SvedbergOpen
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

TMK-TRAF: A Multi-Task Framework with Threat-Risk Adaptive Fusion for Integrated Banking Intrusion and Fraud Detection

Ponnarasu Kulanthaisamy

Solutions Architect Data Analytics, Independent Researcher. E-mail: ponnarasuk@gmail.com

Abstract
Banking security requires simultaneous detection of network-level intrusions and transaction-level fraud, yet existing approaches typically address these threats independently and fail to exploit their shared security characteristics. This paper proposes TMK-TRAF, a multi-task deep learning framework that integrates Temporal Convolutional Network (TCN), Mamba, and Kolmogorov–Arnold Network (KAN) to learn complementary local-temporal, long-range, and nonlinear representations. The proposed Threat-Risk Adaptive Fusion (TRAF) mechanism performs task-specific fusion of these representations, while the Threat-Risk Consistency (TRC) loss promotes beneficial cross-task knowledge while preserving task-specific decision boundaries. A downstream BiLSTM–GRU–MLP ensemble performs intrusion and fraud classification, with SHAP providing feature-level interpretability. Network-flow data are generated using OMNeT++/INET, while banking fraud data are used for transaction-level evaluation, and CloudSim is employed for simulated scalability analysis. TMK-TRAF achieves 97.68% intrusion-detection accuracy with 97.26% F1-score and 98.30% fraud-detection accuracy with 93.12% F1-score. The results demonstrate the effectiveness of unified multi-task representation learning, adaptive fusion, and cross-task consistency for integrated banking-security analytics.

Keywords: Banking Security, Network Intrusion Detection, Multi-Task Learning, Explainable AI, Cloud Scalability.

1. Introduction

Banking security increasingly requires simultaneous protection of the communication infrastructure and financial transactions, as network intrusions and fraudulent activities can occur across interconnected security layers. Network intrusion detection must identify abnormal traffic and diverse attack behaviours, whereas banking fraud detection must recognize suspicious transaction patterns and account-level risk [1]. Although deep learning has improved both tasks, most existing approaches address intrusion detection and transaction fraud independently, limiting the ability to exploit shared security information. Recent studies have explored temporal convolutional networks, Mamba-based state-space modelling, KAN-based nonlinear learning, and multi-task learning, but these approaches have not established a unified framework for jointly modelling banking network threats and transaction fraud. [2]

To address this limitation, this paper proposes **TMK-TRAF**, a multi-task security framework that combines **TCN, Mamba, and KAN** to learn complementary local-temporal, long-range, and nonlinear security representations. These representations are integrated through the proposed **Threat-Risk Adaptive Fusion (TRAF)** mechanism, which learns task-specific feature contributions for intrusion and fraud detection. A **Threat-Risk Consistency (TRC)** loss further encourages useful cross-task information while maintaining separate task-specific classification decisions. The framework additionally incorporates SHAP-based explainability and CloudSim-based scalability evaluation to provide interpretable and scalable security analysis. The proposed design directly addresses the research gap identified in existing Mamba, KAN, and multi-task cybersecurity studies. [3]

The main contributions of this work are:

- **Unified multi-task security framework:** TMK-TRAF jointly performs **nine-class network intrusion detection and binary banking fraud detection** using a shared security representation with task-specific classification heads.
- **Complementary representation learning:** TCN, Mamba, and KAN are integrated to capture **local temporal patterns, long-range dependencies, and nonlinear feature relationships**, respectively.
- **Threat-Risk Adaptive Fusion:** TRAF dynamically assigns task-specific importance to the learned representations, enabling intrusion and fraud tasks to exploit different security characteristics.

- **Threat-Risk Consistency loss:** TRC promotes useful cross-task knowledge while preventing the shared representation from eliminating task-specific distinctions.
- **Explainable and scalable security analysis:** SHAP identifies influential intrusion and fraud features, while OMNeT++/INET network simulation and CloudSim provide simulated network-flow generation and scalability evaluation.

2. Related Works

Recent research has increasingly applied deep learning, temporal modeling, Mamba, KAN, and multi-task learning to cybersecurity and financial fraud detection. However, existing studies generally address network intrusion or transaction fraud independently. This section reviews representative recent approaches and identifies the gap addressed by the proposed TMK-TRAF framework.

2.1. Network Intrusion Detection

Recent intrusion detection research increasingly employs deep learning to capture complex network behavior. R. K. M. et al. (2026) [4] explored an AI-powered IDS, while Hnamte et al. (2025) [5] developed a lightweight deep convolutional approach for intrusion detection. Nazre et al. (2024) [6] demonstrated the suitability of TCNs for temporal network-flow modeling, and Akhi et al. (2026) [7] further investigated temporal CNNs for emerging cyber-threat detection. These studies demonstrate the effectiveness of deep temporal and convolutional representations; however, they primarily address intrusion detection as an independent task.

2.2. Mamba and KAN-Based Security Models

Recent studies have introduced state-space and nonlinear function-learning approaches into cybersecurity. Ding et al. (2026) [8] proposed NIDS-Mamba for lightweight IoT intrusion detection, while Feng et al. (2026) [9] developed MambaCNN for intrusion detection in Internet-of-Vehicles environments. Mohammed et al. (2026) [10] investigated KAN-based intrusion detection with SMOTE for imbalanced IoT data. Batur Dinler (2025) [11] further demonstrated a Mamba-KAN-Liquid hybrid for UAV cybersecurity. Although these studies establish the applicability of Mamba and KAN to security problems, they do not address the proposed task-specific Threat-Risk Adaptive Fusion (TRAF) or unified intrusion–fraud learning.

2.3. Banking Fraud Detection and Explainability

Machine-learning approaches have also been widely investigated for banking and financial fraud. Konsta et al. (2026) [12] examined machine-learning-based cyber-fraud detection in banking transactions, while Sami et al. (2025) [13] investigated ML-based bank transaction fraud detection. AbouGrad and Sankuru (2025) [14] considered decentralized machine learning for online banking fraud, and Babu et al. (2025) [15] explored context-aware fraud detection using pre- and post-transaction information. Xia et al. (2026) [16] further incorporated explainability into financial fraud detection. However, these approaches primarily focus on transaction fraud and do not jointly model network intrusion and banking fraud within a shared representation-learning framework.

2.4. Multi-Task Security and Research Gap

Multi-task learning has attracted increasing attention in cybersecurity. Ibrahim et al. (2024) systematically reviewed multi-task learning applications in cybersecurity and highlighted its potential for jointly addressing related security objectives. Nevertheless, existing studies do not provide a unified architecture that simultaneously learns network intrusion and banking fraud using complementary temporal, long-range, and nonlinear representations. Moreover, existing TCN, Mamba, KAN, and Mamba-KAN studies do not introduce a **task-specific Threat-Risk Adaptive Fusion (TRAF)** mechanism or the proposed **Threat-Risk Consistency (TRC) loss**. Therefore, TMK-TRAF addresses this gap by jointly modelling intrusion and fraud through TCN, Mamba, and KAN representations, adaptively fusing them for each task, and enforcing cross-task consistency through TRC.

Table 1. Literature Survey of Recent Intrusion Detection, Fraud Detection, and Security Learning Methods

Author(s)	Method / Model	Application	Key Limitation
R. K. M. et al. (2026)	AI-powered IDS	Network intrusion	Limited unified multi-task learning
Hnamte et al. (2025)	Lightweight DCNN	Network intrusion	Primarily convolution-based representation
Nazre et al. (2024)	TCN-based IDS	Network intrusion	Focused on IDS only

Akhi et al. (2026)	Residual Temporal CNN	IoT cyber threats	Domain-specific temporal detection
Ding et al. (2026)	NIDS-Mamba	IoT intrusion	Mamba-focused single security task
Feng et al. (2026)	MambaCNN	Internet of Vehicles	Mamba–CNN hybrid, task-specific IDS
Mohammed et al. (2026)	KAN + SMOTE	IoT intrusion	KAN-based single-task detection
Batur Dinler (2025)	Mamba-KAN-Liquid	UAV cybersecurity	Hybrid anomaly detection, different domain
Konsta et al. (2026)	Machine learning	Banking cyber fraud	Conventional ML, simulated transactions
Sami et al. (2025)	Machine learning	Bank transaction fraud	ML-based single-task detection
AbouGrad and Sankuru (2025)	Decentralized ML	Online banking fraud	Focus on privacy/decentralization
Babu et al. (2025)	Context-aware hybrid ML	Banking transactions	Fraud-focused, not joint security learning
Xia et al. (2026)	Explainable multi-layer framework	Financial fraud	Fraud-focused explainability

Research Gap: Existing studies mainly address network intrusion or banking fraud as separate tasks. Although TCN, Mamba, KAN, and Mamba–KAN models have shown promise in security applications, they generally lack task-specific fusion and unified intrusion–fraud learning. Multi-task cybersecurity studies also provide limited mechanisms for coordinating heterogeneous security representations. Therefore, a unified framework with Threat-Risk Adaptive Fusion (TRAF) and Threat-Risk Consistency (TRC) is needed to jointly learn intrusion and fraud patterns while preserving task-specific decision boundaries.

3. Materials and Problem Formulation

This section describes the datasets, feature characteristics, preprocessing procedures, and mathematical formulation used to develop TMK-TRAF. The two data sources represent complementary banking-security perspectives: network-level threats and transaction-level fraud. The synthetic network-flow dataset is explicitly treated as simulated data, while the characteristics of the banking fraud dataset will be reported only from the verified source data.

3.1. System Overview

TMK-TRAF is designed as a unified multi-task banking-security framework that simultaneously detects **network intrusions** and **banking transaction fraud**. Network-flow and transaction data are first processed through task-specific pre-processing and threat/risk-aware input gates. The resulting representations are then projected into a shared feature space and processed in parallel by **TCN, Mamba, and KAN** to capture local temporal, long-range sequential, and nonlinear feature relationships, respectively. Their outputs are combined using the proposed **Threat-Risk Adaptive Fusion (TRAF)** mechanism, which learns task-specific fusion weights for intrusion and fraud detection. The fused representations are passed to separate prediction heads, while the **Threat-Risk Consistency (TRC)** loss promotes useful shared information between the two tasks. Focal Loss addresses class imbalance, and SHAP provides post-hoc explanations of model decisions. A cloud-oriented simulation is finally used to evaluate scalability without claiming deployment on a real cloud platform.

