., & Raza, U. (2025). ZTID-IoV: Zero-trust intrusion detection in IoV using neurosymbolic AI approach with federated meta-learning. *IEEE Transactions on Consumer Electronics*, *71*(4), 12037–12046. https://doi.org/10.1109/TCE.2025.3625081
[9] Barik, K., Misra, S., & Fernandez-Sanz, L. (2024). Adversarial attack detection framework based on optimized weighted conditional stepwise adversarial network. *International Journal of Information Security*, *23*, 2353–2376. https://doi.org/10.1007/s10207-024-00844-w
[10] Wu, Z., Wang, J., Hu, L., Zhang, Z., & Wu, H. (2020). A network intrusion detection method based on semantic re-encoding and deep learning. *Journal of Network and Computer Applications*, *164*, 102688.
[11] Munna, M. M. I., Rahman, M. M., Frnda, J., Anwar, M. S., & Kutlimuratov, A. (2025). Elevating intrusion detection and security fortification in intelligent networks through cutting-edge machine learning paradigms. *Scientific Reports*, *15*(1), 39989. https://doi.org/10.1038/s41598-025-23754-w
[12] Li, J., Wang, Y., Jia, Y., Zeng, L., Feng, W., Jing, X., ... & Fang, B. (2025). IL-IDS: an incremental learning approach with confined data streams for intrusion detection. *Cybersecurity*, *8*(1), 80. https://doi.org/10.1186/s42400-025-00359-4
[13] Farrukh, Y. A., Wali, S., Khan, I., & Bastian, N. D. (2024). AIS-NIDS: An intelligent and self-sustaining network intrusion detection system. *Computers & Security*, *144*, Article 103982. https://doi.org/10.1016/j.cose.2024.103982
[14] Sánchez-Zas, C., Larriva-Novo, X., Villagrá, V. A., Solera-Cotanilla, S., & Sanz-Rodrigo, M. (2026). Dynamic characterisation of cyberattacks based on the MITRE ATT&CK framework applied to the optimisation of a mitigation selection process. *Future Generation Computer Systems*, *177*, Article 108272. https://doi.org/10.1016/j.future.2025.108272
[15] Polónio, J., Moura, J., & Marinheiro, R. N. (2025). Toward automatic detection and mitigation of high-risk cybersecurity vulnerabilities at networked systems. *IEEE Access*, *13*, 181957–181976. https://doi.org/10.1109/ACCESS.2025.3622497
[16] Alam, K., Kifayat, K., Sampedro, G. A., Karovič, V., & Naeem, T. (2024). SXAD: Shapely explainable AI-based anomaly detection using log data. *IEEE Access*, *12*, 95659–95672. https://doi.org/10.1109/ACCESS.2024.3425472
[17] Long, Z., Yan, H., Shen, G., Zhang, X., He, H., & Cheng, L. (2024). A Transformer-based network intrusion detection approach for cloud security. *Journal of Cloud Computing*, *13*(1), 5.
[18] Dai, W., Li, X., Ji, W., & He, S. (2024). Network intrusion detection method based on CNN-BiLSTM-attention model. *IEEE access*, *12*, 53099-53111.
[19] Schmitt, M. (2023). Securing the digital world: Protecting smart infrastructures and digital industries with artificial intelligence (AI)-enabled malware and intrusion detection. *Journal of Industrial Information Integration*, *36*, Article 100520. https://doi.org/10.1016/j.jii.2023.100520
[20] Arreche, O., Guntur, T. R., Roberts, J. W., & Abdallah, M. (2024). E-XAI: Evaluating black-box explainable AI frameworks for network intrusion detection. *IEEE Access*, *12*, 23954–23988. https://doi.org/10.1109/ACCESS.2024.3365140
[21] Zhou, H., Zou, H., Li, W., Li, D., & Kuang, Y. (2025). HiViT-IDS: an efficient network intrusion detection method based on vision transformer. *Sensors*, *25*(6), 1752.
