



ARTICLE

Temporal Explainable Machine Learning for Insider Threat Detection in Intelligent Transportation Systems

Md. Nahian Suhaimee¹, Farhan Shakil², Md. Rifat Al Amin Khan³, Md. Omar Faruq⁴,
Md. Jakir Hossen^{5,*} and M. F. Mridha⁶

¹St. Louis Community College—Forest Park, 5600 Oakland Ave, St. Louis, MO, USA

²George Herbert Walker School of Business & Technology, Webster University, 470 East Lockwood Avenue, Saint Louis, MO, USA

³Electrical and Computer Engineering Department, Lamar University, 4400 MLK Blvd, Beaumont, TX, USA

⁴Department of Computer and Information Sciences, Webster University, 470 E Lockwood Ave, Webster Groves, MO, USA

⁵Center for Advanced Analytics (CAA), COE for Artificial Intelligence, Faculty of Engineering & Technology (FET), Multimedia University, Melaka, Malaysia

⁶Department of Computer Science, American International University—Bangladesh (AIUB), Dhaka, Bangladesh

*Corresponding Author: Md. Jakir Hossen. Email: jakir.hossen@mmu.edu.my

Received: 09 April 2026; Accepted: 02 July 2026; Published: 22 October 2026

ABSTRACT: Insider threats in Intelligent Transportation Systems (ITS) pose significant risks to operational safety and service continuity, as malicious actions often originate from users with legitimate access and evade traditional signature-based detection methods. This study proposes a temporal and explainable machine learning framework that models activity as sequential patterns and provides interpretable insights for each detection decision. The proposed approach integrates recurrent neural networks and attention-based encoders to capture short- and long-term temporal dependencies. To enhance interpretability, a hybrid explanation module combines temporal attention, SHapley Additive exPlanations (SHAP), and counterfactual analysis to identify influential time steps, key risk factors, and actionable changes. Experiments are conducted using a two-stage design: insider-threat modeling on a public insider behavior dataset and ITS-domain transferability evaluation on CICIoV2024 CAN-bus data. Across five folds on the insider dataset, the proposed model achieves precision of 0.89 ± 0.010 , recall of 0.87 ± 0.012 , F1-score of 0.88 ± 0.011 , accuracy of 0.90 ± 0.010 , ROC-AUC of 0.94 ± 0.009 , and PR-AUC of 0.91 ± 0.010 . It also demonstrates robustness under class imbalance (MCC = 0.69), reduces average time-to-detect from 5.4 days to 3.7 days, and improves probability reliability with a Brier score of 0.123. The CICIoV2024 evaluation further shows that the same temporal explainable architecture can process ITS-native IoV packet sequences and distinguish benign, DoS, and spoofing behavior. The generated explanations exhibit high fidelity, stability, and sparsity, enabling efficient analyst interpretation and decision support; CICIoV2024 is used for transferability testing rather than direct insider-threat validation.

KEYWORDS: Intelligent transportation systems; insider threat detection; explainable AI; temporal modeling; CAN-bus security; anomaly detection

1 Introduction

Modern transportation depends on interconnected digital systems. Control centers set signal plans, roadside units exchange messages, and maintenance teams update software on live networks [1]. These actions leave rich trails in logs. The same connectivity that enables efficiency also enlarges the attack surface. External intrusions draw attention, yet insiders with valid access can cause greater harm [2]. Detecting

such threats requires models that understand behavior over time and explain their decisions in terms that operators can use [3].

Existing insider-threat detection methods often rely on static feature snapshots, fixed-window summaries, or signature-based rules. These methods can miss attacks whose risk emerges only through a sequence of small actions, such as repeated authentication failures, off-hours access, and gradual privilege misuse. Deep sequence models can capture such temporal patterns, but many prior approaches still provide limited explanation of which time periods and behavioral variables caused an alert [4]. In ITS environments, this gap is critical because analysts must justify security actions that may affect traffic control, maintenance workflows, and public safety. The research problem addressed in this study is therefore to detect insider threats from temporally ordered user activity while producing faithful, concise, and actionable explanations for each prediction.

This work aims to develop an insider threat detector that learns evolving behavior and explains why it flags a particular case. The motivation is practical: security teams in ITS must triage alerts under tight time and staffing constraints. Explanations should pinpoint the specific periods and variables that drove risk, allowing teams to verify alerts, respond effectively, and refine policy.

An effective solution reduces dwell time, cuts false alarms, and turns model output into concrete actions. Temporal context helps distinguish routine after-hours maintenance from risky escalation. Transparent reasoning supports audit and governance [5]. Calibrated scores enable threshold setting by role and asset criticality. Together these properties move detection closer to safe deployment in live transportation networks.

We model activity as temporal sequences and learn evolving patterns with recurrent and attention-based encoders. An attention mechanism highlights influential time steps, while a hybrid explanation stack combines attention visualization, feature attribution, and counterfactual analysis. We use a two-stage evaluation: insider-threat modeling on a public user-activity dataset and ITS-domain transferability testing on CICIoV2024 CAN-bus data. We train with imbalance-aware objectives and evaluate with discrimination, imbalance-aware, temporal, calibration, and explanation-quality metrics. We compare the proposed framework against recurrent baselines, hybrid deep sequence models, Transformer-based architectures, and sequence-oriented anomaly detection methods. Key contributions are as follows:

- A temporal insider-threat detection framework for ITS-oriented security that models user activity as ordered behavioral sequences rather than isolated feature snapshots.
- A hybrid explainability design that integrates temporal attention, SHAP feature attribution, and counterfactual analysis to explain when, why, and how a threat prediction arises.
- A two-stage evaluation protocol that combines insider-threat modeling with CICIoV2024-based ITS-domain transferability testing on CAN-bus packet sequences.
- Empirical evidence that the proposed temporal and explainable design improves detection quality, time-to-detect, probability reliability, and analyst-oriented explanation quality compared with sequence-oriented and deep anomaly detection baselines.

This paper is organized as follows. [Section 1](#) introduces the problem context, research motivation, and contributions. [Section 2](#) reviews related studies on insider threat detection, ITS security, and explainable modeling. [Section 3](#) presents the proposed methodology, including two-stage preprocessing, temporal modeling, explainability, training, and evaluation metrics. [Section 4](#) reports the experimental results and comparative analysis. [Section 5](#) discusses the findings, operational implications, limitations, and future directions. [Section 6](#) concludes the paper.

2 Related Work

Research on insider threat detection spans behavioral analytics, intrusion detection, and organizational security modeling [2,6]. Early approaches used rule-based checks, thresholding, and signature-based detection over access logs, authentication records, and file operations. Although these methods were simple to deploy, they often failed to detect unseen insider tactics and produced high false alarm rates when user behavior varied across roles, shifts, or operational contexts. Anomaly detection improved coverage by modeling normal behavior and flagging deviations, but many classical methods lacked temporal awareness and produced alerts that were difficult for analysts to interpret.

Recent studies have moved toward machine learning, explainable AI, and privacy-preserving detection for cyber-physical and transportation systems. Patel et al. [7] proposed an ML- and XAI-based framework for real-time threat detection in the Internet of Vehicles (IoV), showing the value of interpretable detection in vehicular cybersecurity. Ullah et al. [8] introduced an ensemble-based hybrid model for attack detection in the Internet of Vehicular Things (IoVT), improving robustness across vehicular network attack scenarios. Taheri et al. [9] developed an explainable federated intrusion detection framework for connected vehicles by combining distributed learning with SHAP-based interpretation. Gurushanker et al. [10] proposed FeXAI, a federated and explainable AI framework for IoT-enabled smart transportation systems, including IoV, EV charging networks, and VANETs. Saiteja and Sudha Rani [11] presented an explainable BiGRU-based model for cyber-attack detection in IoV and intelligent connected vehicle networks. These works confirm the growing importance of explainable, privacy-aware, and sequence-oriented security models in transportation-related environments. CICIoV2024 further provides ITS-native CAN-bus records for benign, DoS, and spoofing behavior, making it useful for testing transferability to IoV packet streams; however, it does not contain user-level insider labels. Thus, it complements insider-behavior datasets rather than replacing them.

Deep learning has improved temporal threat detection by capturing sequential dependencies in behavioral logs. Recurrent models such as LSTM and GRU can learn gradual escalation patterns, including repeated failed logins, off-hours access, unusual privilege use, and data exfiltration attempts. Khan et al. [12] proposed a CNN-GRU model for IoT-based transportation security with SHAP-based explainability, showing that high detection performance can be paired with interpretable outputs in ITS contexts. Ye et al. [13] integrated federated learning with behavior log analysis for insider threat detection by transforming numerical data into image-like inputs for CNN-based modeling. Kotb et al. [14] introduced the Deep Synthesis Insider Intrusion Detection model, which uses automated feature synthesis and deep learning to identify both traditional and AI-generated insider threats. In parallel, Yilmaz and Can [15] reviewed AI-based insider threat detection methods, including user behavior analytics, NLP, LLMs, and graph-based approaches, and highlighted open challenges such as scarce datasets, privacy constraints, and the need for multimodal learning.

Explainability has become central to insider threat detection because analysts need alerts that are not only accurate but also accountable and actionable. Al and Sagioglu [16] reviewed XAI methods for intrusion detection systems and emphasized the role of transparency in improving trust and reliability. Feature attribution methods provide local and global importance scores, while attention mechanisms identify influential time steps in sequential models [17]. Counterfactual explanations add another layer of interpretability by showing minimal changes that would alter a model prediction, supporting recourse and policy refinement [18]. Recent work also stresses that explanations should be faithful, stable, sparse, and easy for analysts to use during investigation.

