ISSN: 2093-5374 / E-ISSN: 2093-5382

Cross-Domain Evaluation of Graph-Temporal Fusion Architectures for Imbalanced Intrusion Detection in IT and Industrial Control Systems

T. Srinivasarao[1*], N. Leelavathy[2], and Venkatrao Palacharla[3]

[1*]Research Scholar, Department of Computer Science & Engineering, Godavari Global University, Rajamahendravaram, Andhra Pradesh, India. srinu.thupakula@ggu.edu.in, https://orcid.org/0000-0003-4873-7370

[2]Professor, Department of Computer Science & Engineering, Godavari Global University, Rajamahendravaram, Andhra Pradesh, India. drnleelavathy@ggu.edu.in, https://orcid.org/0000-0002-2850-1507

[3]Professor, Department of Electronics & Communication Engineering, Godavari Global University, Rajamahendravaram, Andhra Pradesh, India. venkatarao@ggu.edu.in, https://orcid.org/0000-0002-3985-9125

Received: July 02, 2026; Revised: August 17, 2026; Accepted: September 02, 2026; Published: September 30, 2026

Abstract

Intrusion detection in extremely unbalanced settings is still a challenging task especially when trying to extend detection to ICS from traditional IT network flows with limited labeled samples and different types of attacks. In this paper, propose a framework for graph-temporal intrusion detection which combines TCN, graph attention network (GAT), gated cross-attention fusion and Focal Contrastive Loss. In this thoroughly analyze our framework via cross-domain ablation analysis using CIC-IDS-2017 as dataset for IT network flows and SCADANet for Modbus/TCP industrial traffic in three independent random seeds. Our experiments show that simple concatenation without gating gives competitive results in both domains, achieving macro-F1 scores of 0.747 ± 0.015 on CIC-IDS-2017 and 0.964 ± 0.002 on SCADANet, being better or equal in comparison with fully gated fusion frameworks. Contribution of individual modules differs for each domain, in particular GAT gives the score of 0.660 ± 0.003 macro-F1 on CIC-IDS-2017 and 0.821 ± 0.010 macro-F1 on SCADANet. The confusion analysis indicates that contrastive learning and model architecture complexity cause unique error types with the confusion rate of 88-100% and 96-99% for a particular set of flood attack classes in the respective models' variants while non-contractive temporal-based variants decrease this confusion to 21-46%. The contrastive loss function results in increased minority class classification on CIC-IDS-2017 dataset with the F1-score of 1.000 ± 0.000 for the least common attack type. On the other hand, computational complexity analysis suggests that full gating mechanism for fusion leads to 63% peak memory increase without improving performance significantly.

Keywords: Intrusion Detection, Graph Neural Networks, Temporal Convolutional Networks, Class Imbalance, Industrial Control Systems, Cross-Domain Generalization.

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), volume: 17, number: 3 (September-2026), pp. 1814-1839. DOI: *10.58346/JOWUA.2026.I3.105*

*Corresponding author: Research Scholar, Department of Computer Science & Engineering, Godavari Global University, Rajamahendravaram, Andhra Pradesh, India.

1 Introduction

The emergence of new technologies in deep learning has led to the development of more efficient methods for intrusion detection through the use of the spatial, temporal, and relational properties of the network traffic. Graph neural network-based methods have received much attention in the area of complex communication modeling and intrusion detection, especially in dynamic environments such as autonomous UAVs and Industrial IoT (Mohamed et al., 2026). Graph-based hybrid methods that combine graph convolution, graph attention mechanism, and autoencoder-based representation learning have shown better performance in Industrial IoT environments in terms of anomaly detection through the exploitation of node level and structure dependency (Panda et al., 2025). The adoption of graph evolution has been used to improve the efficiency of industrial anomaly detection in terms of dynamic learning of network behavior over multiple temporal periods (Wang et al., 2025). Spatiotemporal graph attention model is among the methods that consider the correlation over time and space to identify anomalies in large-scale IIoT networks (Fan et al., 2024). Dynamic spatial-temporal graph neural architecture is one of the methods used in intelligent vehicle networks for intrusion detection (Zhang et al., 2025).