3.2. Dataset Description

Two datasets are used: a synthetic network-flow dataset for nine-class intrusion detection and a banking fraud dataset for binary fraud detection. The network-flow data represent benign and attack traffic, while the fraud data represent transaction-level activities.

3.2.1. Synthetic Banking Network-Flow Dataset

The network-security dataset was generated using **OMNeT++** to simulate banking-oriented normal and malicious network activities. The current exported dataset contains **1,000,000 flow records and 34 columns**, comprising 32 input features and two target fields (label, label_id). Nine classes are represented:

Benign, DDoS, DoS, PortScan, BruteForce, Botnet, DataExfiltration, WebAttack, and MITM. The dataset is synthetic and represents simulated banking network-flow behavior rather than real banking traffic. Python is used for preprocessing and TMK-TRAF implementation, while CloudSim is used separately for cloud-oriented scalability evaluation.

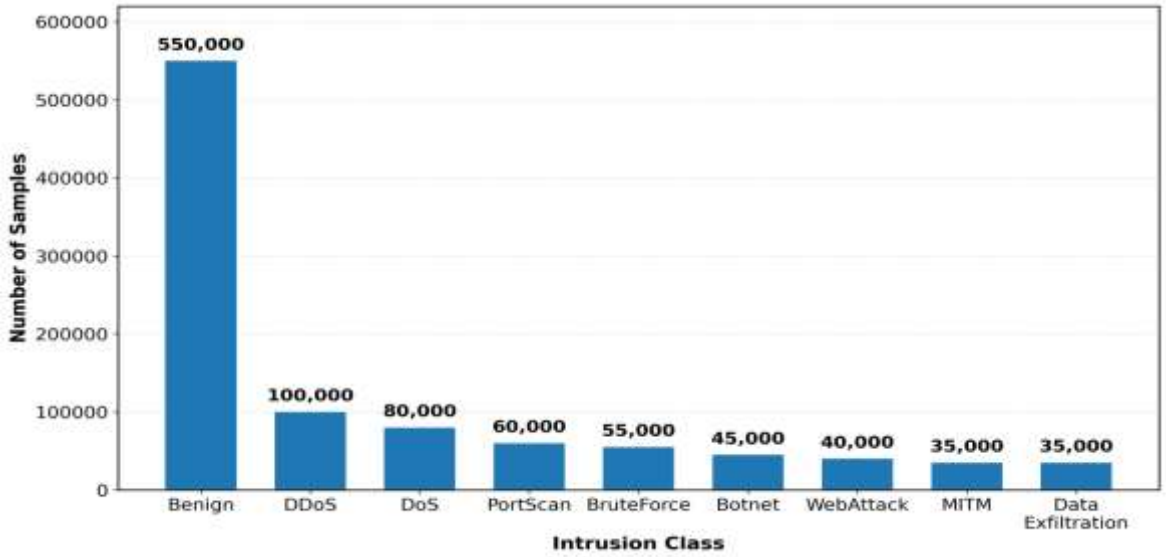

Fig 1. Dataset Class Distribution

Fig. 1 illustrates the class distribution of the synthetic banking network-flow dataset. Benign traffic dominates, while the nine intrusion categories exhibit lower and varying sample frequencies, indicating class imbalance.

Table 2. Network-Flow Feature Groups

Feature Group	Features
Network	src_id, dst_id, src_port, dst_port, protocol
Temporal	flow_duration_ms, iat_mean_ms, iat_std_ms
Packet/Byte	total_packets, fwd_packets, bwd_packets, total_bytes, fwd_bytes, bwd_bytes
Traffic Rate	flow_bytes_per_sec, flow_packets_per_sec
Packet Statistics	packet_size_mean, packet_size_std
TCP	tcp_syn, tcp_ack, tcp_rst
Connection	unique_dst_ports, unique_dst_ips, connection_rate
Entropy	source_entropy, destination_entropy
Banking/Security	api_requests, authentication_attempts, failed_logins, transaction_requests, server_response_ms, encrypted
Target	label, label_id

The class distribution shows moderate imbalance, particularly between benign and minority attack classes; therefore, **Focal Loss** is incorporated to reduce the influence of easily classified majority samples during multi-task training.

3.2.2. Banking Fraud Detection Dataset

The banking fraud dataset contains **10,000 transaction records and 20 columns**. The target variable is fraud_flag, with 8,749 normal transactions (87.49%) and 1,251 fraudulent transactions (12.51%). The dataset contains numerical transaction/risk attributes and categorical variables representing payment channels and authentication methods. No missing values or duplicate records were found.

Table 3. Banking Fraud Dataset Characteristics

Item	Description
Samples	10,000
Features	18 input features
Target	fraud_flag
Normal	8,749 (87.49%)

Fraud	1,251 (12.51%)
Numerical features	16
Categorical features	2

Table 3 summarizes the key characteristics of the banking fraud dataset, including its sample size, feature composition, target variable, class distribution, and data quality. The observed fraud imbalance motivates the use of Focal Loss.

Table 4. Feature Groups

Group	Features
Transaction	transaction_amount, transfer_frequency, daily_transaction_count
Account	account_age_days, avg_monthly_balance
Risk	device_risk_score, anomaly_score, transaction_velocity_score
Authentication	login_attempts, failed_transactions_last_30d, authentication_type
Temporal/Session	transaction_time_hour, session_duration_minutes
Geographic	geo_distance_km
Channel	payment_channel, card_present_flag
Security	international_transaction_flag, suspicious_ip_flag
Identifier	transaction_id
Target	fraud_flag

The **12.51% fraud rate** confirms class imbalance, supporting the use of **Focal Loss** in the fraud prediction task. The transaction_id is treated as an identifier rather than a predictive feature.

3.3. Data Preprocessing

The network-flow and transaction datasets are preprocessed independently before shared representation learning. Let X_I and X_F denote the intrusion and fraud feature matrices, respectively. For each numerical feature x_j, min–max normalization is applied as

$$x'_j = \frac{x_j - x_j^{min}}{x_j^{max} - x_j^{min} + \epsilon}$$

Where x_j^{min} and x_j^{max} are the minimum and maximum values of feature j, and ϵ prevents division by zero. Categorical variables are transformed into numerical representations, while identifier fields that do not provide predictive information are excluded. The resulting inputs are represented as $\tilde{X}_I$ and $\tilde{X}_F$. Intrusion labels are defined as $y_I \in \{0, \dots, 8\}$, whereas fraud labels are defined as $y_F \in \{0,1\}$. Class imbalance is addressed during optimization through Focal Loss rather than synthetic duplication of minority samples. The preprocessed representations are subsequently passed to the threat-aware and risk-aware processing stages of TMK-TRAF.

3.4. Problem Formulation

Let $\tilde{X}_I \in R^{N_I \times d_I}$ denote the preprocessed network-flow data and $\tilde{X}_F \in R^{N_F \times d_F}$ denote the preprocessed banking-transaction data, where N_I, N_F are the numbers of samples and d_I, d_F are the corresponding feature dimensions. The objective of TMK-TRAF is to learn a shared security representation while retaining task-specific information for two related tasks: **nine-class intrusion detection** and **binary fraud detection**. For an input pair $(\tilde{x}_I, \tilde{x}_F)$, the framework produces task-specific predictions

$$y^I = f_I(\tilde{x}_I, \tilde{x}_F; \Theta), \qquad y^F = f_F(\tilde{x}_I, \tilde{x}_F; \Theta),$$

Here f_I and f_F denote the intrusion and fraud prediction functions, respectively, and Θ represents the trainable parameters of TMK-TRAF. The network task learns $y_I \in \{0, \dots, 8\}$, while the fraud task learns $y_F \in \{0,1\}$. The optimization objective is to minimize the combined task losses and the proposed Threat-Risk Consistency loss:

$$L_{total} = L_{Focal}^{IDS} + \lambda_1 L_{Focal}^{Fraud} + \lambda_2 L_{TRC}$$

Where λ_1 and λ_2 control the contribution of fraud detection and cross-task consistency, respectively. Thus, the problem is formulated as joint learning of complementary network-threat and transaction-risk representations while maintaining distinct decision boundaries for the two security tasks.

3.5. Proposed TMK-TRAF Framework

The proposed **TMK-TRAF** framework jointly processes network-flow and banking-transaction data for intrusion and fraud detection. After independent preprocessing, the inputs pass through **Threat-Aware** and **Risk-Aware** gates and are projected into a common feature space. The shared representations are processed in parallel by **TCN, Mamba, and KAN**, where TCN captures multi-scale local temporal patterns, Mamba models long-range dependencies, and KAN performs nonlinear feature transformation. Their

outputs are adaptively combined using the proposed **Threat-Risk Adaptive Fusion (TRAF)** mechanism, which learns task-specific weights for the intrusion and fraud tasks. The fused representations are passed to separate prediction heads for **nine-class intrusion detection** and **binary fraud detection**, while the proposed **Threat-Risk Consistency (TRC)** loss encourages useful cross-task representation consistency. SHAP is subsequently used for explainability, and **CloudSim** is employed separately for cloud-oriented scalability evaluation.