Evaluation practices have also evolved beyond accuracy-based reporting. For imbalanced security data, metrics such as precision, recall, F1-score, MCC, balanced accuracy, and G-mean provide a more reliable view of detection quality. Temporal measures, including time-to-detect, early capture rate, and missed detection rate, are important because insider threats often unfold over time [3]. Calibration metrics such as Brier score and expected calibration error further support risk-aware alert thresholding. Runtime and memory requirements also matter for near real-time monitoring and edge deployment in operational networks [19]. Recent studies have also explored data augmentation, cost-sensitive learning, active learning, and semi-supervised learning to address data scarcity and severe class imbalance.

Within Intelligent Transportation Systems, prior work has mostly addressed external attacks, sensor anomalies, and network-level intrusions across control centers, roadside units, and vehicle-to-everything components. Less attention has been given to insiders who hold valid credentials and may abuse access over time. Studies on human-in-the-loop analytics show the need for dashboards, ranked alerts, timelines, and explanation artefacts that support analyst review [20]. Other recent work highlights the importance of operational technology security, policy-aware monitoring, and domain-specific threat analysis in transportation environments [21]. Overall, the literature shows a clear need for insider threat detection methods that combine temporal modeling, explainability, calibration, and operational relevance.

Against this background, the present work addresses a specific gap: temporally explainable insider threat detection with ITS-domain transferability analysis. Unlike static or purely network-level approaches, the proposed framework models user activity as ordered behavioral sequences and combines temporal attention, SHAP-based feature attribution, and constrained counterfactual explanations. It further tests the same temporal explainable architecture on CICIoV2024 CAN-bus packet sequences to assess transferability to ITS-native communication data. Detection performance is evaluated using classification, discrimination, imbalance-aware, time-to-detect, calibration, and explanation-quality metrics, supporting accurate detection with transparent and actionable evidence for analysts.

3 Methodology

In this section, we present the proposed framework for explainable insider threat detection in Intelligent Transportation Systems (ITS). The framework follows a two-stage validation design: Stage I models insider behavior using the Insider Threat Dataset for Classified Environments, while Stage II assesses ITS-domain transferability using CICIoV2024 CAN-bus data. The framework consists of four major components: (i) temporal modeling of activity sequences, (ii) hybrid explainability, (iii) model training and optimization, and (iv) evaluation metrics. The preprocessing pipeline described below structures the input data for deep temporal learning.

3.1 Data Preprocessing

The Insider Threat Dataset for Classified Environments contains daily activity logs with both categorical and numerical attributes describing user behavior in a secure organizational context. To evaluate transferability to ITS-native vehicle communication logs, we further use CICIoV2024 as a second-stage dataset containing decimal CAN-bus records from benign, DoS, and spoofing scenarios. Prior to model training, both datasets were processed to ensure data quality, consistency, and suitability for temporal deep learning models.

3.1.1 Handling Missing Values

Missing entries were imputed using a hybrid approach:

$$x_i^* = \begin{cases} x_i, & \text{if } x_i \neq \text{NaN} \\ \mu_c, & \text{if } x_i \in \mathbb{R} \\ \arg \max_{v \in V_c} P(v), & \text{if } x_i \in \mathbb{C} \end{cases} \quad (1)$$

where x_i is the original feature value, μ_c is the mean of the feature within class c (for numeric features), and $P(v)$ is the empirical probability of category v for categorical features $\mathbb{C}$.

3.1.2 Normalization of Numerical Features

All numerical features, including user-activity variables in Stage I and CAN-bus fields in Stage II, were normalized to zero mean and unit variance:

$$\hat{x}_i = \frac{x_i - \mu}{\sigma} \quad (2)$$

where μ and σ are the feature-wise mean and standard deviation, respectively. This ensures stable gradient propagation in deep networks.

3.1.3 Encoding of Categorical Variables

Categorical variables such as *department*, *clearance level*, and *browser type* were transformed into dense embeddings using trainable embedding layers:

$$\mathbf{e}_i = \mathbf{W}_{\text{emb}} \cdot \mathbf{l}_i \quad (3)$$

where $\mathbf{l}_i$ is the one-hot representation of category i and $\mathbf{W}_{\text{emb}} \in \mathbb{R}^{|\mathbb{C}| \times d}$ is the embedding matrix with dimension d . These embeddings capture semantic relationships between categories and improve generalization. For CICIoV2024, the decimal CAN-bus fields were already numerical; therefore, no categorical embedding was applied to packet features.

3.1.4 Temporal Batching

Since insider behavior evolves over time, we constructed sequences of daily activities per user:

$$\mathcal{S}_u = \{\mathbf{x}_{u,1}, \mathbf{x}_{u,2}, \dots, \mathbf{x}_{u,T}\} \quad (4)$$

where $\mathbf{x}_{u,t}$ denotes the feature vector of user u on day t , and T is the temporal horizon. Sliding windows of length L were generated:

$$\mathcal{W}_{u,k} = \{\mathbf{x}_{u,k}, \mathbf{x}_{u,k+1}, \dots, \mathbf{x}_{u,k+L-1}\} \quad (5)$$

to enable the model to learn temporal dependencies in user behavior. For CICIoV2024, each packet was represented as $\mathbf{z}_t = [ID_t, DATA_{0,t}, \dots, DATA_{7,t}]$, and packet streams were divided into sliding windows $\mathcal{V}_k = \{\mathbf{z}_k, \dots, \mathbf{z}_{k+L-1}\}$. This allows the same temporal encoder to process ITS-native CAN-bus sequences for transferability evaluation.

3.1.5 Label Alignment

Each sequence $\mathcal{W}_{u,k}$ was assigned a binary label $y \in \{0,1\}$ corresponding to whether the user was identified as a malicious insider during the sequence window:

$$y_{u,k} = \begin{cases} 1, & \text{if malicious activity detected in } \mathcal{W}_{u,k} \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

For CICIoV2024, `decimal_benign.csv` was assigned label 0, while `decimal_DoS.csv` and all spoofing files were assigned label 1. A CAN-bus window was labeled malicious if it contained at least one attack packet. For multiclass analysis, attack windows were separated into DoS, gas spoofing, RPM spoofing, speed spoofing, and steering-wheel spoofing. CICIoV2024 is used only for ITS-domain transferability, not as an insider-threat dataset.

3.1.6 Dataset Splitting

The dataset was divided into training, validation, and testing subsets while preserving temporal order to avoid data leakage:

$$\mathcal{D}_{\text{train}} = \{\mathcal{W}_{u,k} \mid t \leq T_{\text{train}}\} \quad (7)$$

$$\mathcal{D}_{\text{val}} = \{\mathcal{W}_{u,k} \mid T_{\text{train}} < t \leq T_{\text{val}}\} \quad (8)$$

$$\mathcal{D}_{\text{test}} = \{\mathcal{W}_{u,k} \mid t > T_{\text{val}}\} \quad (9)$$

where $T_{\text{train}} < T_{\text{val}} < T$ denote temporal cutoffs. This ensures that the model is evaluated on unseen future behaviors. For Stage II, CICIoV2024 packet windows were also split chronologically to preserve packet order and avoid leakage from future traffic patterns.

While temporal splitting prevents information leakage across time, it does not fully eliminate the possibility of user-specific patterns being shared between training and testing sets. In particular, the same user may appear in both sets at different time intervals, which can introduce implicit memorization of user-specific behavioral signatures. A stricter user-disjoint evaluation requires that users in the test set are absent from the training data, ensuring that the model is assessed on generalized insider threat patterns rather than identity-specific fingerprints. This distinction is important in operational ITS environments, where models must detect previously unseen malicious actors based on behavioral evidence rather than memorized user profiles. Therefore, we report an additional user-disjoint robustness evaluation in [Section 4.2](#).

For robustness analysis, we also used a blocked five-fold cross-validation protocol. The temporal sequence windows were first sorted chronologically within each user. The folds were then constructed to preserve temporal ordering and maintain approximately similar malicious/benign class ratios. To reduce leakage from overlapping sliding windows, adjacent windows crossing fold boundaries were excluded using a gap of $L - 1$ time steps, where L is the sequence length. In each fold, models were trained using the same hyperparameters, weighted loss, optimizer, and early stopping settings. The final reported results are expressed as mean $\pm$ standard deviation across the five folds.

3.1.7 Class Imbalance Handling

Insider threat and IoV attack datasets are typically imbalanced, with far fewer malicious cases than benign. To address this, we applied class-weighted loss:

$$\mathcal{L} = - \sum_{i=1}^N w_{y_i} \cdot \left[y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \right] \quad (10)$$

where $w_{y_i} = \frac{N}{2N_{y_i}}$, N is the total number of samples, and N_{y_i} is the number of samples of class y_i . This weighting prevents bias toward the majority class.

The choice of a class-weighted loss function is motivated by the highly imbalanced nature of insider threat datasets, where malicious instances are rare and often embedded within normal behavioral sequences. Weighted loss functions increase the contribution of minority-class samples during optimization without altering the underlying data distribution, allowing the model to learn subtle patterns associated with insider activity while preserving temporal consistency. In contrast, data-level approaches such as random oversampling, undersampling, or synthetic sample generation (e.g., SMOTE) may introduce duplicated or artificially generated sequences that do not accurately reflect real user behavior, potentially distorting temporal dependencies and leading to overfitting. This limitation is particularly critical in sequential models, where the order and continuity of events carry important contextual information. By operating directly on the original data, the weighted loss approach maintains sequence integrity while improving sensitivity to rare malicious events, making it well-suited for temporal insider threat detection tasks.