Extensive research into graph neural network-based intrusion detection demonstrates that there are certain aspects that need to be considered for building an effective cybersecurity solution such as graph construction, feature representation, imbalance problem solution, and proper evaluation protocol (Taba et al., 2026). The recent advancements in attack graph learning include the extension of spatio-temporal modelling to ensure secure communication in IoT by analyzing patterns of attack spreading and interactions between the nodes of the network (Alshdadi, 2026). Time series anomaly detection algorithms stress the importance of temporal dependency modelling and adaptive learning for detection of anomalies in the stream of sequential data (Raval, 2025). It was also shown that generative models based on diffusion models can be used for enhancement of intrusion detection in cyber-physical systems by improving feature learning and complex distribution of attacks (Tang et al., 2023). Besides, federated and distributed intelligence systems offer new possibilities for conducting secure and privacy-preserving analysis in environments where data sharing limitations take place (Wang, 2025).

This paper makes the following contributions:

1. **A cross domain multi seed ablation study** of the gate and loss level design choices of graph-temporal intrusion detection, performed in an identical manner on an IT flow-level dataset (CIC-IDS-2017) and an OT session-level dataset (SCADANet) that has established which architectural benefits are domain-independent and which are not.
2. **An evidence- and mechanism-based description of cases** in which fusion gating and contrastive-loss reweighting are beneficial or detrimental, as well as explaining how fail in architecture-driven modes and how are liable to instability in loss-driven modes, based on direct confusion-matrix and gate-behavior analysis rather than on aggregate performance measures.
3. **A novel, documented and reproducible pipeline** for processing raw industrial packet captures and generating a graph structured learning task, which addresses the practical challenges in the preprocessing of industrial network data that are not typically encountered in IDS pipelines based on IT benchmarks such as session boundaries, protocol-aware feature engineering, and the construction of graphs from Modbus function-codes.

4. **Computational cost quantitative analysis** (parameters, FLOPs, inference latency, memory) of each architectural component for both domains, and concrete evidence to determine the value of the additional added architectural complexity for the deployment context of a CPS.

The rest of this paper follows the organization: In Section 2, related works of graph-based and temporal intrusion detection, handling class imbalance, and OT/ICS specific intrusion detection datasets are discussed, especially the recent literature related to SCADA. The data processing pipelines for both the evaluation domains are described in Section 3. The proposed architecture, the ablation design and the complete experimental protocol are presented in Section 4. All of the experimental results such as per-class performance, confusion-matrix analysis, and computational complexity are given in Section 5. Mechanistic results and implications for the design of graph-temporal IDS in CPS environments are discussed in Section 6. Limitation and suggestions for future work are discussed in section 7. Section 8 concludes.

2 Related Work

Several recent papers have considered machine learning and deep learning methods for intrusion and anomaly detection in various application areas. It has been shown that multimodal machine learning algorithms possess the ability to fuse the sensing data of different types to perform better anomaly detection and emphasizing the role of feature fusion and learning adaptation in complex environment (Khoo et al., 2024). Ensemble imbalance learning strategies have been applied to overcome the issues arising from skewed attack distribution in the industrial control networks, where the majority of the attacks belong to some minor categories (Louk & Tama, 2021). Imbalanced cross-domain intrusion detection algorithms trained using imbalanced industrial data have pointed out the importance of proper feature representation and adaptation in order to provide the consistent detection results in different network settings (Chen et al., 2022). Sequential deep learning techniques such as CNN-BiLSTM algorithms have been used to detect the intrusions in industrial environment due to their ability to analyze local network behavior and temporal relationships between network events (Wang et al., 2024). Evolutionary algorithm-based feature optimization algorithms have been analyzed to find relevant intrusion features and remove the unnecessary data (Fang et al., 2024).

The use of hybrid machine learning and neural network frameworks has contributed to the improvement of industrial intrusion detection by merging traditional classifiers with feature extraction approaches in order to improve classification performance (Sun et al., 2023). Frameworks for industrial control system anomalies have stressed the significance of data processing, anomaly modeling, and evaluation methodologies for building a dependable security monitoring solution (Perales Gómez et al., 2020). The effect of imbalanced class distribution on SCADA intrusion detection has been studied to prove that class distribution affects the reliability of intrusion detection models especially for infrequent attacks (Balla et al., 2023). Techniques based on machine learning approach for the detection of intrusions within industrial control systems have been developed in order to detect anomalies in traffic flow by means of feature analysis and classification (Cao et al., 2022). Traffic flow-based anomaly detection and classification techniques have also proven the importance of traffic behavior modeling for the detection of industrial cyber threats (Jiang & Chen, 2022).