[22] Hassan, A., Rauf, A., Shafqat, N., Latif, R., & Khan, H. (2025). ZenGuard a machine learning based zero trust framework for context aware threat mitigation using SIEM SOAR and UEBA. *Scientific Reports*, *15*(1), 35871. https://doi.org/10.1038/s41598-025-20998-4
[23] Yang, H., & Wang, F. (2019). Wireless network intrusion detection based on improved convolutional neural network. *Ieee Access*, *7*, 64366-64374.
[24] Liang, W., Li, K. C., Long, J., Kui, X., & Zomaya, A. Y. (2019). An industrial network intrusion detection algorithm based on multifeature data clustering optimization model. *IEEE Transactions on Industrial Informatics*, *16*(3), 2063-2071.

[25] Ghanbarzadeh, R., Hosseinalipour, A., & Ghaffari, A. (2023). A novel network intrusion detection method based on metaheuristic optimisation algorithms: R. Ghanbarzadeh et al. *Journal of ambient intelligence and humanized computing*, *14*(6), 7575-7592.
[26] Ozdem, M. (2025). A novel approach for real-time anomaly detection in dynamic computer networks using temporal graph networks and explainable artificial intelligence. *Alexandria Engineering Journal*, *132*, 369–382. https://doi.org/10.1016/j.aej.2025.11.001
[27] Zhang, Y., Zhang, Y., Zhang, N., & Xiao, M. (2020). A network intrusion detection method based on deep learning with higher accuracy. *Procedia Computer Science*, *174*, 50-54.
[28] Wang, X., Yin, S., Li, H., Wang, J., & Teng, L. (2020). A network intrusion detection method based on deep multi-scale convolutional neural network. *International Journal of Wireless Information Networks*, *27*(4), 503-517.
[29] Javeed, D., Gao, T., Kumar, P., & Jolfaei, A. (2024). An explainable and resilient intrusion detection system for Industry 5.0. *IEEE Transactions on Consumer Electronics*, *70*(1), 1342–1350. https://doi.org/10.1109/TCE.2023.3283704
[30] Mary, D. S., Dhas, L. J. S., Deepa, A. R., Chaurasia, M. A., & Sheela, C. J. J. (2024). Network intrusion detection: An optimized deep learning approach using big data analytics. *Expert Systems with Applications*, *251*, Article 123919. https://doi.org/10.1016/j.eswa.2024.123919
[31] Karn, A. L., Ghanimi, H. M., Iyengar, V., Siddiqui, M. S., Alharbi, M. G., Alroobaea, R., ... & Sengan, S. (2025). Applying the defense model to strengthen information security with artificial intelligence in computer networks of the financial services sector. *Scientific Reports*, *15*(1), 30292. https://doi.org/10.1038/s41598-025-15034-4
[32] Ebrahimi, F., Javidan, R., Akbari, R., & Hosseini, Y. (2025). Intrusion detection in the internet of things using convolutional neural networks: an explainable AI approach. *Cybersecurity*, *8*(1), 66. https://doi.org/10.1186/s42400-025-00369-2
[33] Tserenkhuu, M., Hossain, M. D., Taenaka, Y., & Kadobayashi, Y. (2025). Intrusion detection system framework for SDN-based IoT networks using deep learning approaches with XAI-based feature selection techniques and domain-constrained features. *IEEE Access*, *13*, 136864–136880. https://doi.org/10.1109/ACCESS.2025.3595595
[34] Hore, S., Ghadermazi, J., Shah, A., & Bastian, N. D. (2024). A sequential deep learning framework for a robust and resilient network intrusion detection system. *Computers & Security*, *144*, Article 103928. https://doi.org/10.1016/j.cose.2024.103928
[35] Lee, H.-W., Han, T.-H., & Lee, T.-J. (2023). Reference-based AI decision support for cybersecurity. *IEEE Access*, *11*, 143324–143339. https://doi.org/10.1109/ACCESS.2023.3342868