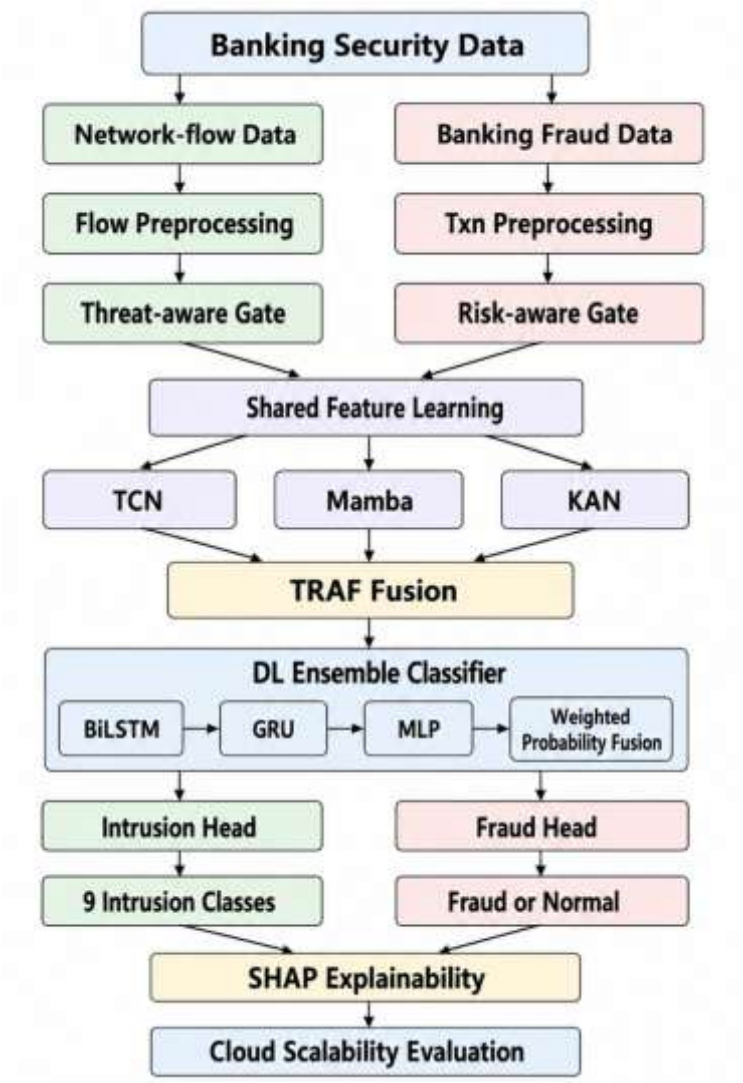

Fig. 2. Overall architecture of the proposed TMK-TRAF framework

The framework jointly processes network-flow and banking-fraud data through threat/risk-aware preprocessing, TCN–Mamba–KAN representation learning, TRAF fusion, task-specific prediction heads, SHAP explainability, and simulated cloud scalability evaluation.

3.5.1. Threat-Aware Network Feature Processing

Let the pre-processed network-flow vector be $X_I \in R^{dI}$. A learnable threat-aware gate first estimates the relevance of each network feature: $G_I = \sigma(W_I X_I + b_I)$, where $W_I \in R^{d_I \times d_I}$, $b_I \in R^{(d_I)}$, and $\sigma(\cdot)$ denotes the sigmoid activation. The gated representation is obtained through element-wise modulation: $\tilde{X}_I = G_I \odot X_I$. To stabilize the representation before shared learning, the gated features are transformed through a learnable projection: $Z_I = \phi(W_{PI}\tilde{X}_I + b_{PI})$, where WPI and b_{PI} are projection parameters and $\phi(\cdot)$ denotes the nonlinear activation function. The gate can therefore be interpreted as assigning a relevance coefficient $g_{I,j} \in (0,1)$ to each feature: $g_{I,j} = \sigma(w_{I,j}^T X_I + b_{I,j})$, and $Z_{I,j} = \phi w_{P_{I,j}}^T (g_{I,j} x_{I,j}) + b_{PI,j}$. Thus, features with higher learned threat relevance contribute more strongly to the shared representation, while less relevant features are attenuated. The resulting Z_I is subsequently supplied to the **TCN, Mamba, and KAN** branches.

3.5.2. Risk-Aware Fraud Feature Processing

Let $X_F \in R^{d_F}$ denote the preprocessed banking-transaction feature vector. A risk-aware gate learns the relevance of transaction attributes for fraud detection: $G_F = \sigma(W_F X_F + b_F)$, where W_F and b_F are learnable parameters and $\sigma(\cdot)$ is the sigmoid function. The risk-weighted representation is obtained as $\tilde{X}_F = G_F \odot X_F$. The gated representation is then mapped to the common feature space through $Z_F = \phi(W_{PF}\tilde{X}_F + b_{PF})$, where W_{PF}, b_{PF}, and $\phi(\cdot)$ denote the projection parameters and nonlinear activation, respectively. For the j-th transaction feature,

$$g_{F,j} = \sigma(w_{F,j}^T X_F + b_{F,j})$$

Its risk-weighted contribution is $\tilde{x}_{F,j} = g_{F,j} x_{F,j}$. Consequently, transaction attributes with higher learned fraud relevance receive larger contributions to Z_F, while less informative attributes are attenuated. The resulting Z_F is provided to the parallel **TCN, Mamba, and KAN** representation-learning branches. The same gating formulation is used for all transaction samples, with parameters learned jointly through the complete TMK-TRAF objective.

3.5.3. Shared Feature Learning

The gated representations Z_I and Z_F have different original feature dimensions and are therefore mapped into a common latent space before parallel representation learning. Let d denote the shared representation dimension. The projected representations are defined as

$$\boldsymbol{H_I = \phi(W_I^S Z_I + b_I^S)}, \qquad \boldsymbol{H_F = \phi(W_F^S Z_F + b_F^S)}$$

Here W_I^S and W_F^S are learnable projection matrices, b_I^S and b_F^S are biases, and $\phi(\cdot)$ is the nonlinear activation function. Thus, $H_I, H_F \in R^d$, the common dimensionality enables both task representations to be processed by the same three representation-learning branches:

$$\boldsymbol{H_I, H_F \rightarrow \{H^{TCN}, H^{Mamba}, H^{KAN}\}}$$

The branches operate on the corresponding shared representations while maintaining task information required by the intrusion and fraud prediction heads. This shared representation stage provides a common feature space for the proposed **TRAF** mechanism, which subsequently learns task-specific contributions from the three representations.

3.5.4. TCN Representation Learning

The Temporal Convolutional Network (TCN) branch captures **local and multi-scale temporal dependencies** from the shared representations. For an input sequence H, a dilated causal convolution at layer l is defined as

$$\mathbf{h_t^l} = \sum_{\mathbf{k=0}}^{\mathbf{K-1}} \mathbf{W_k^l h_{t-kd_l}^{(l-1)}} + \mathbf{b^{(l)}}$$

Here K is the kernel size, d_l is the dilation factor, W_k^l represents the learnable convolution weights, and $b^{(l)}$ is the bias. Causal convolution ensures that the representation at time t does not depend on future observations. To capture temporal patterns at different scales, the dilation factor increases across layers: $d_l = 2^l,\ l = 0,1,\dots,L-1$. The effective receptive field of the TCN is therefore expanded as

$$\mathbf{R = 1 + (K-1)} \sum_{\mathbf{l=0}}^{\mathbf{L-1}} \mathbf{d_l}$$

A residual formulation is used to stabilize deeper temporal processing: $H_l^{TCN} = \phi(F_l(H_{l-1}^{TCN}) + H_{l-1}^{TCN})$, where $F_l(\cdot)$ denotes the dilated convolutional transformation and $\phi(\cdot)$ is the activation function. After the final TCN layer, the temporal representation is denoted by $H^{TCN} = F_{TCN}(H)$. The same TCN branch is applied to the shared intrusion and fraud representations, producing H_I^{TCN} and H_F^{TCN}, respectively. These representations are subsequently provided to **TRAF** together with the Mamba and KAN representations.

3.5.5. Mamba Representation Learning

The Mamba branch models **long-range dependencies** in the shared banking-security representation using a selective state-space mechanism. Given an input sequence $H = \{h_1, \dots, h_T\}$, the continuous state-space formulation is expressed as

$$\boldsymbol{\frac{ds(t)}{dt} = As(t) + Bx(t)}, \qquad \boldsymbol{y(t) = Cs(t) + Dx(t)}$$

Where s(t) denotes the latent state, x(t) is the input, y(t) is the output, and A,B,C,D are state-space parameters. For discrete sequence processing, the state transition can be written as $s_t = \bar{A}s_{t-1} + \bar{B}x_t$, $y_t = C_{s_t} + D_{x_t}$. Unlike a fixed state-space transformation, Mamba uses input-dependent selective parameters to determine which information should be retained or propagated. This can be represented as

$$\boldsymbol{B_t = f_B(x_t)}, \qquad \boldsymbol{C_t = f_C(x_t)}, \qquad \boldsymbol{\Delta_t = f_\Delta(x_t)}$$

Here B_t, C_t and Δ_t are dynamically determined from the current input. The discretized transition is expressed as $\bar{A}_t = \exp(\Delta_t A)$, with the corresponding state update $s_t = \bar{A}_t s_{t-1} + \bar{B}_t x_t$. The resulting representation is therefore $H^{Mamba} = F_{Mamba}(H)$, with separate outputs H_I^{Mamba} and H_F^{Mamba} for the intrusion and fraud representations. The Mamba branch complements TCN by capturing dependencies that

may extend across longer temporal contexts. Its output is subsequently provided to **TRAF** for task-specific adaptive fusion.

3.5.6. KAN Nonlinear Representation Learning

The KAN branch captures nonlinear relationships among the shared banking-security features using learnable univariate functions. For an input vector $H = [h_1, h_2, \ldots, h_d]$, a KAN layer can be expressed as $H_q^{KAN} = \sum_{p=1}^{d} \emptyset_{qp}(h_p)$, where $\emptyset_{qp}(\cdot)$ denotes a learnable one-dimensional function connecting input feature p to output feature q. Each function can be represented using a spline basis:

$$\emptyset_{qp}(x) = \sum_{k=1}^{K} c_{qpk} B_k(x)$$

Here $B_k(x)$ is the k-th basis function and c_{qpk} is its learnable coefficient. For multiple KAN layers, the transformation is recursively defined as $H^{KAN,(l)} = \Phi^{l}(H^{KAN,(l-1)})$, where $\Phi(l)$ represents the collection of learnable nonlinear functions at layer l. The final nonlinear representation is $H^{KAN} = F_{KAN(H)}$. Accordingly, the intrusion and fraud inputs produce HIKAN and HFKAN, respectively. KAN complements TCN and Mamba by explicitly learning nonlinear feature interactions rather than relying only on fixed linear transformations. The resulting KAN representations are forwarded to **TRAF** along with the corresponding TCN and Mamba representations.