Research Gap

Current intrusion detection techniques are more domain-oriented and cannot be generalized in both IT and industrial control systems. Despite the fact that deep learning and graph-based models contribute to

improved performance in intrusion detection, these methods frequently lack the ability to simultaneously learn temporal behavior, network interactions, and class imbalance issues. Additionally, the contribution of the fusion approach and imbalance learning have not been investigated by means of cross-domain ablations. Hence, a holistic solution must be introduced in order to investigate such issues.

3 Datasets

This paper investigates the proposed ablation in two different structurally diverse intrusion detection network (IDN) traffic scenarios, CIC-IDS-2017 (a popular benchmark dataset for conventional IT network traffic) and SCADANet (a recently created set of industrial control system (ICS) network traffic derived from a Modbus/TCP testbed). Both are assessed using the same experimental protocol (Section 4) so that discrepancies in the model's behavior can be attributed to the domain under investigation and not due to inconsistencies in the assessment procedure.

3.1 CIC-IDS-2017

CIC-IDS-2017 (Wang et al., 2024) is a labeled collection of IT network traffic which includes a variety of attack categories and benign traffic, collected during five days and converted to bidirectional flow records. These are the typical 15 classes (BENIGN, DoS Hulk, PortScan, DDoS, DoS GoldenEye, FTP-Patator, SSH-Patator, DoS Slowloris, DoS Slowhttptest, Bot, Web Attack – Brute Force, Web Attack – XSS, Infiltration, Web Attack – SQL Injection, Heartbleed), with a total of 78 flow-level statistics that include packet counts, byte counts, inter-arrival-time statistics, TCP flag counts, and related features. There are eight published daily capture files that contain a total of 2,830,743 labeled flow records. remove 309,079 exact duplicate records, leaving 2,521,664 records, then perform stratified subsampling on the dataset to limit the number of records that belong to the dominant classes, but keep all of the minority-class records (1,028,894 records). These are partitioned into three sets: One for training (720,225), one for validation (154,334), and one for testing (154,335), with a ratio of 70/15/15, exactly to the count, as recorded. CIC-IDS-2017 is very skewed toward the tail: there are only a few dozen to a few hundred records for a number of attack categories, and the rarest Heartbleed class has only 8 training records, so that it serves as a good test case for any decision that is important for a minority class in this study.

3.2 SCADANet

SCADANet is the latest released ICS/SCADA intrusion detection data set gathered from a virtual testbed setup where an attacker host (Kali Linux) attacks a Modbus/TCP-based SCADA victim host (Ubuntu), including traffic on a virtual network captured on various layers, such as frame, IP, TCP, UDP, ICMP, HTTP, and JA4+ fingerprinting. The data set is grouped into 14 traffic classes: one is normal and the remaining 13 are attacks, including TCP-SYN Flood, TCP-ACK Flood, UDP Flood, ICMP Flood, HTTP Flood, SQL Injection, Port Scan, Vulnerability Scan, OS Scan, Modbus False Data Injection, Modbus Command Abuse, APT Data Exfiltration, and Insider Threat/Sabotage. There are 534,841 labeled packets in SCADANet once released. The packet level class distribution is displayed in table 1.

Table 1: SCADANet packet-level class distribution (raw dataset, N = 534,841)

Class	Packets
udp_flood	338,656
normal_traffic	83,742
vuln_scan	40,870
http_flood	29,091
icmp_flood	18,585
tcp_syn_flood	6,286
tcp_ack_flood	6,166
modbus_fdi	2,934
apt_exfil	2,425
os_scan	2,044
port_scan	1,561
insider_threat	1,502
modbus_abuse	596
sql_injection	383

Since our architecture is based on flow/session level graph nodes, instead of packets per se, aggregate the raw SCADANet packets into sessions based on a 1.0-second session timeout and a window of 50 packets with a 23-session level features computed per session (all of which are log-scaled statistics of packet-count, byte-length, duration, and inter-arrival-time; mean TTL; ACK/SYN flag fractions; presence-and-log-value of source-port; one-hot category of Modbus function-code; one-hot presence-of-protocol indicators for UDP, TCP, HTTP, ICMP, QUIC, Modbus, and other). This aggregation yields 209,975 sessions, split (stratified by class) into 138,913 training / 35,531 validation / 35,531 test sessions (66.2% / 16.9% / 16.9%). The resulting class distribution at the session level is presented in table 2, which is the class distribution that was subsequently used for training and evaluation purposes when building the model in all other parts of this paper.