[36] Arya, K., Siddhant, S., & Upadhyay, L. (2025). An explainable hybrid deep learning framework for network intrusion detection using feature-guided CNN models. *IEEE Access*, *13*, 204954–204977. https://doi.org/10.1109/ACCESS.2025.3637857
[37] Ghadermazi, J., Shah, A., & Bastian, N. D. (2025). Towards real-time network intrusion detection with image-based sequential packets representation. *IEEE Transactions on Big Data*, *11*(1), 157–173. https://doi.org/10.1109/TBDATA.2024.3403394
[38] Volpe, G., Fiore, M., La Grasta, A., Albano, F., Stefanizzi, S., Mongiello, M., & Mangini, A. M. (2024). A Petri net and LSTM hybrid approach for intrusion detection systems in enterprise networks. *Sensors*, *24*, Article 7924. https://doi.org/10.3390/s24247924
[39] Alamro, H., Alahmari, S., Nemri, N., Aljebreen, M., Alhashmi, A. A., Alamro, S., ... & Al Duhayyim, M. (2025). Enhanced intrusion detection in cybersecurity through dimensionality reduction and explainable artificial intelligence. *Scientific Reports*, *15*(1), 33848. https://doi.org/10.1038/s41598-025-06761-9
[40] Sivamohan, S., & Sridhar, S. S. (2023). An optimized model for network intrusion detection systems in Industry 4.0 using XAI-based Bi-LSTM framework. *Neural Computing and Applications*, *35*, 11459–11475. https://doi.org/10.1007/s00521-023-08319-0
[41] Yang, L., Rajab, M. E., Shami, A., & Muhaidat, S. (2024). Enabling AutoML for zero-touch network security: Use-case driven analysis. *IEEE Transactions on Network and Service Management*, *21*(3), 3555–3582. https://doi.org/10.1109/TNSM.2024.3376631
[42] He, K., Kim, D. D., & Asghar, M. R. (2025). MTD-AD: Moving target defense as adversarial defense. *IEEE Transactions on Dependable and Secure Computing*, *22*(5), 5047–5059. https://doi.org/10.1109/TDSC.2025.3560246
[43] Kim, Y., Kim, J., & Kim, D. (2024). Hi-MLIC: Hierarchical multilayer lightweight intrusion classification for various intrusion scenarios. *IEEE Access*, *12*, 120098–120115. https://doi.org/10.1109/ACCESS.2024.3450671
[44] Gong, S., Cho, J., & Choi, K. K. (2025). Detection of multi-stage attacks through attack pattern segmentation. *IEEE Access*, *13*, 204155–204167. https://doi.org/10.1109/ACCESS.2025.3635053
[45] Janati Idrissi, M., Alami, H., El Mahdaouy, A., El Mekki, A., Oualil, S., Yartaoui, Z., & Berrada, I. (2023). Fed-ANIDS: Federated learning for anomaly-based network intrusion detection systems. *Expert Systems with Applications*, *234*, Article 121000. https://doi.org/10.1016/j.eswa.2023.121000

[46] Kim, H., Lee, J., & Park, J.-G. (2024). SITRAN: Self-supervised IDS with transferable techniques for 5G industrial environments. *IEEE Internet of Things Journal, 11*(21), 35465–35476. https://doi.org/10.1109/JIOT.2024.3437448
[47] Zhou, Y., Mazzuchi, T. A., & Sarkani, S. (2020). M-AdaBoost-A based ensemble system for network intrusion detection. *Expert Systems with Applications, 162*, 113864.
Wardana, A. A., Kołaczek, G., Warzyński, A., Sukarno, P., & Adiwijaya. (2026). SNI-CIDS: Collaborative intrusion detection using modified ensemble stacking deep neural networks for new network integration in heterogeneous networks. *Future Generation Computer Systems, 177*, Article 108252. https://doi.org/10.1016/j.future.2025.108252
[48] Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)* (pp. 108–116). https://doi.org/10.5220/0006639801080116
[49] Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems. In *Proceedings of the Military Communications and Information Systems Conference (MilCIS)* (pp. 1–6). https://doi.org/10.1109/MilCIS.2015.7348942