3.1.8 Final Dataset

After preprocessing, the datasets contained:

- Normalized numerical features
- Learned embeddings for Stage I categorical variables
- Temporal sequences of daily user activity
- CICIoV2024 CAN-bus packet windows using ID and $DATA_0$ – $DATA_7$
- Class imbalance addressed through weighted loss

This structured representation was then used for training temporal deep learning models such as LSTMs and Transformers.

3.2 Proposed Methodology

3.2.1 Temporal Modeling of Insider Behavior

We represent each user u by a sequence of daily activity vectors:

$$\mathcal{S}_u = \{\mathbf{x}_{u,1}, \mathbf{x}_{u,2}, \dots, \mathbf{x}_{u,T}\}, \quad (11)$$

where $\mathbf{x}_{u,t} \in \mathbb{R}^d$ is the d -dimensional feature vector of user u at day t , and T is the total number of days observed. For Stage II, CICIoV2024 CAN-bus packet windows are processed using the same temporal encoder after mapping packet vectors $\mathbf{z}_t = [ID_t, DATA_{0,t}, \dots, DATA_{7,t}]$ into the model input space.

To capture temporal dependencies, we employ recurrent and attention-based models, which enable the detector to focus on informative periods and long-range patterns.

Although the Stage I insider-threat dataset used in this study is not collected directly from an ITS deployment, the proposed framework is designed to generalize to heterogeneous ITS log environments.

In Intelligent Transportation Systems, data sources include roadside units (RSUs), traffic control centers, and vehicle-to-everything (V2X) communication logs, which vary in structure, timing, and reliability. The preprocessing and modeling pipeline transforms heterogeneous categorical and numerical attributes into a unified temporal sequence representation, enabling consistent analysis across diverse ITS components. The temporal modeling approach supports irregular event intervals, which is essential for handling intermittent connectivity in V2X systems. In addition, the framework allows incremental inference, making it suitable for low-latency deployment scenarios such as RSUs and edge-based monitoring nodes. While protocol-specific parsing (e.g., NTCIP message structures) is not explicitly modeled in the Stage I dataset, the architecture can incorporate such structured inputs through feature encoding, enabling adaptation to real ITS communication logs.

Recurrent neural networks (RNNs), particularly LSTM variants, capture temporal dynamics by maintaining a hidden state that evolves over time and encodes prior behavioral context. Through gating mechanisms, LSTMs selectively retain or discard historical information, allowing the model to represent both short-term fluctuations (e.g., sudden login spikes) and long-term dependencies (e.g., gradual escalation patterns across days). This design is effective for modeling sequential insider behavior where risk emerges cumulatively rather than instantaneously.

Attention mechanisms complement recurrent modeling by assigning adaptive importance weights to different time steps in a sequence. Instead of compressing all historical information into a single hidden state, attention explicitly learns which temporal segments contribute most to the prediction. This improves both performance and interpretability, as the model can highlight critical periods such as abnormal access times or bursts of suspicious activity.

Compared with alternative sequence modeling approaches, temporal convolutional networks (TCNs) use fixed receptive fields and hierarchical filters, which can efficiently capture local temporal patterns but may require deeper architectures to model long-range dependencies. Transformer-based models rely entirely on self-attention to capture global relationships without recurrence, enabling parallel computation and flexible dependency modeling. However, they often require larger datasets and higher computational cost. In contrast, the proposed approach leverages LSTM-based sequential encoding combined with attention to balance modeling capacity, computational efficiency, and interpretability for insider threat detection scenarios with structured temporal logs.

Long Short-Term Memory (LSTM)

The Long Short-Term Memory (LSTM) network processes sequential inputs using gating mechanisms:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (12)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (13)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (14)$$

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (15)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (16)$$

$$h_t = o_t \odot \tanh(c_t), \quad (17)$$

where f_t , i_t , o_t are the forget, input, and output gates, c_t is the memory cell, and h_t is the hidden state.

Transformer Encoder

For comparison, we also experimented with Transformer encoders, which rely on self-attention:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (18)$$

where Q, K, V are the query, key, and value matrices derived from the input. Multi-head attention captures relationships across multiple subspaces:

$$\text{MHA}(X) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O. \quad (19)$$

3.2.2 Architectural Details

The proposed framework leverages a temporal deep learning backbone for modeling activity sequences. We designed two architectures: an LSTM-based model and a Transformer-based encoder. Both models were carefully tuned to balance predictive accuracy and computational efficiency. Table 1 summarizes the layer-wise configuration of the LSTM-based model, which served as our primary architecture, and Fig. 1 provides a visual overview of these components.

Table 1: Architectural details of the LSTM-based temporal threat detection model.

Layer	Type	Units/Heads	Activation	Output Shape
Input	Feature Sequence	$d = 64$ (features)	–	$(B, T, 64)$
Embedding/Projection	Dense Mapping	$d_{emb} = 32$ per category or CAN projection	–	$(B, T, 96)$
LSTM-1	LSTM	128 hidden units	tanh	$(B, T, 128)$
LSTM-2	LSTM	64 hidden units	tanh	$(B, 64)$
Dropout	Regularization	$p = 0.3$	–	$(B, 64)$
Dense-1	Fully Connected	32 neurons	ReLU	$(B, 32)$
Attention Layer	Additive Attention	Context vector $d = 32$	Softmax	$(B, 32)$
Dense-2	Fully Connected	16 neurons	ReLU	$(B, 16)$
Output	Dense	1 neuron	Sigmoid	$(B, 1)$

Here, B denotes the batch size and T is the temporal sequence length. For Stage I, the embedding layer reduces categorical sparsity, while stacked LSTMs capture temporal dynamics of insider activity. For Stage II, the nine-dimensional CICIoV2024 CAN-bus packet vector is mapped through a dense projection layer before entering the same temporal encoder. Dropout prevents overfitting, and the attention layer highlights critical time steps. The final sigmoid output provides a threat probability. Fig. 1 depicts the same flow from input features and embeddings/projections through temporal encoders, attention, and dense layers to the binary output.

For completeness, the Transformer-based variant replaces the LSTM layers with two Transformer encoder blocks:

- Multi-head self-attention with $h = 4$ heads, each of dimension $d_k = 32$.
- Position-wise feed-forward layers of dimension 128.
- Residual connections and layer normalization.

This variant achieved comparable performance and interpretable temporal salience via attention weights.

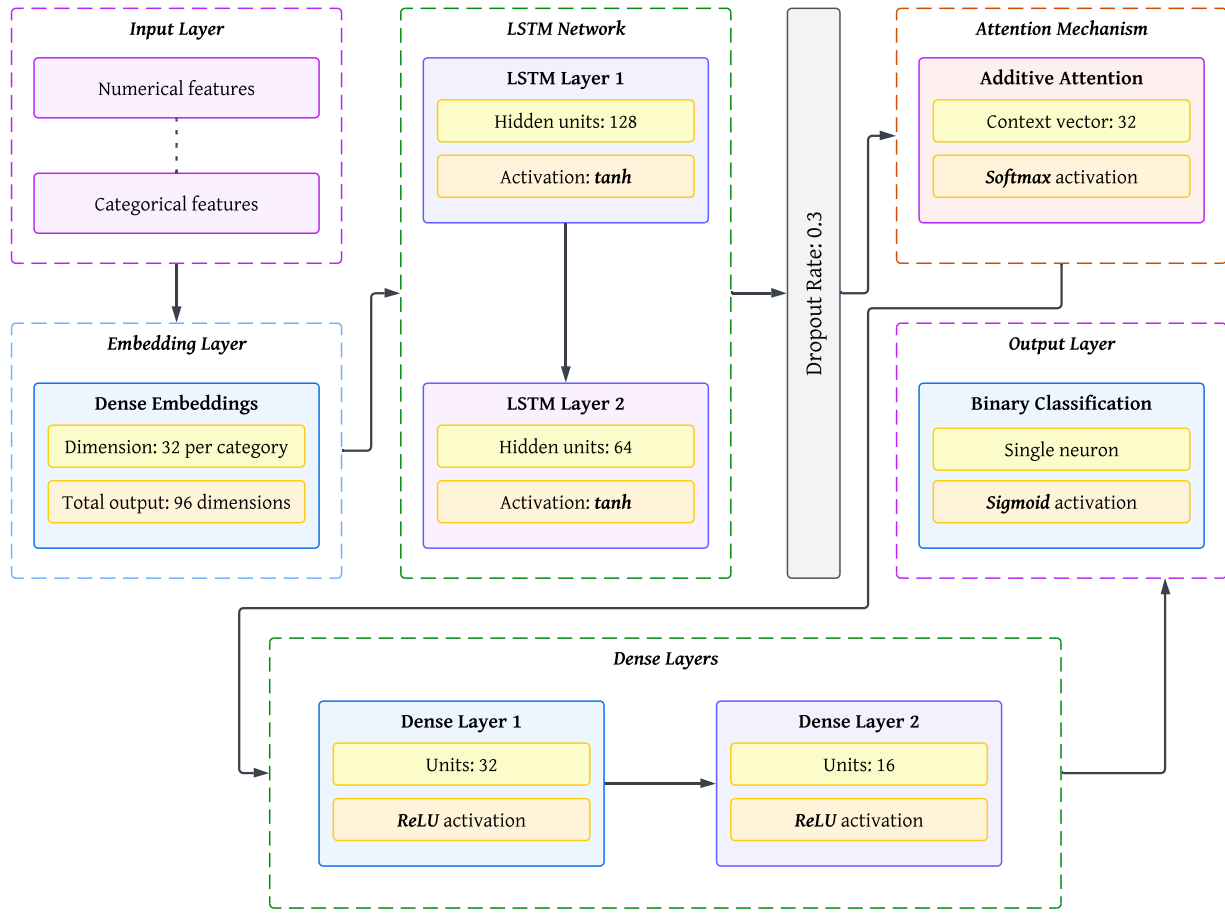


Figure 1: Layer-wise architecture of the proposed detector: dense embeddings or CAN-bus projection, stacked LSTMs, dropout, additive attention, fully connected layers, and a sigmoid output for binary classification.