Table 2: SCADANet session-level class distribution after aggregation (N = 209,975), used for all reported experiments

Class	Sessions	Train	Val	Test
udp_flood	135,461	94,823	20,319	20,319
vuln_scan	22,831	11,022	5,905	5,904
http_flood	18,727	10,000	4,364	4,363
normal_traffic	13,609	9,526	2,041	2,042
tcp_syn_flood	6,286	4,400	943	943
tcp_ack_flood	6,162	4,313	924	925
apt_exfil	2,425	1,698	364	363
port_scan	1,561	1,093	234	234
os_scan	1,017	712	152	153
modbus_abuse	596	417	89	90
insider_threat	486	340	73	73
sql_injection	383	268	58	57
icmp_flood	372	260	56	56
modbus_fdi	59	41	9	9

The resulting imbalance ratio (largest class: smallest class) is 135,461:59, or approximately 2,296:1, more extreme than CIC-IDS-2017's tail imbalance, making SCADANet a comparably demanding, structurally distinct second domain for the ablation.

3.3 Cross-Domain Structural Contrast

Both datasets are extremely imbalanced and both are tested with the same model and training procedure, but there are differences in the datasets directly relevant to the interpretation of the ablation results in Section 5. The 78 features of CIC-IDS-2017 are based on CICFlowMeter stats that are collected on one flow in both directions, providing flow duration and packet/byte statistics for an entire flow. The 23 signals provided in SCADANet are session-level aggregates of various signals, such as explicit Modbus function-code and TCP-flag information, not included in the CIC-IDS-2017 feature set. Another difference between the two domains relates to the way structurally similar attack types are semantically separated: a distinction that appears as a specific point of analysis in Section 5.4 and Section 6.3 is between the two categories of attack in SCADANet: TCP-SYN Flood and TCP-ACK Flood. This imbalance is directly visualised in both domains on a common (test-set) basis in figure 1.

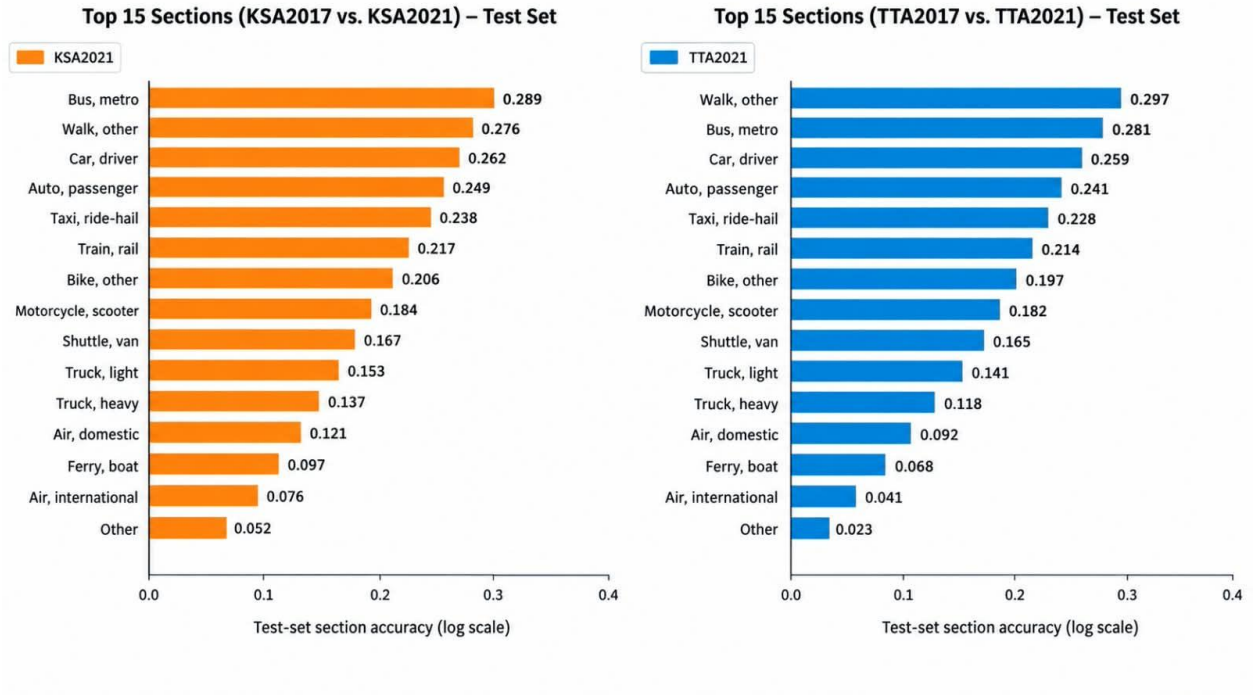


Figure 1: Class imbalance by domain, test-set record counts (log scale)