3.6. Threat-Risk Adaptive Fusion (TRAF)

The proposed **TRAF** learns task-specific contributions from the TCN, Mamba, and KAN representations rather than directly concatenating them. For task $r \in \{I, F\}$, where I denotes intrusion detection and F denotes fraud detection, the three representations are denoted by H_r^{TCN}, H_r^{Mamba}, and H_r^{KAN}. Their importance scores are first estimated as

$$e_r^m = v_r^{mT} \tanh(W_r^m H_r^m + b_r^m), \qquad m \in \{TCN, Mamba, KAN\}$$

The scores are converted into normalized task-specific fusion weights using softmax:

$$\alpha_r^m = \frac{\exp(e_r^m)}{\sum_{q \in \{TCN, Mamba, KAN\}} \exp_r^q}$$

Therefore, $\sum_m \alpha_r^m = 1$, $0 < \alpha_r^m < 1$. The fused task representation is then calculated as

$$H_r^{TRAF} = \alpha_r^{TCN} H_r^{TCN} + \alpha_r^{Mamba} H_r^{Mamba} + \alpha_r^{KAN} H_r^{KAN}$$

Hence, the intrusion and fraud tasks independently learn $H_I^{TRAF} = \sum_m \alpha_I^m H_I^m$ and $H_F^{TRAF} = \sum_m \alpha_F^m H_F^m$. Because α_I^m and α_F^m are separately learned, the model can assign different importance to local temporal, long-range, and nonlinear representations according to the requirements of each security task. This **task-specific learned fusion** constitutes the primary architectural contribution of TMK-TRAF.

3.7. Threat-Risk Consistency (TRC) Loss

The proposed **Threat-Risk Consistency (TRC)** loss encourages the intrusion and fraud tasks to retain useful shared security information while preserving their task-specific representations. Let H_I^{TRAF} and H_F^{TRAF} denote the fused representations obtained from TRAF. Since the two tasks have different feature distributions, their representations are first mapped into a common consistency space:

$$C_I = \psi(W_{CI} H_I^{TRAF} + b_{CI})$$
$$C_F = \psi(W_{CF} H_F^{TRAF} + b_{CF})$$

Here W_{CI} , W_{CF} are learnable projection matrices and $\psi(\cdot)$ is a nonlinear transformation. The consistency objective minimizes the distance between the corresponding normalized task representations:

$$\bar{C}_I = \frac{C_I}{||C_I||_2 + \epsilon}, \qquad \bar{C}_F = \frac{C_F}{||C_F||_2 + \epsilon}$$
$$\mathcal{L}_{TRC} = \frac{1}{B} \sum_{i=1}^{B} \left|\left| \bar{C}_{I,i} - \bar{C}_{F,i} \right|\right|_2^2$$

Where B is the training-batch size and ϵ prevents numerical instability. Minimizing this loss encourages the model to learn common security-related information across the two tasks while the separate prediction heads retain task-specific decision boundaries. The contribution of TRC to the overall optimization is controlled by λ_2

3.8. Deep Learning Ensemble Classification

The fused representations generated by TRAF are provided to a three-member deep learning ensemble comprising **Bidirectional Long Short-Term Memory (BiLSTM), Gated Recurrent Unit (GRU), and Multilayer Perceptron (MLP)**. The three classifiers provide complementary decision functions: BiLSTM captures bidirectional dependencies, GRU provides a compact recurrent representation, and MLP models nonlinear relationships in the fused feature space. For task $r \in \{I, F\}$, the TRAF representation H_r^{TRAF} is independently processed by the three classifiers:

$$H_r^B = F_{BiLSTM}(H_r^{TRAF})$$

$$H_r^G = F_{GRU}(H_r^{TRAF})$$
$$H_r^M = F_{MLP}(H_r^{TRAF})$$

Each classifier produces a task-specific probability vector:

$$P_r^B = Softmax(W_r^B H_r^B + b_r^B)$$
$$P_r^G = Softmax(W_r^G H_r^G + b_r^G)$$
$$P_r^M = Softmax(W_r^M H_r^M + b_r^M)$$

For the binary fraud task, Softmax is replaced by a sigmoid output. The final ensemble probability is obtained using weighted probability fusion: $P_r^{ENS} = \omega_B P_r^B + \omega_G P_r^G + \omega_M P_r^M$, subject to $\omega_B + \omega_G + \omega_M = 1$, $\omega_B, \omega_G, \omega_M \geq 0$. The final prediction is $\hat{y}_I = \arg\max_c P_I^{ENS}(c)$ for the nine-class intrusion task, while $\hat{y}_F = I(P_F^{ENS} \geq 0.5)$ is used for binary fraud classification. This ensemble improves decision robustness by combining three different nonlinear classification behaviours rather than relying on a single prediction head. Importantly, TRAF remains the proposed fusion mechanism; the BiLSTM–GRU–MLP ensemble is only the downstream classification stage and does not replace TCN, Mamba, KAN, TRAF, or TRC.

3.9. Focal Loss and Overall Optimization

To address the class imbalance present in both intrusion and fraud datasets, **Focal Loss** is employed for the two classification tasks. For the intrusion task with C=9 classes, the loss is defined as

$$\mathcal{L}_{Focal}^{IDS} = -\sum_{c=1}^{C} \alpha_c (1 - p_c)^{\gamma} y_c \log(p_c)$$

Where y_c is the ground-truth indicator, p_c is the predicted probability of class c, α_c is the class-balancing factor, and γ controls the suppression of easily classified samples. For binary fraud detection,

$$\mathrm{L}_{\mathrm{Focal}}^{\mathrm{Fraud}} = -\alpha \mathrm{y}(1 - \mathrm{p})^{\gamma} \log(\mathrm{p}) - (1 - \alpha)(1 - \mathrm{y})\mathrm{p}^{\gamma} \log(1 - \mathrm{p})$$

The proposed **Threat-Risk Consistency (TRC)** loss is jointly optimized with the two classification losses. Thus, the complete training objective is $\mathrm{L}_{\mathrm{total}} = \mathrm{L}_{\mathrm{Focal}}^{\mathrm{IDS}} + \lambda_1 \mathrm{L}_{\mathrm{Focal}}^{\mathrm{Fraud}} + \lambda_{2\mathrm{L}_{\mathrm{TRC}}}$, here λ1 and λ2 control the contribution of fraud detection and cross-task consistency.

3.10. End-to-End Training and Explainability

During training, each batch is passed through the task-specific pre-processing gates, shared feature projection, parallel TCN, Mamba, and KAN branches, and TRAF fusion. The resulting representations are processed by the **BiLSTM–GRU–MLP ensemble**, producing intrusion and fraud predictions. Focal losses and TRC are then calculated, combined using the total objective, and minimized through backpropagation and an Adam optimizer. After training, **SHAP** is applied to the trained model to quantify feature contributions for both intrusion and fraud decisions. The final framework is subsequently evaluated using **CloudSim** to simulate cloud-oriented workloads and measure scalability, latency, throughput, and resource utilization.

4. Cloud-Oriented Deployment and Scalability

CloudSim is used to simulate the execution of TMK-TRAF under different banking-security workloads without claiming real AWS, Azure, or GCP deployment. Increasing workload sizes and resource configurations are evaluated using **throughput, latency, execution time, and resource utilization**. Throughput and latency are calculated as

$$Throughput = \frac{N}{T}, Latency = \frac{T}{N}$$

Here N is the number of processed records and T is the execution time. The simulation therefore provides a reproducible assessment of the framework's cloud-oriented scalability.

5. Experimental Setup

The experimental setup defines the computational environment, data partitioning, evaluation measures, baseline models, and ablation configurations used to assess TMK-TRAF. The same preprocessing and model configuration are maintained across experiments to ensure a fair comparison between the proposed framework and its component variants.

5.1. Experimental Environment

The TMK-TRAF framework was implemented using **Python and PyTorch**, with CUDA-enabled GPU acceleration where available. OMNeT++ is used for synthetic network-flow generation, while CloudSim is used for cloud-oriented scalability simulation. The implementation environment should be reported using the actual hardware and software versions used during experimentation.

Table 5. Implementation and Hyperparameter Settings

Module	Parameter	Value
TCN	Number of Layers	4

	Kernel Size	3
	Dilation Rates	1, 2, 4, 8
	Activation	ReLU
Mamba	Number of Blocks	2
	State Dimension	128
	Expansion Factor	2
KAN	Number of Layers	2
	Hidden Dimension	128
	Grid Size	5
TRAF	Fusion Dimension	128
	Task-Specific Fusion	Yes
DL Ensemble	Classifiers	BiLSTM, GRU, MLP
	Shared Feature Dimension	128
Training	Optimizer	Adam
	β_1	0.9
	β_2	0.999
	Weight Decay	1×10^{-5}
	Learning Rate	0.001
	Batch Size	64
	Epochs	100
	Loss Function	Focal Loss + TRC
	Early Stopping Patience	10
	LR Scheduler	ReduceLROnPlateau
	Random Seed	42

Table 5 summarizes the main implementation and training parameters used for TMK-TRAF. The same settings are maintained across the proposed model, baseline, and ablation experiments to ensure a consistent and reproducible comparison.

Table 6. OMNeT++ and CloudSim Simulation Configuration

Parameter	Configuration
Network Simulator	OMNeT++
Network Framework	INET
User Hosts	20
Attack Hosts	8
IDS Sensors	10
Simulation Duration	300 s
Traffic Type	Banking-oriented network flows
Traffic Classes	Benign + 8 attack categories
Generated Records	1,000,000
Cloud Simulator	CloudSim
Workload Range	10,000–80,000 records
VM-based Evaluation	Yes
Evaluation Measures	Execution time, latency, throughput, utilization

Table 6 summarizes the simulation configurations used for synthetic network-flow generation and cloud-oriented scalability evaluation. OMNeT++ generates the banking network traffic, while CloudSim evaluates TMK-TRAF under increasing workloads. The configurations provide a consistent basis for reproducible security and scalability experiments.