3.2.3 Hybrid Explainability Framework

A major contribution of this work is the integration of multiple explainability techniques to provide actionable insights.

Feature Attribution with SHAP

We employ SHapley Additive exPlanations (SHAP) to quantify the contribution of each feature to the model output:

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} [f(S \cup \{i\}) - f(S)], \quad (20)$$

where F is the set of features and $f(\cdot)$ is the model prediction function.

Temporal Attention Visualization

In both LSTM and Transformer models, attention weights α_t highlight which time steps were most influential:

$$\alpha_t = \frac{\exp(e_t)}{\sum_{k=1}^T \exp(e_k)}, \quad e_t = v^T \tanh(Wh_t + b). \quad (21)$$

These weights provide a temporal explanation, revealing when suspicious behavior escalates in Stage I or when malicious packet behavior appears in Stage II.

However, attention weights alone do not guarantee faithful explanations. They primarily indicate which time steps the model focuses on rather than providing causal attribution for the prediction. For this reason, the proposed framework does not rely solely on attention for interpretability. Instead, attention-based temporal highlighting is complemented by SHAP-based feature attribution and assessed using explanation-quality metrics such as fidelity, comprehensiveness, and sufficiency, as reported in [Section 4](#). These measures evaluate how explanation components relate to the model's decision behavior by examining the effect of removing or retaining important features. This combined approach treats temporal attention as an informative localization mechanism while supporting more faithful and actionable interpretation through complementary attribution analysis.

While temporal attention highlights when critical behavioral changes occur, SHAP provides a complementary view of why the model assigns a high-risk prediction. Attention weights identify the most influential time steps in a sequence, but they do not quantify the contribution of individual features within those steps. In contrast, SHAP decomposes the model output into feature-level contributions, enabling fine-grained attribution across behavioral variables. The combination enables a two-level explanation: attention localizes the time intervals where anomalous behavior emerges, and SHAP identifies the key features within those intervals (e.g., failed logins, off-hours access, or unusual network activity) that drive the prediction. By linking when an anomaly occurs with what behavioral factors triggered it, the framework provides structured and actionable explanations that support targeted investigation and intervention.

Counterfactual Explanations

To generate actionable feedback, we construct counterfactuals:

$$\tilde{\mathbf{x}} = \mathbf{x} + \delta, \quad f(\tilde{\mathbf{x}}) \neq f(\mathbf{x}), \quad (22)$$

where δ is the minimal perturbation needed to change the model prediction. In Stage I, this may correspond to changing behavioral indicators such as off-hours access. In Stage II, perturbations are constrained to valid CAN-bus feature ranges.

Counterfactual explanations are generated under feature-space constraints to ensure validity and actionability in cybersecurity settings. While [Eq. \(22\)](#) presents a general additive formulation, the perturbation δ is restricted based on feature type. For continuous features, δ is bounded within observed ranges to maintain realistic values. For discrete features (e.g., number of login attempts), perturbations are constrained to integer increments or decrements. For categorical variables (e.g., role, clearance level), counterfactuals are generated by selecting valid alternative categories from the predefined domain rather than applying arithmetic changes. These constraints help keep generated counterfactuals within feasible feature domains and support physically plausible and operationally meaningful scenarios. For example, instead of producing fractional login counts, the method identifies feasible behavioral adjustments such as reducing repeated login

attempts or modifying access patterns. This constrained formulation enables actionable explanations that can be interpreted and applied by ITS security analysts in decision-making contexts.

3.3 Training and Implementation Details

This subsection outlines the training strategies and implementation choices adopted to optimize the proposed two-stage threat detection framework. We describe the loss function, optimization algorithm, regularization methods, and evaluation metrics used for Stage I insider-threat modeling and Stage II CICIoV2024 transferability evaluation.

3.3.1 Loss Function

Given the inherent class imbalance in insider threat and IoV attack detection, we adopt a weighted binary cross-entropy loss:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^N w_{y_i} [y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)], \quad (23)$$

where $y_i \in \{0, 1\}$ is the ground truth label, $\hat{y}_i$ is the predicted probability, and $w_{y_i} = \frac{N}{2N_{y_i}}$ is the class weight computed from the number of samples per class. For the multiclass CICIoV2024 attack-type analysis, the same class-weighting strategy was applied with categorical cross-entropy.

3.3.2 Optimization Strategy

We employ the Adam optimizer for efficient gradient-based optimization. The parameter updates follow:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t, \quad (24)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2, \quad (25)$$

$$\theta_{t+1} = \theta_t - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t + \varepsilon}}, \quad (26)$$

where g_t is the gradient at iteration t , m_t and v_t are the first- and second-moment estimates, η is the learning rate, and ε is a small constant to prevent division by zero.

3.3.3 Regularization Techniques

To mitigate overfitting, we apply two regularization techniques:

- **Dropout:** A dropout rate $p = 0.3$ is applied after hidden layers, randomly deactivating neurons to improve generalization.
- **Early Stopping:** Training is halted if validation loss does not improve for $K = 10$ consecutive epochs, preventing overtraining.

3.3.4 Evaluation Metrics

To evaluate the performance of the proposed framework, we consider multiple metrics tailored for imbalanced temporal classification:

- Precision, Recall, and F1-score for the malicious insider or attack class.
- Area Under the Receiver Operating Characteristic Curve (ROC-AUC).

- Area Under the Precision-Recall Curve (PR-AUC).
- Matthews Correlation Coefficient (MCC).
- Time-to-Detect (TTD), measured in days for Stage I and packet windows for Stage II.
- Calibration metrics such as the Brier score.
- Macro-F1 and per-class recall for CICIoV2024 multiclass attack-type analysis.

These metrics collectively capture classification accuracy, robustness to imbalance, temporal effectiveness, probabilistic calibration, and interpretability within the ITS-oriented two-stage threat detection context.

3.4 Algorithmic Summary of the Proposed Architecture

The overall workflow of the proposed two-stage temporal explainable framework is summarized in Algorithm 1. It highlights the sequence of operations from dataset-specific preprocessing to explainable threat classification.

Algorithm 1: Temporal explainable threat detection framework

Require: Preprocessed temporal sequence $\mathcal{X} = \{\mathbf{a}_1, \dots, \mathbf{a}_T\}$ from Stage I user activity logs or Stage II CICIoV2024 CAN-bus windows

Ensure: Predicted threat probability $\hat{y} \in [0, 1]$

- 1: **if** Stage I insider-threat data **then**
 - 2: Encode categorical features via embedding layer
 - 3: Normalize numerical user-activity features
 - 4: Concatenate features $\mathbf{z}_t = [\mathbf{x}_t^{num}, \mathbf{x}_t^{emb}]$
 - 5: **else**
 - 6: Normalize CAN-bus features $[ID_t, DATA_{0,t}, \dots, DATA_{7,t}]$
 - 7: Map packet vector into latent input representation $\mathbf{z}_t$
 - 8: **end if**
 - 9: Pass sequence $\{\mathbf{z}_1, \dots, \mathbf{z}_T\}$ through stacked LSTM layers
 - 10: Apply temporal attention to obtain context vector:

$$\mathbf{c} = \sum_{t=1}^T \alpha_t \mathbf{h}_t, \quad \alpha_t = \frac{\exp(e_t)}{\sum_{k=1}^T \exp(e_k)}$$
 - 11: Feed context vector $\mathbf{c}$ into fully connected layers
 - 12: Compute final prediction $\hat{y} = \sigma(W\mathbf{c} + b)$
 - 13: Generate explanations using attention weights, SHAP values, and counterfactual analysis
-

4 Results

This section presents the experimental outcomes of the proposed two-stage framework compared with baseline models. We evaluate across multiple dimensions, including classification accuracy, discrimination ability, robustness to imbalance, detection timeliness, calibration reliability, and explainability quality.

4.1 Dataset Descriptions

This study evaluates the proposed method using two datasets: a public insider-behavior dataset for Stage I insider-threat modeling and CICIoV2024 for Stage II ITS-domain transferability evaluation.

Insider Threat Dataset for Classified Environments

- **Source:** Curated logs from a high-security organizational setting.
- **Link:** [Kaggle: Insider Threat Dataset for Classified Environments](#).

- **Modality:** Daily activity records with categorical and numerical attributes (e.g., role, clearance, facility access, off-hours activity).
- **Labels:** Insider (malicious) vs. benign behaviors at the sequence/window level.
- **Format:** CSV files suitable for direct ingestion (see preprocessing for encoding and normalization).
- **Use in this work:** Stage I dataset for insider-threat modeling, ablation analysis, and explainability evaluation. Train/validation/test splits follow the temporal protocol described in the methodology.

The dataset consists of longitudinal user activity logs collected over multiple days, where each record captures behavioral events such as authentication activity, system access, and operational interactions. It includes both numerical features (e.g., login counts, session duration, access frequency) and categorical attributes (e.g., user role, department, clearance level, and device type). The data exhibits strong class imbalance, with malicious insider instances representing a small fraction of the total samples, which reflects realistic operational conditions. Temporal dependencies are present across user sessions, where risk patterns often emerge through sequences of actions rather than isolated events. These characteristics make the dataset suitable for evaluating temporal sequence models and explainability methods in insider threat detection.