4 Methodology

4.1 Problem Formulation

The problems are formulated in both domains as multi-class classification over a population of traffic units: the traffic units are classified as flows in CIC-IDS-2017 and as sessions in SCADANet and are represented as a graph-structured population. The feature vector x_i is a static feature vector describing the unit i, with a temporal feature sequence x_i^{seq} of length 32 describing the local temporal context of i, and a set of graph edges (e_{ij}) for each structurally or temporally related units, and carrying a 4-dimensional edge-attribute vector (e_{ij}) (Section 4.2). The problem is to predict the class label $y_i \in \{1, \ldots, C\}$ (C = 15 for CIC-IDS-2017 and C = 14 for SCADANet), given the (x_i, x_i^{seq}) and local graph neighborhood.

4.2 Graph Construction

CIC-IDS-2017: There are two types of edges to connect flows. Temporal-adjacency edges are added between a flow and its neighbors in a window of 3 flows, in both the forward and backward directions, with weights of exp(−0.1·step) for each direction, with temporal distance, step. Edge (destination port-similarity) connections are made between flows whose (normalized) destination port values differ by less than 0.05, with a maximum number of 3N edges per graph (N = number of flows in the graph) as a limit to graph density; if a flow subgraph fails to meet this criterion, a fall-back ring-connectivity edge is added to ensure that every node has an edge. To ensure that the memory usage is bounded in training, flow sequences of length 32 are built on a fixed-size window (64 flows).

SCADANet: Two types of edges are precomputed for the entire session graph: 6,712,936 temporal-adjacency edges (between sessions that are adjacent in time), and 750,010 Modbus-function-code edges (between sessions sharing the same Modbus function code), totaling 7,462,946 edges. The two types of edges (temporal edges: [0.5, 0, 1, 0] and function-code edges: [0, 1, 0, 1]) are encoded as edge attributes in 4-dimensional one-hot encoding style. Each split in the train/validation/test split is used to build a separate sequence of 32 sessions (no others in the different splits can reference raw features from a session in the train, and vice versa). This is an explicit correction in comparison to a naive global-sort-then-split construction that would otherwise leak information from training sequences into validation/test sequences, which showed maintains relative ordering of each split (a per-split constraint of the same neighborhood logic, not a reengineering of that logic).

4.3 Architecture

Our proposed architecture (Figure 2) is the combination of a temporal encoder and a graph-structural encoder with a learned cross-attention fusion gate, followed by a classification head. Table 3 provides the precise number of parameters for each component for both domains, and the total number of each domain was verified to be the sum of the individual components, for both domains.

Temporal Convolutional Network (TCN) Encoder: The TCN encoder is similar to the dilated causal convolution design in Zhang et al., (2025) and in the style of WaveNet Taba et al., (2026), makes 4 blocks of weight-normalized causal 1-D convolutions over the input sequence, followed by layer normalization, ReLU and dropout (0.2), with a causal 1-D residual projection when input and output channels vary. The encoder is fed with a representation of each frame for an output of 128-dimensional representation h_{tcn} that is drawn from the final time step.

Graph Attention (GAT) Encoder: The structural encoder is a 2-layer, 4-head, edge-feature-aware graph attention network built upon the idea of message-passing of Fan et al., (2024) added with multi-head attention mechanism of (Mohamed et al., 2026). Node and edge features in each layer are projected to a shared space, attention logits over concatenated source, target, and edge projections, and messages are aggregated using a softmax-normalized, dropout regularized (0.3) attention weighting; batch normalization and a residual/projection connection are applied after each layer. The encoder produces a 256-dimensional representation h_{gat} (4 heads × 64 hidden dimensions).