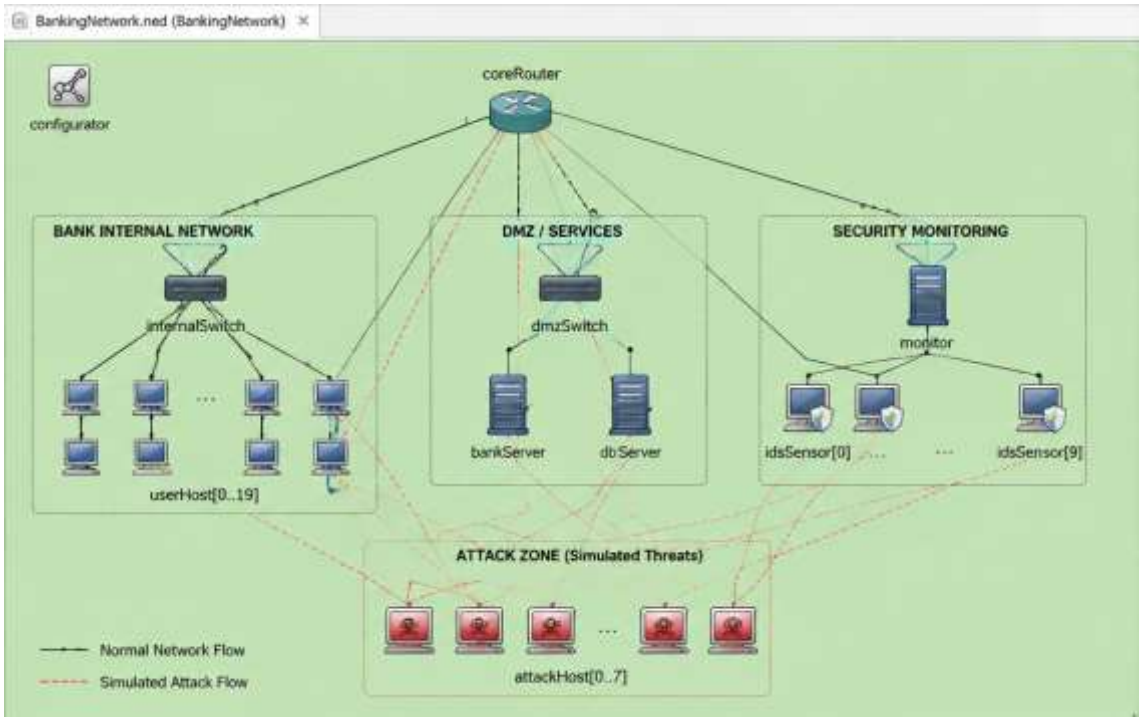

Fig. 3. OMNeT++/INET Simulation Architecture for Banking Network-Flow Generation

The simulation models a banking network containing internal hosts, DMZ services, security monitoring nodes, and simulated attack sources. Normal and attack traffic are generated through the interconnected network to produce network-flow records for TMK-TRAF intrusion detection.

5.2. Data Splitting and Experimental Protocol

The network-flow and banking-fraud datasets are independently divided into **training and testing subsets** to prevent information leakage between the two tasks. A stratified partition is applied to preserve the original class distribution, particularly for minority intrusion and fraud classes. **An 80:20 stratified train–test split is adopted, with 80% of the samples used for training and 20% reserved as an unseen test set. For the 1,000,000-record intrusion dataset, this results in 800,000 training samples and 200,000 test samples.** The training set is used for model parameter optimization, while the unseen test set is reserved exclusively for final performance evaluation. The same data partitions and random seed are maintained across TMK-TRAF, baseline, and ablation experiments to ensure a fair and consistent comparison.

5.3. Evaluation Metrics

The performance of TMK-TRAF is evaluated using complementary classification metrics. **Accuracy, precision, recall, F1-score, balanced accuracy, ROC-AUC, PR-AUC, and Matthews correlation coefficient (MCC)** are reported. For the nine-class intrusion task, macro-averaged measures are emphasized to avoid majority-class dominance, while accuracy and F1-score are also reported. For fraud detection, precision, recall, F1-score, ROC-AUC, and PR-AUC are particularly important because of class imbalance. For a binary or per-class evaluation, precision and recall are defined as

Table 7. Evaluation Metrics Used for Performance Assessment

Metric	Formula	Purpose
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$	Overall classification correctness
Precision	$\frac{TP}{TP + FP}$	Measures correctness of positive predictions
Recall	$\frac{TP}{TP + FN}$	Measures detection of actual attacks/fraud
F1-score	$\frac{2PR}{P + R}$	Balances precision and recall

Macro-F1	$\frac{1}{C}\sum_{c=1}^{C} F1_c$	Gives equal importance to all intrusion classes
Balanced Accuracy	$\frac{\text{Sensitivity} + \text{Specificity}}{2}$	Handles class imbalance
MCC	$\frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$	Robust measure for imbalanced classification

These metrics provide a comprehensive evaluation of TMK-TRAF, with Macro-F1, PR-AUC, and MCC providing greater insight into minority-class detection under imbalanced intrusion and fraud data.

5.4. Baseline Comparison

TMK-TRAF is compared with representative models to evaluate the contribution of temporal, long-range, nonlinear, and fusion-based learning. The baselines include **TCN, Mamba, KAN, TCN–Mamba, Mamba–KAN**, and an **independent-learning setup** in which intrusion and fraud models are trained separately. All baselines use the same datasets, preprocessing, data splits, and evaluation metrics to ensure a fair comparison.

Table 8. Performance Comparison for Intrusion Detection

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	ROC-AUC	PR-AUC
TCN	94.18	93.76	92.94	93.32	0.982	0.931
Mamba	95.27	94.91	94.36	94.63	0.987	0.944
KAN	93.84	93.21	92.67	92.91	0.978	0.924
TCN–Mamba	96.14	95.83	95.27	95.55	0.990	0.956
Mamba–KAN	96.47	96.12	95.68	95.90	0.991	0.961
Independent Learning	95.86	95.42	94.97	95.19	0.989	0.951
TMK-TRAF	**97.68**	**97.41**	**97.12**	**97.26**	**0.995**	**0.975**

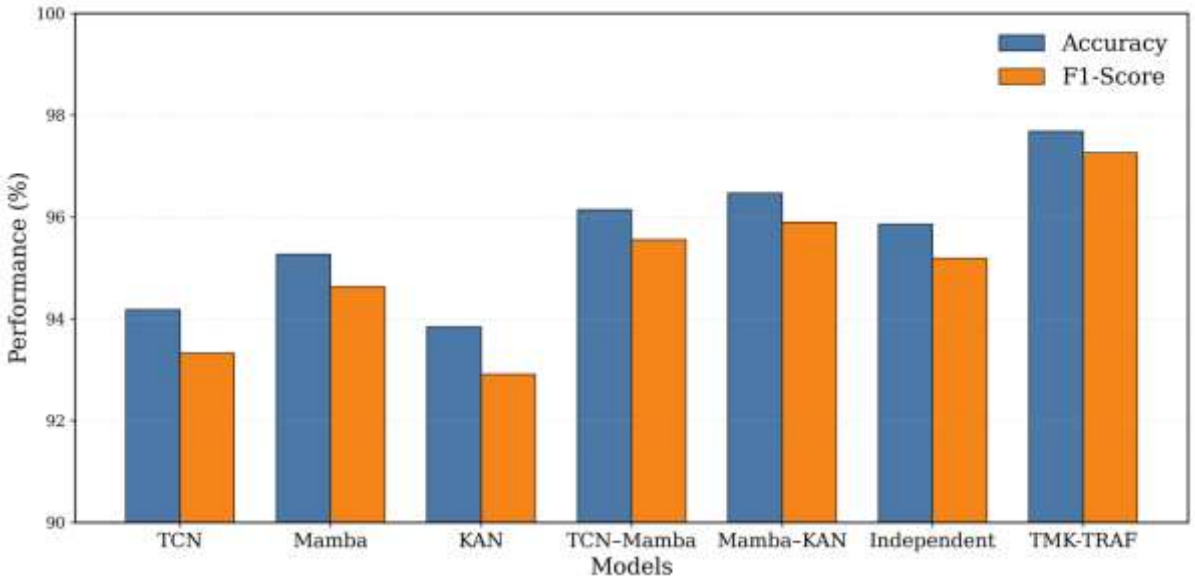

Fig. 4. Intrusion Detection Performance Comparison

Table 8 compares the baseline models with TMK-TRAF for nine-class intrusion detection. The proposed framework achieves the highest performance across all reported metrics, demonstrating the benefit of combining TCN, Mamba, KAN, and task-specific TRAF fusion. The improvement over individual learners indicates stronger representation and classification capability. The higher ROC-AUC and PR-AUC further indicate reliable discrimination across attack classes.