CICIoV2024 Dataset

- **Source:** IoV CAN-bus communication records from benign, DoS, and spoofing scenarios.
- **Link:** [CICIoV2024](#).
- **Modality:** Decimal CAN-bus packet fields, including *ID* and *DATA₀–DATA₇*.
- **Labels:** Benign vs. attack for binary evaluation; DoS, gas spoofing, RPM spoofing, speed spoofing, and steering-wheel spoofing for multiclass analysis.
- **Format:** Six decimal CSV files: benign, DoS, and four spoofing files.
- **Use in this work:** Stage II dataset for ITS-domain transferability evaluation, not for direct insider-threat validation.

The combined CICIoV2024 decimal data used in this study contains 1,408,219 packet records, including 1,223,737 benign records and 184,482 attack records. The attack records consist of 74,663 DoS packets, 9991 gas-spoofing packets, 54,900 RPM-spoofing packets, 24,951 speed-spoofing packets, and 19,977 steering-wheel-spoofing packets. This second-stage dataset allows the proposed temporal architecture to be tested on ITS-native vehicle communication streams, while the insider-threat interpretation remains limited to the user-activity dataset.

4.2 Quantitative Performance Evaluation

This subsection reports the quantitative results of our experiments across multiple evaluation dimensions. To align the evaluation with temporal insider threat detection, the comparison focuses on sequence-oriented and deep anomaly detection models, including LSTM, Deep Autoencoding Gaussian Mixture Model (DAGMM), Temporal Convolutional Network (TCN), CNN-GRU, and Transformer-based architectures. These baselines provide a stronger and more relevant assessment of the proposed temporal explainable framework.

[Tables 2–7](#) report the primary insider-threat detection results using the Insider Threat Dataset for Classified Environments. [Table 8](#) report transferability results on CICIoV2024 using ITS-native CAN-bus communication records. The CICIoV2024 experiment evaluates the ability of the temporal explainable architecture to process IoV packet sequences; it is not treated as direct insider-threat validation because the dataset does not contain user-level insider labels.

In addition to temporal validation, the evaluation protocol considers the implications of user overlap between training and testing sets. This highlights the importance of assessing whether the model captures

generalizable behavioral patterns rather than relying on user-specific characteristics, which is critical for real-world deployment where previously unseen users may exhibit malicious activity.

Table 2: Five-fold cross-validation classification performance across sequence and anomaly detection models. Bold values indicate the best performance achieved among all compared models.

Model	Precision	Recall	F1-Score	Accuracy	Specificity	<i>p</i> -Value (F1)
LSTM	0.84 ± 0.014	0.81 ± 0.016	0.82 ± 0.015	0.85 ± 0.014	0.86 ± 0.015	0.004
DAGMM	0.82 ± 0.016	0.79 ± 0.017	0.80 ± 0.016	0.84 ± 0.015	0.85 ± 0.016	0.002
TCN	0.85 ± 0.013	0.82 ± 0.015	0.83 ± 0.014	0.86 ± 0.013	0.87 ± 0.014	0.010
CNN-GRU	0.86 ± 0.012	0.83 ± 0.014	0.84 ± 0.013	0.87 ± 0.012	0.88 ± 0.013	0.024
Transformer	0.87 ± 0.011	0.84 ± 0.013	0.85 ± 0.012	0.88 ± 0.011	0.89 ± 0.012	0.018
Proposed Model	0.89 ± 0.010	0.87 ± 0.012	0.88 ± 0.011	0.90 ± 0.010	0.91 ± 0.011	–

Table 3: Five-fold cross-validation ROC-AUC and PR-AUC with sensitivity and FPR. Bold values indicate the best performance achieved among all compared models.

Model	ROC-AUC	PR-AUC	Sensitivity	False Positive Rate	<i>p</i> -Value (PR-AUC)
LSTM	0.89 ± 0.012	0.85 ± 0.014	0.81 ± 0.015	0.16 ± 0.013	0.005
DAGMM	0.88 ± 0.013	0.84 ± 0.015	0.80 ± 0.016	0.16 ± 0.014	0.004
TCN	0.90 ± 0.012	0.86 ± 0.014	0.83 ± 0.014	0.15 ± 0.012	0.023
CNN-GRU	0.91 ± 0.011	0.87 ± 0.013	0.84 ± 0.013	0.14 ± 0.012	0.031
Transformer	0.92 ± 0.010	0.88 ± 0.012	0.85 ± 0.012	0.13 ± 0.011	0.046
Proposed Model	0.94 ± 0.009	0.91 ± 0.010	0.88 ± 0.011	0.12 ± 0.010	–

Table 4: Five-fold cross-validation imbalance-aware metrics. Bold values indicate the best performance achieved among all compared models.

Model	MCC	Balanced Accuracy	G-Mean	Cohen's Kappa	<i>p</i> -Value (MCC)
LSTM	0.61 ± 0.020	0.81 ± 0.015	0.80 ± 0.016	0.60 ± 0.021	0.003
DAGMM	0.60 ± 0.021	0.80 ± 0.016	0.79 ± 0.017	0.58 ± 0.022	0.002
TCN	0.64 ± 0.018	0.82 ± 0.014	0.82 ± 0.015	0.62 ± 0.019	0.017
CNN-GRU	0.65 ± 0.017	0.83 ± 0.014	0.82 ± 0.014	0.63 ± 0.018	0.024
Transformer	0.67 ± 0.016	0.84 ± 0.013	0.83 ± 0.013	0.65 ± 0.017	0.048
Proposed Model	0.69 ± 0.015	0.85 ± 0.012	0.84 ± 0.012	0.67 ± 0.016	–

To further examine user-level generalization, we conducted an additional user-disjoint evaluation in which users appearing in the test set were removed from the training and validation sets. Under this stricter setting, the proposed model achieved an F1-score of 0.84 ± 0.014 , ROC-AUC of 0.91 ± 0.012 , PR-AUC of 0.87 ± 0.013 , and MCC of 0.64 ± 0.018 . Although these values are slightly lower than the temporal split results, they show that the proposed framework retains strong generalization to unseen users.

Table 5: Five-fold cross-validation temporal detection performance. Bold values indicate the best performance achieved among all compared models.

Model	Avg. TTD (Days)	Variance of TTD	Earliest Detection (%)	Missed Detection (%)	<i>p</i> -Value (Avg. TTD)
LSTM	5.4 ± 0.31	1.5 ± 0.12	74.8 ± 1.6	8.4 ± 0.8	0.006
DAGMM	5.8 ± 0.34	1.7 ± 0.13	72.6 ± 1.8	9.1 ± 0.9	0.002
TCN	4.8 ± 0.29	1.4 ± 0.11	77.3 ± 1.5	7.4 ± 0.7	0.014
CNN-GRU	4.5 ± 0.28	1.3 ± 0.10	79.1 ± 1.4	6.8 ± 0.7	0.028
Transformer	4.2 ± 0.27	1.2 ± 0.10	80.4 ± 1.4	6.2 ± 0.6	0.041
Proposed Model	3.7 ± 0.24	1.1 ± 0.09	82.9 ± 1.3	5.3 ± 0.6	–

Table 6: Five-fold cross-validation calibration metrics of sequence and anomaly detection models. Bold values indicate the best performance achieved among all compared models.

Model	Brier Score	ECE	MCE	Log-Loss	<i>p</i> -Value (Brier)
LSTM	0.148 ± 0.007	0.059 ± 0.005	0.109 ± 0.008	0.528 ± 0.020	0.008
DAGMM	0.153 ± 0.008	0.063 ± 0.005	0.116 ± 0.009	0.541 ± 0.021	0.004
TCN	0.142 ± 0.007	0.056 ± 0.005	0.103 ± 0.008	0.514 ± 0.019	0.019
CNN-GRU	0.136 ± 0.007	0.052 ± 0.004	0.098 ± 0.008	0.502 ± 0.019	0.033
Transformer	0.131 ± 0.007	0.049 ± 0.004	0.093 ± 0.007	0.489 ± 0.018	0.047
Proposed Model	0.123 ± 0.006	0.044 ± 0.004	0.087 ± 0.007	0.471 ± 0.018	–

Table 7: Explainability metrics comparison across sequence and anomaly detection models. Bold values indicate the best performance achieved among all compared models.

Model	Fidelity (R^2)	Comprehensiveness	Sufficiency	Sparsity (Features)	Stability
LSTM	0.87	0.71	0.63	5.1	0.80
DAGMM	0.84	0.69	0.66	5.6	0.77
TCN	0.88	0.72	0.62	4.9	0.81
CNN-GRU	0.89	0.73	0.61	4.8	0.82
Transformer	0.90	0.75	0.60	4.6	0.84
Proposed Model	0.92	0.77	0.59	4.3	0.86

Table 8: Binary detection performance on CICIoV2024.