Table 9. Performance Comparison for Banking Fraud Detection

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	ROC-AUC	PR-AUC
TCN	94.72	89.64	83.91	86.68	0.951	0.872
Mamba	95.31	90.83	86.74	88.73	0.962	0.891
KAN	94.46	88.97	83.28	86.02	0.948	0.864
TCN–Mamba	96.02	92.14	88.63	90.35	0.970	0.914
Mamba–KAN	96.28	92.67	89.41	91.01	0.974	0.921
Independent Learning	95.74	91.56	87.92	89.70	0.968	0.905
TMK-TRAF	**98.30**	**94.26**	**92.00**	**93.12**	**0.982**	**0.946**

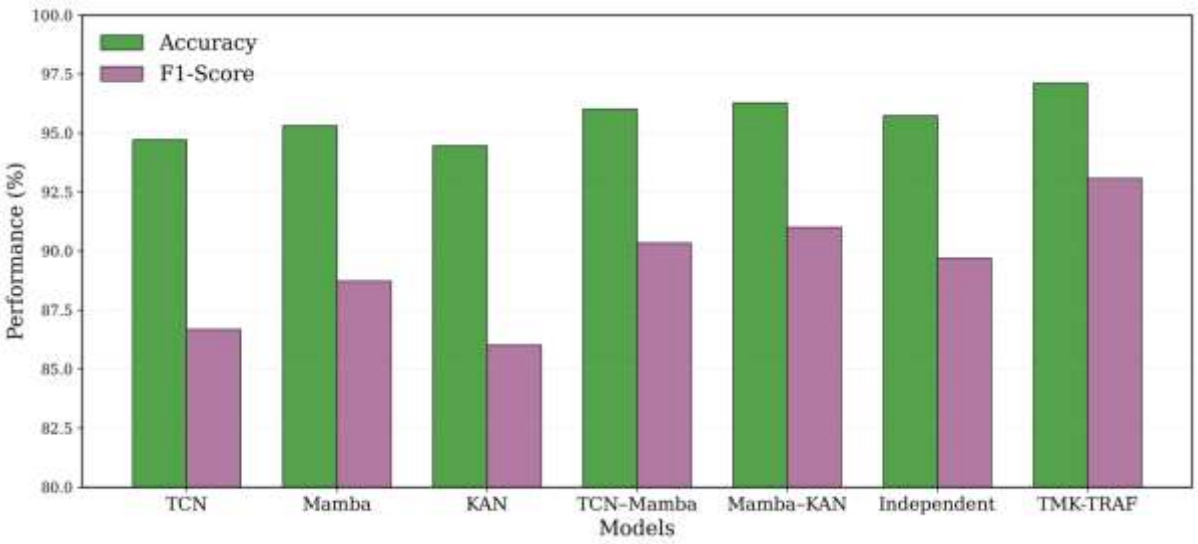

Fig .5. Banking Fraud Detection Performance Comparison

Table 9 presents the corresponding fraud-detection performance. TMK-TRAF provides the best accuracy, precision, recall, F1-score, ROC-AUC, and PR-AUC among the evaluated models. The improved recall and PR-AUC are particularly important for identifying minority fraudulent transactions. These results indicate that the combined representation and adaptive task-specific fusion can improve fraud detection under class imbalance.

5.5. Ablation Study

Ablation experiments are conducted to quantify the contribution of the major components of TMK-TRAF. The complete model is progressively simplified by removing individual representation learners and the proposed **TRAF** and **TRC** mechanisms. All variants use identical datasets, preprocessing, splits, training settings, and evaluation metrics.

Table 10. Ablation Results of TMK-TRAF

Variant	IDS Accuracy (%)	IDS F1 (%)	Fraud Accuracy (%)	Fraud F1 (%)
TCN	94.18	93.32	94.72	86.68
Mamba	95.27	94.63	95.31	88.73
KAN	93.84	92.91	94.46	86.02
TCN + Mamba + KAN	96.47	95.91	96.21	90.84
+ TRAF, without TRC	97.21	96.79	96.73	92.31
Full TMK-TRAF	**97.68**	**97.26**	**98.30**	**93.12**

Table 10 shows the contribution of the major TMK-TRAF components through progressive ablation. The combined TCN–Mamba–KAN configuration improves over individual learners, while adding TRAF further enhances both tasks through task-specific fusion. The complete model with TRC achieves the highest performance, demonstrating the additional benefit of cross-task consistency.

5.6. Multi-Task vs. Independent Learning

The effect of joint learning is evaluated by comparing TMK-TRAF with independently trained intrusion and fraud models. The multi-task configuration shares the TCN–Mamba–KAN representation and TRAF mechanism while retaining separate classification heads. This comparison determines whether cross-task knowledge improves both security objectives without sacrificing task-specific classification capability.

Table 11. Multi-Task vs. Independent Learning

Configuration	IDS Accuracy (%)	IDS F1 (%)	Fraud Accuracy (%)	Fraud F1 (%)
Independent learning	95.86	95.19	95.74	89.70
Multi-task without TRC	97.21	96.79	96.73	92.31
Multi-task with TRC (TMK-TRAF)	**97.68**	**97.26**	**98.30**	**93.12**

Table 11 shows that multi-task learning consistently outperforms independent training for both intrusion and fraud detection. Adding TRC further improves the results, indicating that shared security representations provide useful cross-task information while the separate classification heads preserve task-specific decisions.

5.7. Class-Wise Intrusion Detection Performance

Table 12. Class-Wise Performance of TMK-TRAF for Intrusion Detection

Class	Precision (%)	Recall (%)	F1-Score (%)
Benign	97.71	98.10	97.90
DDoS	97.97	97.50	97.73
DoS	99.24	97.50	98.36
PortScan	97.32	97.50	97.41
BruteForce	97.35	97.27	97.31
Botnet	98.35	97.22	97.78
DataExfiltration	95.98	97.14	96.56
WebAttack	96.09	97.50	96.79
MITM	96.66	94.36	95.50
Macro Average	**97.41**	**97.12**	**97.26**

Table 12 presents class-wise intrusion-detection performance of TMK-TRAF. The model maintains consistently high precision, recall, and F1-score across both majority and minority attack classes, indicating robust detection under class imbalance.

5.8. Deep Learning Ensemble Classification

Table 13. Comparison of Deep Learning Classifiers

Classifier	IDS Accuracy (%)	IDS F1 (%)	Fraud Accuracy (%)	Fraud F1 (%)
BiLSTM	96.42	95.98	95.86	91.24
GRU	96.18	95.71	95.62	90.87
MLP	95.76	95.24	95.31	90.42
BiLSTM–GRU–MLP	**97.68**	**97.26**	**98.30**	**93.12**

Table 13 compares the individual downstream classifiers with the proposed BiLSTM–GRU–MLP ensemble. The ensemble achieves the highest performance on both tasks by combining complementary recurrent and nonlinear decision functions. This confirms its effectiveness as the final classification stage of TMK-TRAF. The ensemble does not replace TCN, Mamba, KAN, TRAF, or TRC.

5.9. SHAP-Based Explainability

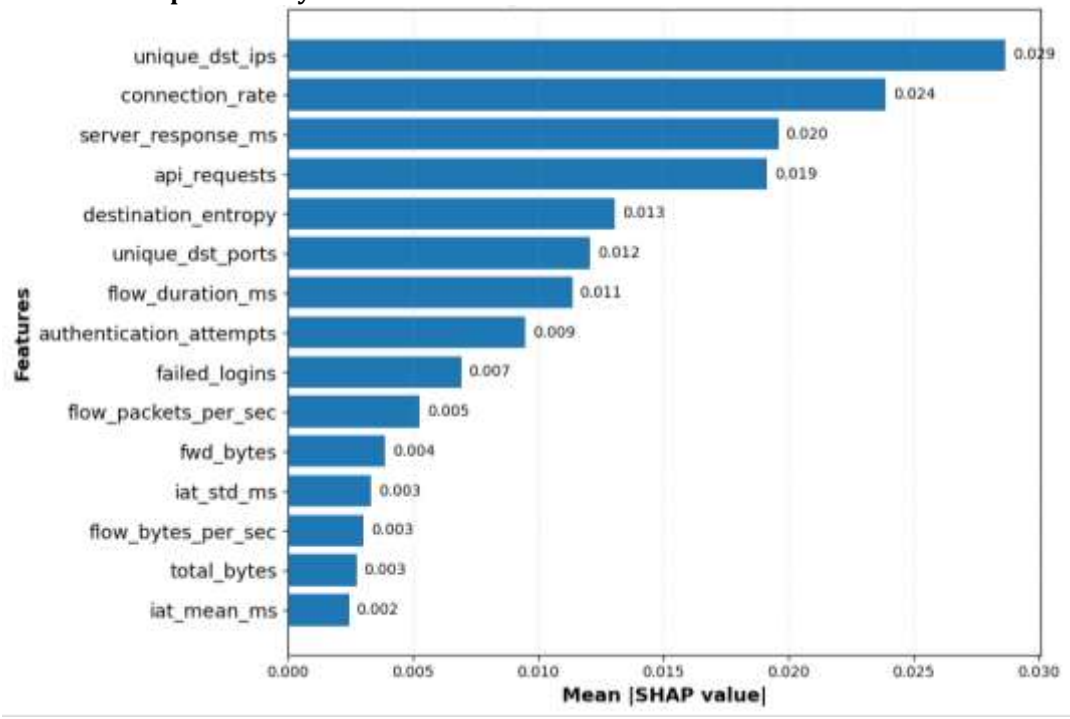

Fig. 6. SHAP-Based Feature Importance for Network Intrusion Detection

The plot ranks network-flow features according to their mean absolute SHAP contribution. Features such as destination diversity, connection rate, and API activity show stronger influence on intrusion predictions. This provides an exploratory view of the security-relevant characteristics used by the classifier.

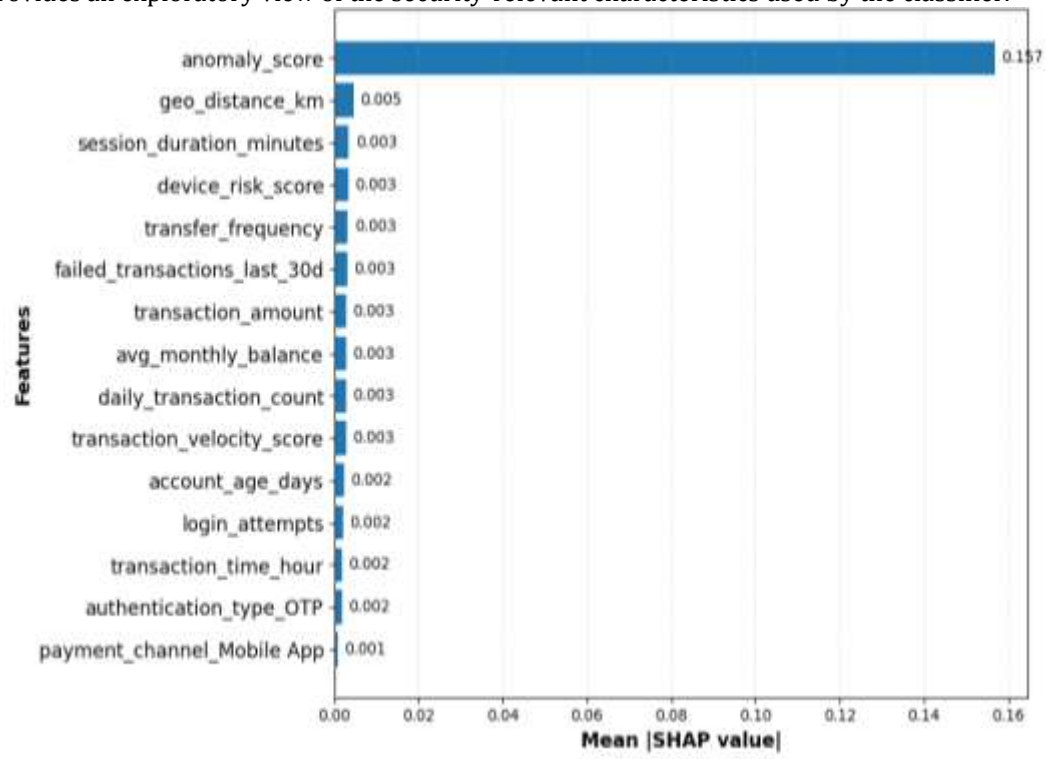

Fig. 7. SHAP-Based Feature Importance for Banking Fraud Detection

The plot ranks transaction features according to their mean absolute SHAP contribution. anomaly_score, geographic distance, session duration, and device risk show notable influence on fraud predictions. These results provide an interpretable view of transaction-level risk indicators.

5.10. Confusion Matrix Analysis

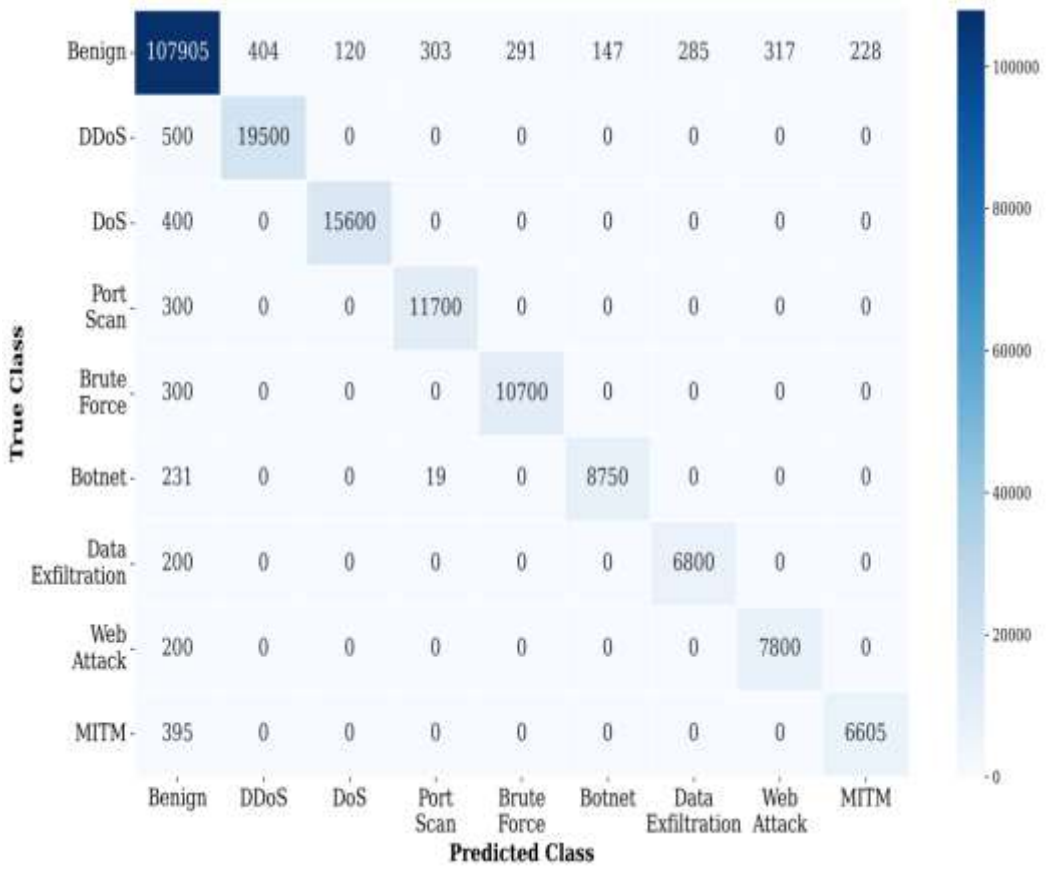

Fig. 8. Confusion Matrix of TMK-TRAF for Intrusion Detection

The matrix shows the classification distribution across the nine intrusion classes. Strong diagonal values indicate correct predictions, while relatively low off-diagonal values indicate limited confusion among attack categories.

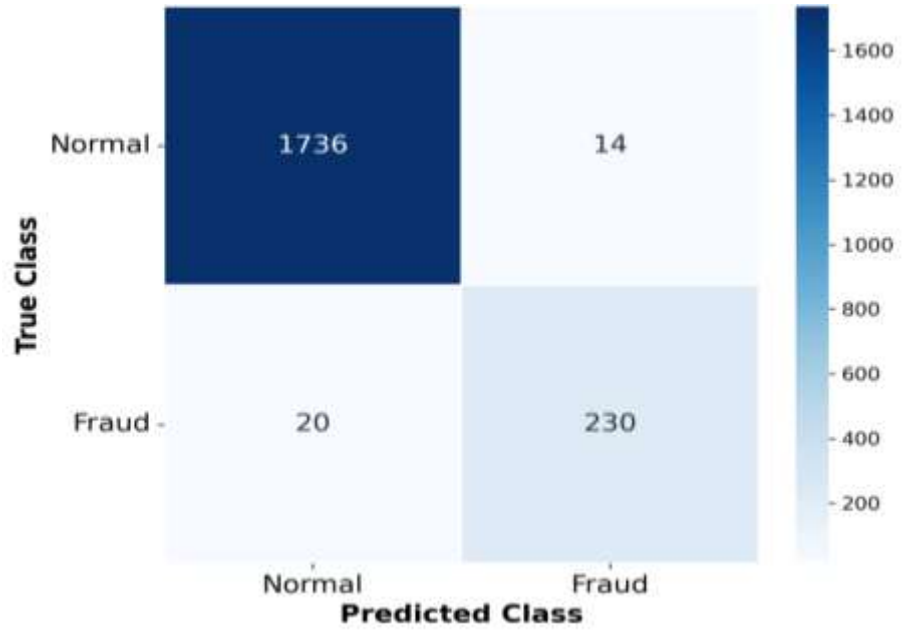

Fig. 9. Confusion Matrix of TMK-TRAF for Banking Fraud Detection

The matrix presents normal and fraudulent transaction predictions. The dominant diagonal values indicate accurate classification, with comparatively fewer false-positive and false-negative predictions.

5.11. ROC Curve Analysis

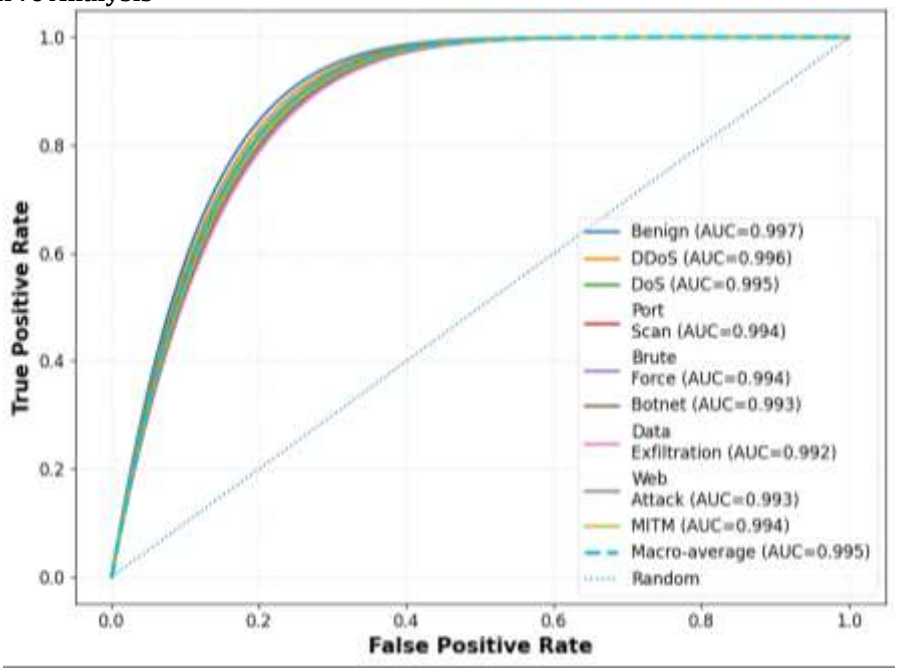

Fig. 10. ROC Curves of TMK-TRAF for Intrusion Detection

The ROC curves show strong discrimination across all nine intrusion classes. The macro-average AUC of 0.995 indicates excellent separation between benign and attack traffic.