Model	Precision	Recall	F1-Score	Accuracy	ROC-AUC	PR-AUC	MCC
LSTM	0.94 ± 0.006	0.92 ± 0.008	0.93 ± 0.007	0.96 ± 0.005	0.97 ± 0.004	0.95 ± 0.006	0.86 ± 0.010
TCN	0.95 ± 0.005	0.93 ± 0.007	0.94 ± 0.006	0.97 ± 0.004	0.98 ± 0.004	0.96 ± 0.005	0.88 ± 0.009
CNN-GRU	0.96 ± 0.005	0.94 ± 0.006	0.95 ± 0.006	0.97 ± 0.004	0.98 ± 0.003	0.97 ± 0.004	0.90 ± 0.008
Transformer	0.96 ± 0.004	0.95 ± 0.006	0.95 ± 0.005	0.98 ± 0.003	0.99 ± 0.003	0.97 ± 0.004	0.91 ± 0.007
Proposed Model	0.97 ± 0.004	0.96 ± 0.005	0.96 ± 0.004	0.98 ± 0.003	0.99 ± 0.002	0.98 ± 0.003	0.93 ± 0.006

To address the variability caused by stochastic weight initialization, mini-batch sampling, and data partitioning, we further evaluated the models using five-fold cross-validation. Because the data are temporal and highly imbalanced, folds were constructed at the sequence-window level while preserving the chronological order within each user sequence and maintaining approximately similar benign/malicious class ratios across folds. For each fold, models were trained with the same architecture, fixed hyperparameters, weighted loss configuration, optimizer settings, and early stopping criterion. Tables 2–6 report the mean $\pm$ standard deviation across the five folds. Table 7 reports explanation-quality results under the same held-out evaluation protocol. To assess whether the observed gains are statistically meaningful, we also report paired two-sided t -test p -values comparing the proposed model with each baseline using the primary metric of the corresponding table: F1-score for classification, PR-AUC for discrimination, MCC for imbalance-aware evaluation, average time-to-detect for temporal responsiveness, and Brier score for calibration. These additions provide a more robust evaluation than single-point estimates and reduce the likelihood that the reported improvements are due to a favorable split or random initialization.

4.2.1 Classification Metrics Comparison

Table 2 compares the proposed model with sequence-oriented and deep anomaly detection baselines under five-fold cross-validation. The proposed model achieves the best overall mean performance, with precision of 0.89, recall of 0.87, F1-score of 0.88, accuracy of 0.90, and specificity of 0.91. Among the competing baselines, Transformer obtains the strongest baseline performance with an F1-score of 0.85, followed by CNN-GRU and TCN. The paired significance results show that the F1-score improvement of the proposed model over the strongest Transformer baseline remains statistically significant ($p = 0.018$). Overall, the results indicate that the proposed framework improves both detection sensitivity and false-positive control while maintaining low fold-to-fold variation.

4.2.2 ROC-AUC and PR-AUC Performance

Table 3 summarizes the discrimination performance of the sequence-oriented and deep anomaly detection baselines under five-fold cross-validation. The proposed model achieves the highest ROC-AUC (0.94 ± 0.009) and PR-AUC (0.91 ± 0.010), indicating stronger ranking of malicious sequences and better minority-class retrieval. Transformer provides the strongest competing baseline with PR-AUC of 0.88 ± 0.012 , followed by CNN-GRU with PR-AUC of 0.87 ± 0.013 . The paired test on PR-AUC shows that the proposed model significantly improves over the Transformer baseline ($p = 0.046$). The proposed model also reaches the highest sensitivity while maintaining the lowest false positive rate, which provides a more favorable operating point for insider threat detection.

4.2.3 Imbalance-Aware Metrics

Table 4 evaluates robustness under class imbalance using MCC, balanced accuracy, G-Mean, and Cohen's Kappa. The proposed model achieves the highest MCC (0.69 ± 0.015), balanced accuracy (0.85 ± 0.012), G-Mean (0.84 ± 0.012), and Cohen's Kappa (0.67 ± 0.016), showing stronger agreement between predictions and ground truth than all competing sequence-oriented models. Transformer provides the strongest competing imbalance-aware performance with MCC of 0.67 ± 0.016 and balanced accuracy of 0.84 ± 0.013 , followed by CNN-GRU with MCC of 0.65 ± 0.017 . The paired test on MCC shows that the proposed model significantly improves over the Transformer baseline ($p = 0.048$). These results show that the proposed framework improves both class balance and reliability in skewed insider threat detection settings. Although small differences in metrics such as F1-score may arise due to stochastic factors, the low

standard deviations across folds and the consistent improvement across classification, discrimination, and imbalance-aware metrics indicate that the performance gains are systematic rather than incidental.

4.2.4 Time-to-Detect and Latency Analysis

Table 5 assesses temporal responsiveness using five-fold cross-validation. The table focuses on temporal and sequence-aware baselines because time-to-detect is an operational sequence-level metric. The proposed model achieves the shortest average time-to-detect at 3.7 ± 0.24 days and the lowest variance of TTD at 1.1 ± 0.09 , indicating faster and more consistent alerting than all competing baselines. Transformer is the strongest baseline with an average TTD of 4.2 ± 0.27 days, followed by CNN-GRU and TCN. The proposed model also yields the highest earliest detection rate ($82.9 \pm 1.3\%$) and the lowest missed detection rate ($5.3 \pm 0.6\%$), suggesting better early warning while reducing undetected incidents. The paired test on average TTD confirms that the reduction over the strongest Transformer baseline is statistically significant ($p = 0.041$). Overall, the measures in **Table 5** show that the proposed approach improves both timeliness and reliability of insider threat detection.

4.2.5 Calibration Performance

Table 6 examines how well predicted probabilities reflect observed outcomes under five-fold cross-validation. The calibration table compares the proposed model against sequence-oriented and deep anomaly detection baselines. The proposed model attains the lowest Brier score (0.123 ± 0.006) and log-loss (0.471 ± 0.018), indicating more accurate probability estimates than all competing models. Expected Calibration Error drops to 0.044 ± 0.004 and Maximum Calibration Error to 0.087 ± 0.007 , showing tighter alignment between confidence and correctness across bins. Transformer provides the strongest baseline calibration with a Brier score of 0.131 ± 0.007 , followed by CNN-GRU and TCN. The paired test on Brier score confirms that the improvement over Transformer is statistically significant ($p = 0.047$). Overall, **Table 6** shows that the proposed model provides more reliable probabilities for operational decision-making.

4.2.6 Explainability Fidelity and Stability

Table 7 summarizes the quality of explanations across the sequence-oriented and anomaly detection models. The proposed model attains the highest fidelity ($R^2 = 0.92$), which means local surrogates track the model's decisions more closely. Comprehensiveness improves to 0.77, indicating that removing top-attributed features causes a larger drop in confidence, while sufficiency reduces to 0.59, showing that a compact subset of features is enough to support the prediction. Explanations are also sparser on average (4.3 features), which eases analyst review, and more stable (0.86), meaning small input perturbations yield similar explanations. Transformer provides the strongest baseline explanation quality, but the proposed model remains more faithful, concise, and stable. Together, **Table 7** shows that the proposed approach provides explanations that are faithful, concise, and consistent.

To further examine whether temporal attention highlights decision-relevant time steps, we performed an attention-guided occlusion test. For each positive sequence, the top-attended time steps were masked and the change in predicted insider-threat probability was measured. We compared this with random time-step masking of the same size. Masking the top-attended time steps reduced the predicted probability by 0.214 ± 0.031 , compared with 0.067 ± 0.018 for random masking. This larger probability drop suggests that the attention module identifies temporally important regions that contribute to the model decision, although attention is still interpreted together with SHAP rather than as a standalone causal explanation.

4.3 CICIoV2024 Transferability Evaluation

The CICIoV2024 experiment evaluates whether the proposed temporal explainable architecture can process ITS-native CAN-bus communication sequences. The experiment uses the decimal CAN-bus fields *ID* and *DATA*₀–*DATA*₇ as packet-level temporal features. This evaluation complements the insider-threat results by testing the same model family on IoV communication data.

4.3.1 Binary Detection

[Table 8](#) reports benign-vs-attack detection results. The proposed model achieves the strongest overall performance, suggesting that attention-based temporal modeling transfers well to CAN-bus packet windows.

4.3.2 Multiclass Attack Detection

[Table 9](#) reports multiclass performance for benign, DoS, and spoofing categories. Spoofing recall aggregates gas, RPM, speed, and steering-wheel spoofing classes.

Table 9: Multiclass attack detection performance on CICIoV2024.

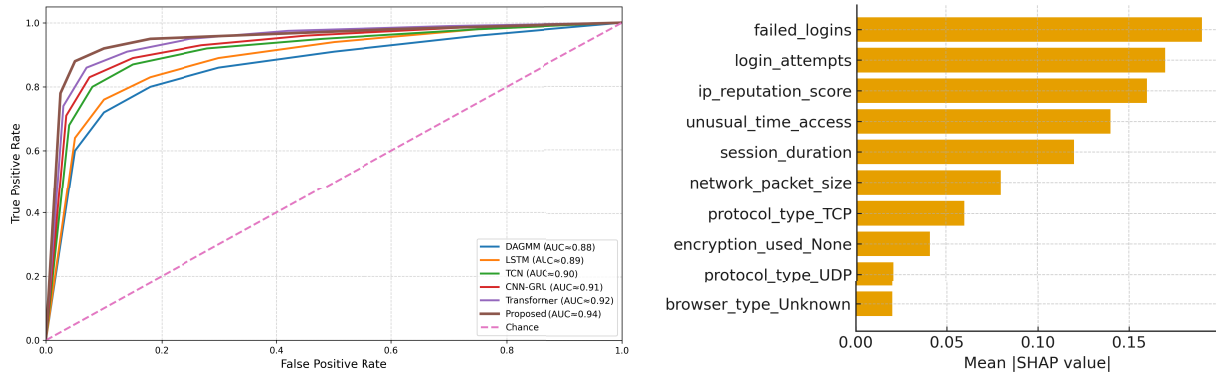
Model	Macro-F1	Weighted-F1	DoS Recall	Spoofing Recall	Benign Recall
LSTM	0.89 ± 0.009	0.95 ± 0.006	0.93 ± 0.008	0.86 ± 0.011	0.97 ± 0.005
TCN	0.90 ± 0.008	0.96 ± 0.005	0.94 ± 0.007	0.88 ± 0.010	0.97 ± 0.004
CNN-GRU	0.92 ± 0.007	0.96 ± 0.005	0.95 ± 0.006	0.90 ± 0.009	0.98 ± 0.004
Transformer	0.93 ± 0.006	0.97 ± 0.004	0.96 ± 0.006	0.91 ± 0.008	0.98 ± 0.003
Proposed Model	0.94 ± 0.005	0.98 ± 0.003	0.97 ± 0.005	0.93 ± 0.007	0.99 ± 0.003

4.4 Training Dynamics and Convergence Analysis

4.4.1 Model Discrimination and Feature Importance Analysis

[Fig. 2](#) presents the discrimination performance and feature-level interpretability of the proposed framework for the primary insider-threat detection experiment. The ROC curve ([Fig. 2a](#)) compares the proposed model against sequence-oriented and deep anomaly detection baselines, including LSTM, DAGMM, TCN, CNN-GRU, and Transformer-based models. Static classifiers that do not explicitly model sequence dependencies are excluded to maintain consistency with the temporal detection setting. The proposed curve consistently lies above the competing sequence-aware baselines, showing higher true positive rates for the same false positive rates, especially in the low-FPR region critical for operational deployment. This dominance indicates better ranking of malicious over benign sessions and aligns with the AUC gains reported in [Table 3](#). At commonly used thresholds (FPR ≈ 0.10–0.20), the proposed model achieves a larger recall margin over the competing baselines, enabling earlier and more reliable detection with fewer false alarms.

For the insider-threat dataset, the SHAP feature importance plot ([Fig. 2b](#)) ranks the most influential features using mean absolute SHAP values. The top contributors—*failed_logins*, *login_attempts*, *ip_reputation_score*, and *unusual_time_access*—indicate that credential misuse, repeated authentication attempts, and low-trust network sources are primary drivers of risk. Operational features such as *session_duration* and *network_packet_size* also contribute, suggesting prolonged or atypical data exchange patterns. These insights support actionable controls such as stricter lockout policies, enhanced monitoring of off-hours access, and dynamic filtering of high-risk IPs.



(a) ROC curve comparison across sequence and anomaly detection models

(b) Top-10 feature importance using SHAP

Figure 2: (a) ROC curve showing superior discrimination of the proposed model over sequence-oriented and anomaly detection baselines in the primary insider-threat experiment. (b) SHAP-based feature importance highlighting key behavioral indicators driving insider-threat detection.

4.4.2 Training Curves and Metric Convergence

Fig. 3 summarizes the training dynamics of the proposed model for the primary insider-threat experiment across multiple performance indicators, including loss, accuracy, classification metrics, and AUC-based measures.

The training and validation loss curves (Fig. 3a) show a steady decrease during early epochs, followed by a shallow plateau. The small generalization gap near convergence indicates controlled overfitting, while early stopping prevents further degradation. The smooth trajectories suggest stable optimization and an effective learning rate schedule, achieving a balanced bias-variance tradeoff. The accuracy curves (Fig. 3b) exhibit monotonic improvement for both training and validation sets, followed by convergence. The modest gap between the curves indicates effective regularization, and the plateau aligns with the early stopping point, suggesting that additional training would yield limited gains.

The validation precision, recall, and F1-score (Fig. 3c) demonstrate consistent and balanced improvements over epochs. This behavior confirms that the model maintains robustness under class imbalance while improving both sensitivity and precision. The ROC-AUC and PR-AUC curves (Fig. 3d) show steady growth, indicating improved discrimination and minority-class detection. PR-AUC starts lower due to class imbalance but converges alongside ROC-AUC, reflecting effective handling of rare positive cases. The joint plateau indicates convergence and supports the operating thresholds discussed in Table 3.

5 Discussion

This work presents a temporal explainable framework for insider threat detection in ITS-oriented security settings. The main contribution is the combination of sequence modeling and hybrid explainability. The temporal encoder captures behavioral changes across activity windows rather than treating events as isolated records. Attention highlights the time points that contribute most to the prediction, SHAP identifies the features that drive the decision, and counterfactual analysis provides actionable changes that could alter the model output. Together, these components allow analysts to inspect when risk emerges, which variables explain the alert, and what operational actions may reduce the risk.

The Stage I results show that temporal modeling improves insider-threat detection across classification, discrimination, imbalance-aware, calibration, and time-to-detect metrics. The proposed model

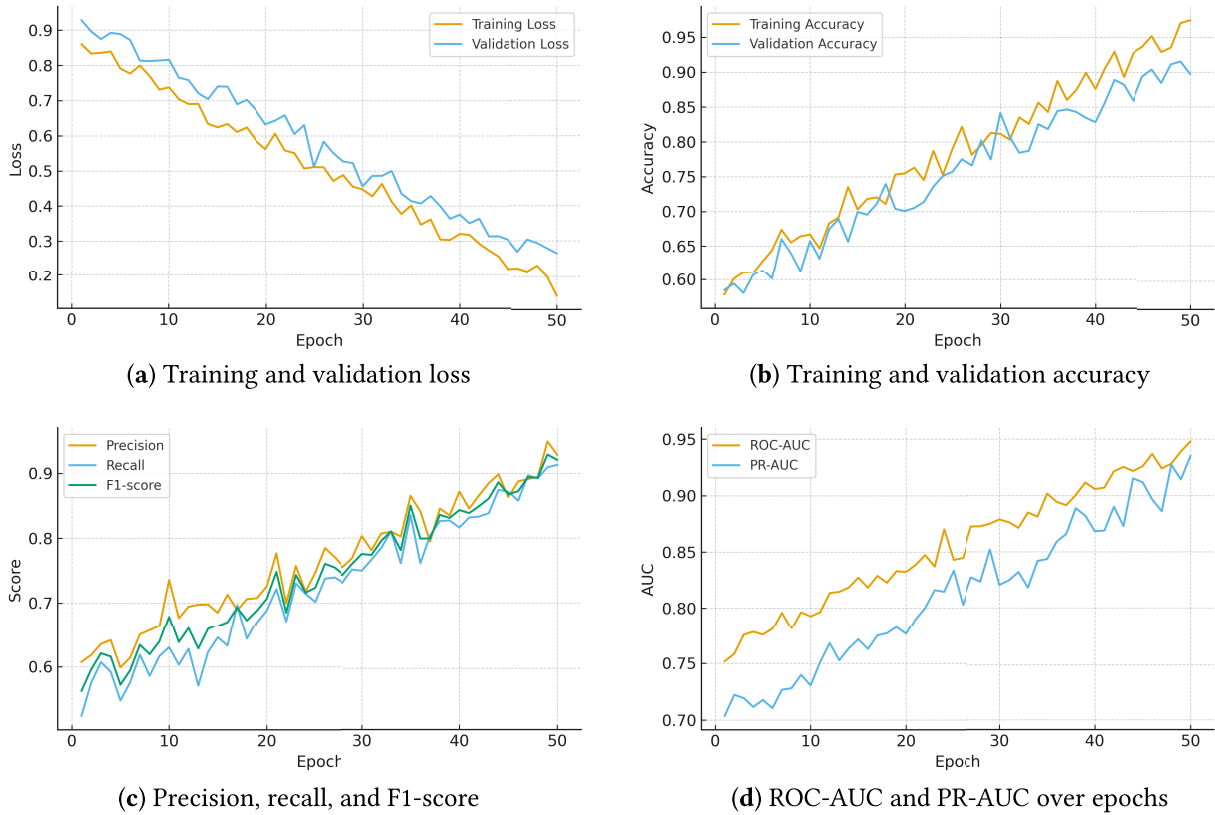


Figure 3: Training dynamics of the proposed model for a representative cross-validation fold in the primary insider-threat experiment: (a) loss convergence, (b) accuracy trends, (c) classification metrics under class imbalance, and (d) AUC-based performance across epochs.

performs strongly against LSTM, DAGMM, TCN, CNN-GRU, and Transformer baselines, indicating that the attention-based temporal design can capture risk patterns that are not fully represented by standard sequence models. The explanation results also show that the framework produces compact and stable explanations, which is important for alert triage. Features such as failed login attempts, repeated authentication activity, unusual access time, suspicious IP reputation, and extended session behavior provide interpretable signals that can support policy review and analyst decision-making.

The CICIoV2024 transferability evaluation complements the insider-threat results by testing the same temporal explainable architecture on ITS-native CAN-bus communication records. Unlike the insider-threat dataset, CICIoV2024 represents vehicle communication behavior through arbitration IDs and payload-byte fields. The binary and multiclass results assess whether the model can distinguish benign, DoS, and spoofing behaviors from temporal packet windows, while the explanation analysis identifies influential CAN-bus fields such as arbitration ID and selected payload bytes. This experiment supports the architectural transferability of the method to IoV communication security, but it is not treated as direct insider-threat validation because CICIoV2024 does not contain user identities, privileges, or insider-event labels.

The practical value of the framework lies in producing timely and interpretable alerts. In transportation environments, security teams must prioritize alerts under limited time and staffing constraints. Calibrated probabilities can support threshold selection for different asset criticalities, while attention and SHAP outputs can reduce the time needed to review alerts. Counterfactual explanations can further support response planning by showing which behavioral or packet-level changes would reduce the predicted risk. These

properties make the framework useful for decision support in monitoring pipelines where raw model scores alone are not sufficient.