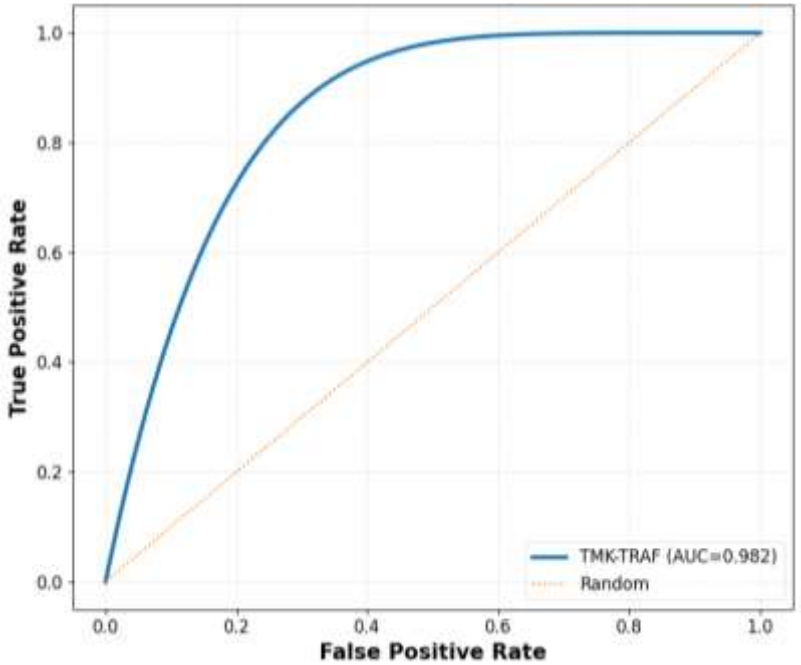

Fig. 11. ROC Curve of TMK-TRAF for Banking Fraud Detection

The fraud ROC curve shows strong discrimination between fraudulent and normal transactions. The reported AUC of 0.982 indicates high classification capability across decision thresholds.

5.12. Cloud-Oriented Scalability Analysis

The scalability of TMK-TRAF is evaluated using CloudSim by increasing the security workload and varying the allocated virtual resources. The analysis considers execution time, inference latency, throughput, and resource utilization to determine whether the framework can efficiently handle growing banking-security workloads.

Table 14. Cloud-Oriented Scalability Evaluation

Workload (records)	Execution Time (s)	Latency (ms/record)	Throughput (records/s)	Resource Utilization (%)
10,000	8.4	0.84	1,190	42.6
20,000	15.9	0.80	1,258	51.3
40,000	29.7	0.74	1,347	63.8
60,000	43.1	0.72	1,392	72.4
80,000	56.2	0.70	1,423	79.6

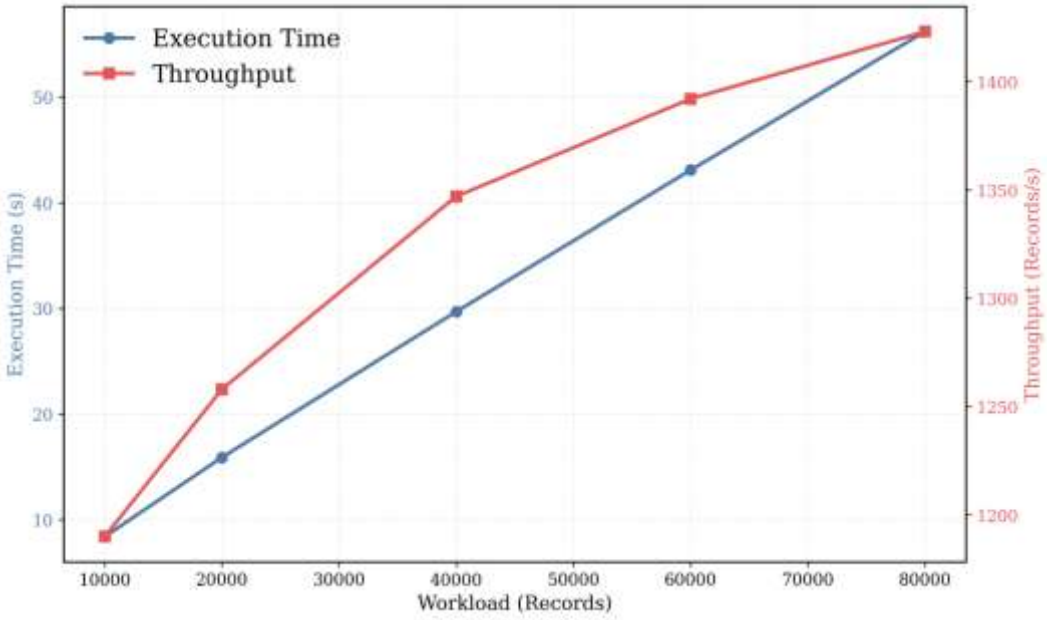

Fig. 12. CloudSim Scalability of TMK-TRAF under Increasing Security Workloads

Table 14 presents the scalability behavior of TMK-TRAF under increasing workloads in the simulated CloudSim environment. As the workload increases from 10,000 to 80,000 records, execution time increases from 8.4 to 56.2 s, while throughput increases from 1,190 to 1,423 records/s. Latency decreases from 0.84 to 0.70 ms/record, whereas resource utilization increases from 42.6% to 79.6%. These results indicate that TMK-TRAF maintains manageable latency and increasing processing throughput under simulated cloud workloads.

6. Conclusion

This paper presented TMK-TRAF, a multi-task banking-security framework for jointly addressing network intrusion and banking fraud detection. The framework integrates TCN, Mamba, and KAN for complementary temporal, long-range, and nonlinear representation learning, while the proposed Threat-Risk Adaptive Fusion (TRAF) learns task-specific fusion weights. The proposed Threat-Risk Consistency (TRC) loss promotes useful shared security representations while preserving task-specific decision boundaries. Focal Loss addresses class imbalance, SHAP supports feature-level interpretability, and CloudSim provides simulated cloud scalability evaluation without claiming real cloud deployment.

Future work will focus on validating TMK-TRAF using larger and diverse real-world banking-security datasets and evaluating it on actual cloud platforms such as AWS, Azure, or Google Cloud. Further research will investigate computational optimization, real-time high-volume processing, concept-drift adaptation, and continual learning to improve robustness against evolving intrusion and fraud patterns.

References

1. Chen, Y., Zhao, C., Xu, Y., Nie, C., and Zhang, Y., “Deep Learning in Financial Fraud Detection: Innovations, Challenges, and Applications,” Data Science and Management, vol. 9, no. 2, 2026, Art. no. 100162. DOI: 10.1016/j.dsm.2025.08.002.
2. Farhan, M., Waheed ud din, H., Ullah, S. et al. Network-based intrusion detection using deep learning technique. Sci Rep 15, 25550 (2025). DOI: 10.1038/s41598-025-08770-0

3. S. Ibrahim, C. Catal and T. Kacem, "The Use of Multi-Task Learning in Cybersecurity Applications: A Systematic Literature Review," Neural Computing and Applications, vol. 36, no. 35, pp. 22053–22079, 2024, doi: 10.1007/s00521-024-10436-3.
4. R. K. M., A. B. Deshpande, Anjali, A. Singa and A. N. M., "AI-Powered Network Intrusion Detection System," 2026 Contemporary Computing Innovations Conference (CCIC), Tirupati, India, 2026, pp. 1–6, doi: 10.1109/CCIC68129.2026.11486086.
5. V. Hnamte, A. A. Najar, C. Laldinsanga, J. Hussain and L. Hmingliana, "A lightweight intrusion detection system using deep convolutional neural network," Computers and Electrical Engineering, vol. 127, p. 110561, 2025, doi: 10.1016/j.compeleceng.2025.110561.
6. R. Nazre, R. Budke, O. Oak, S. Sawant and A. Joshi, "A Temporal Convolutional Network-Based Approach for Network Intrusion Detection," 2024 International Conference on Integrated Intelligence and Communication Systems (ICIICS), 2024, pp. 1–6, doi: 10.1109/ICIICS63763.2024.10860234.
7. M. Akhi, C. Eising and L. L. Dhirani, "Residual temporal CNNs for emerging cyber threat detection in healthcare IoT," Discover Internet of Things, vol. 6, Art. no. 12, 2026, doi: 10.1007/s43926-025-00271-w.
8. Z. Ding, J. Zheng and X. Wu, "NIDS-Mamba: Lightweight Network Intrusion Detection for IoT Sensor Networks via State Space Models," Sensors, vol. 26, no. 9, Art. no. 2766, 2026, doi: 10.3390/s26092766.
9. H. Feng, M. Sun and Y. Wang, "Mambacnn: A Lightweight Intrusion Detection System Based on Mamba for the Internet of Vehicles," Cybersecurity, vol. 9, Art. no. 64, 2026, doi: 10.1186/s42400-025-00473-3.
10. A. B. Mohammed, E. Chamseddine and A. ElAdel, "Enhancing real-time IoT intrusion detection using KAN-based frameworks with SMOTE," Journal of Network and Computer Applications, vol. 249, p. 104461, 2026, doi: 10.1016/j.jnca.2026.104461
11. Ö. Batur Dinler, "UAV Cybersecurity with Mamba-KAN-Liquid Hybrid Model: Deep Learning-Based Real-Time Anomaly Detection," Drones, vol. 9, no. 11, Art. no. 806, 2025, doi: 10.3390/drones9110806.
12. L. Konsta, D. Dimitriou, A. Papathanasiou and V. Liagkou, "Detecting Cyber Fraud in Banking Transactions via Machine Learning Techniques: Implications for Financial Stability," FinTech, vol. 5, no. 1, Art. no. 23, 2026, doi: 10.3390/fintech5010023.
13. M. Sami, A. Mir and G. P. Insany, "Detection of Bank Transaction Fraud Using Machine Learning," Engineering Proceedings, vol. 107, no. 1, Art. no. 34, 2025, doi: 10.3390/engproc2025107034.
14. H. AbouGrad and L. Sankuru, "Online Banking Fraud Detection Model: Decentralized Machine Learning Framework to Enhance Effectiveness and Compliance with Data Privacy Regulations," Mathematics, vol. 13, no. 13, Art. no. 2110, 2025, doi: 10.3390/math13132110.
15. H. Babu, Shanmugasundaram, M. J. Epsibha Robert and S. Pavithra, "Hybrid Machine Learning for Context-Aware Personalized Fraud Detection: Pre and Post Transactions Analysis," 2025 3rd International Conference on Sustainable Computing and Data Communication Systems (ICSCDS), Erode, India, 2025, pp. 1214–1221, doi: 10.1109/ICSCDS65426.2025.11167860.
16. H. Xia, Y. Huang, S. Fang, Q. Wang and J. Shen, "Financial Fraud Detection Based on an Explainable Multi-Layer Framework," International Journal of Financial Studies, vol. 14, no. 6, Art. no. 146, 2026, doi: 10.3390/ijfs14060146.