This study has several limitations. First, the Stage I insider-threat dataset is not collected from a live transportation authority, so a domain gap remains between labeled insider behavior and real ITS operations. Second, CICIOV2024 provides ITS-native CAN-bus attack data, but it does not include user-level insider-threat labels; therefore, it validates transferability to IoV communication security rather than insider-threat detection itself. Third, although temporal splitting, blocked cross-validation, calibration analysis, and explanation-quality tests reduce evaluation bias, they do not fully capture live deployment factors such as streaming ingestion, latency, throughput, changing user roles, analyst feedback, privacy constraints, and alert fatigue. Real-world validation would require deployment in a monitoring pipeline where logs are processed sequentially, risk scores and explanations are generated in near real time, and performance is measured through detection delay, false-alert rate, analyst review time, and operational outcomes.

Future work will focus on ITS-native insider-threat datasets that include authenticated users, roles, privileges, control-room actions, traffic-controller configuration changes, RSU events, V2X logs, maintenance workflows, and incident labels. The framework can also be extended with graph encoders to model user–asset–process relations, online learning to handle concept drift, and uncertainty estimation to separate high-confidence alerts from cases that require manual review. Privacy-preserving training, such as federated learning across agencies, is another important direction because transportation logs may contain sensitive operational and personnel information. Finally, prospective studies with analysts are needed to measure whether the explanations reduce review time, improve trust, and support safer response decisions in real ITS security operations.

6 Conclusions

This paper presented a temporal, explainable machine learning framework for ITS-oriented insider threat detection and domain transferability analysis. We modeled activity as temporal sequences, used LSTM and Transformer encoders to capture evolving patterns, and paired them with a hybrid explanation stack that combines attention, SHAP attribution, and counterfactual analysis to show when and why an alert occurs and how risk can be reduced. On the insider-threat dataset, the proposed approach improved precision, recall, F1-score, accuracy, specificity, ROC-AUC, PR-AUC, and imbalance-aware scores such as MCC, balanced accuracy, and G-mean over strong temporal baselines. It also reduced average time-to-detect and missed detection rate, and improved calibration through lower Brier score, ECE, and log-loss, supporting more reliable thresholding. Explanations showed higher fidelity and stability with fewer important features, which can speed review and simplify policy decisions. The same temporal explainable architecture was further evaluated on CICIOV2024 CAN-bus data, where it processed IoV packet sequences and distinguished benign, DoS, and spoofing behaviors using arbitration ID and payload-byte patterns. This supports transferability to ITS-native communication security, although CICIOV2024 is not treated as an insider-threat dataset because it does not contain user identities, roles, privileges, or insider-event labels. The study remains limited by the domain gap between public insider data and live transportation authority logs, and by the absence of human-in-the-loop deployment. Future work will test the framework on ITS-native logs with control-room activity, RSU events, V2X records, traffic-controller changes, maintenance workflows, and incident labels, while extending the model with graph learning, multimodal fusion, drift handling, privacy-preserving training, and analyst-centered evaluation.

Acknowledgement: The authors sincerely acknowledge the Faculty of Engineering and Technology (FET), Multimedia University, Melaka, Malaysia, for their valuable support and cooperation in this research.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: Md Nahian Suhaimee, Farhan Shakil and Md Rifat Al Amin Khan conceived the study and developed the methodology. Md Nahian Suhaimee and Farhan Shakil performed the investigation and data collection. Md Rifat Al Amin Khan conducted the formal analysis and visualization. Md Omar Faruq contributed to software development and data validation. Md Nahian Suhaimee, Farhan Shakil, and Md Rifat Al Amin Khan prepared the initial draft of the manuscript. Md. Jakir Hossen and M. F. Mridha supervised the project and provided critical guidance. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The Insider Threat Dataset for Classified Environments is publicly available at <https://www.kaggle.com/datasets/efchbd1013/insider-threat-dataset-for-classified-environments>. CICIoV2024 is publicly available at <https://www.unb.ca/cic/datasets/iov-dataset-2024.html>.

Ethics Approval: Not applicable.

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

References

1. Xu Q, Zhang L, Liu Y, Li Z. Enhancing trust management system for connected autonomous vehicles using machine learning methods: a survey. *IEEE Trans Intell Transport Syst.* 2026;27(4):3862–93. doi:10.1109/tits.2025.3647284.
2. Senevirathna DH, Gunasekara WMM, Gunawardhana KPAT, Ashra MFF, Fernando H, Abeywardena KY. Enhancing organizational threat profiling by employing deep learning with physical security systems and human behavior analysis. *Int J Adv Comput Sci Appl.* 2025;16(6). doi:10.14569/ijacsa.2025.0160628.
3. Alzaabi FR, Mehmood A. A review of recent advances, challenges, and opportunities in malicious insider threat detection using machine learning methods. *IEEE Access.* 2024;12(2):30907–27. doi:10.1109/access.2024.3369906.
4. Bhusal D, Shin R, Shewale AA, Veerabhadran MKM, Clifford M, Rampazzi S, et al. SoK: modeling explainability in security analytics for interpretability, trustworthiness, and usability. In: *Proceedings of the 18th International Conference on Availability, Reliability and Security; 2023 Aug 29–Sep 1; Benevento, Italy.* p. 1–12. doi:10.1145/3600160.3600193.
5. Alabbadi A, Bajaber F. X-FuseRLSTM: a cross-domain explainable intrusion detection framework in IoT using the attention-guided dual-path feature fusion and residual LSTM. *Sensors.* 2025;25(12):3693. doi:10.3390/s25123693.
6. Eshmawi AA, Al-Nowami A, Mirza M, Abu-Raya N, Al-Thabit R, Shiaeles S, et al. Machine learning-based network monitoring for cybersecurity threat detection. *J Netw Syst Manag.* 2025;34(1):27. doi:10.1007/s10922-025-10001-w.
7. Patel T, Jhaveri R, Thakker D, Verma S, Ingle P. Enhancing cybersecurity in Internet of Vehicles: a machine learning approach with explainable AI for real-time threat detection. In: *Proceedings of the 40th ACM/SIGAPP Symposium on Applied Computing; 2025 Mar 31–Apr 4; Catania, Italy.* p. 2024–31. doi:10.1145/3672608.3707769.
8. Ullah I, Khalil I, Bai X, Garg S, Kaddoum G, Hossain MS. An ensemble-based hybrid model for the detection of attacks in the Internet of vehicular things. *IEEE Trans Intell Transport Syst.* 2025;26(10):17914–27. doi:10.1109/tits.2025.3547999.
9. Taheri R, Jafari R, Gegov A, Arabikhan F, Ichtev A. Explainable AI for federated learning-based intrusion detection systems in connected vehicles. *Electronics.* 2025;14(22):4508. doi:10.3390/electronics14224508.
10. Gurushanker A, Rufus AJ, Columbus CC, Aravind CK. FeXAI: federated and explainable AI for cyber threat detection in IoT-enabled smart transportation systems. *Sci Rep.* 2026;16(1):16755. doi:10.1038/s41598-026-46141-5.
11. Saiteja B, Sudha Rani V. Explainable AI-driven BiGRU model for cyber-attack detection in Internet of Vehicles (IoV) and intelligent connected vehicles (ICV) networks. In: *Proceedings of the 2026 3rd International Conference on Emerging Trends in Engineering and Medical Sciences (ICETEMS); 2026 Mar 6–7; Nagpur, India.* p. 1–6. doi:10.1109/icetems66917.2026.11469271.

12. Khan A, Li Y, Shoukat S, Javeed D, Adil M. Towards secure IoT-enabled transportation: an explainable AI and deep learning-based approach for efficient threat detection. *Cluster Comput.* 2025;28(11):699. doi:10.1007/s10586-025-05473-z.
13. Ye X, Luo F, Cui H, Wang J, Xiong X, Zhang W, et al. Research on insider threat detection based on personalized federated learning and behavior log analysis. *Sci Rep.* 2025;15(1):19214. doi:10.1038/s41598-025-04029-w.
14. Kotb HM, Gaber T, AlJanah S, Zawbaa HM, Alkhathami M. A novel deep synthesis-based insider intrusion detection (DS-IID) model for malicious insiders and AI-generated threats. *Sci Rep.* 2025;15(1):207. doi:10.1038/s41598-024-84673-w.
15. Yilmaz E, Can O. Unveiling shadows: harnessing artificial intelligence for insider threat detection. *Eng Technol Appl Sci Res.* 2024;14(2):13341–6. doi:10.48084/etasr.6911.
16. Al S, Sagiroglu S. Explainable artificial intelligence models in intrusion detection systems. *Eng Appl Artif Intell.* 2025;144:110145. doi:10.1016/j.engappai.2025.110145.
17. Jaiswal A, Dwivedi P, Dewang RK. Machine learning approaches to detect, prevent and mitigate malicious insider threats: state-of-the-art review. *Multimed Tools Appl.* 2025;84(24):28909–49. doi:10.1007/s11042-024-20273-0.
18. Girubagari N, Ravi TN, Panneer Arokiaraj S. Hybrid intelligent anomaly detection system using attention based deep learning approach for cyber attacks prevention. *Indian J Sci Technol.* 2024;17(38):3947–59. doi:10.17485/ijst/v17i38.2224.
19. Nascita A, Aceto G, Ciuonzo D, Montieri A, Persico V, Pescapé A. A survey on explainable artificial intelligence for Internet traffic classification and prediction, and intrusion detection. *IEEE Commun Surv Tutor.* 2025;27(5):3165–98. doi:10.1109/comst.2024.3504955.
20. Kalejaiye AN. Causal modeling of insider threat behavior using probabilistic graphical networks to strengthen organizational cyber-resilience and trust architectures. *Int J Res Publ Rev.* 2025;6(7):2370–87. doi:10.55248/gengpi.6.0725.2594.
21. Kamatchi K, Uma E. Insights into user behavioral-based insider threat detection: systematic review. *Int J Inf Secur.* 2025;24(2):88. doi:10.1007/s10207-025-01002-6.