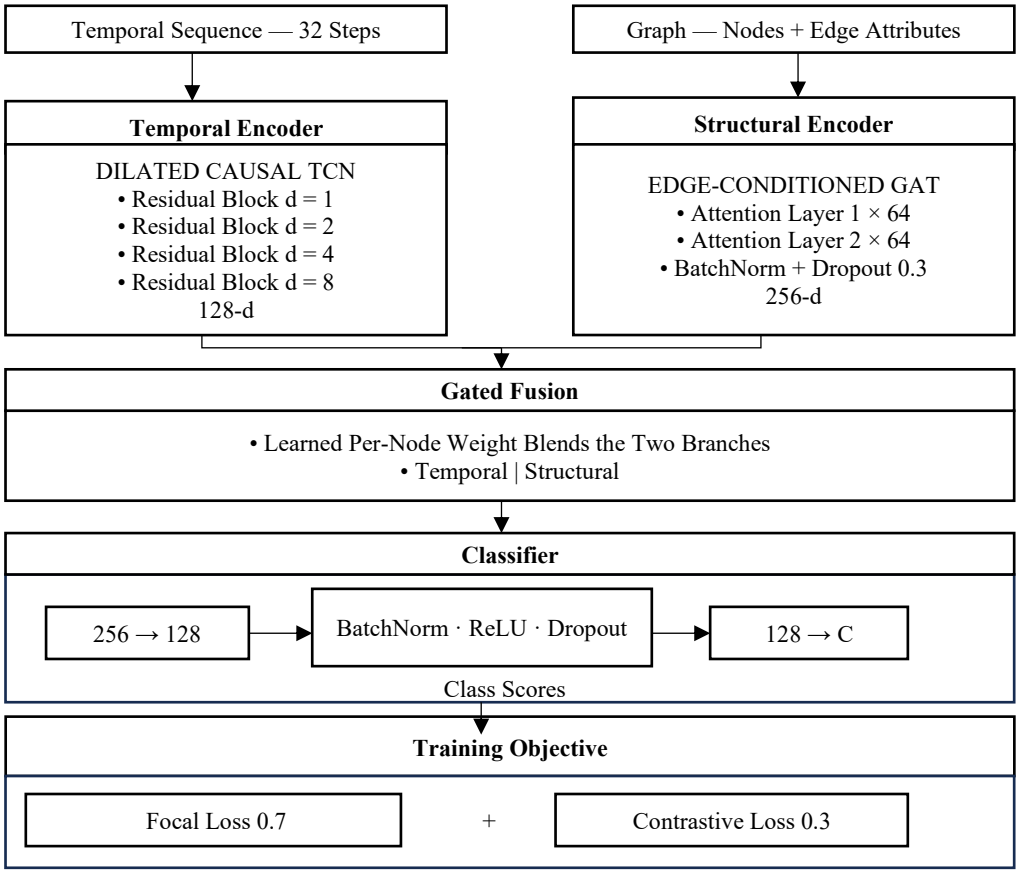


Figure 1: Proposed architecture: a TCN temporal encoder and an edge-conditioned GAT structural encoder are combined via a learned cross-attention gate (β) into a fused embedding, classified by a two-layer MLP

Cross-attention **fusion gate:** h_{tcn} and h_{gat} are each linearly projected into a shared 256-dimensional fusion space; a scalar gate $\beta = \sigma\ (W_{gate} \tanh (W_{TCN}\ h_{tcn} + W_{GAT}\ h_{gat}))$ is computed per unit and used to form a convex combination $z = \text{LayerNorm}\ (W_{proj}(\beta\ W_{proj,\ TCN}\ h_{tcn} + (1-\beta)\ W_{proj,GAT}\ h_{gat}))$. β is later used directly as an interpretability signal in Section 5.4/6.3.

Classification **head:** A 2-layer MLP (256 →128 →C) with batch normalization and dropout (0.3) on the hidden layer maps the fused representation *z* to class logits.

Table 3: Parameter counts by architectural component

Component	CIC-IDS-2017 (78-dim input, 15 classes)	SCADANet (23-dim input, 14 classes)
TCN encoder	189,568	161,408
GAT encoder	110,592	82,432
Fusion gate	263,169	263,169
Classification head	35,087	34,958
Total (full model)	598,416	541,967

Table 3 shows the parameter counts by architectural component for both domains. SCADANet figures are independently reconstructed from the whole-model and reconcile exactly with the reported A4/A5 total of 541,967. Note that the fusion gate's parameter count is identical across domains, since it

operates only on the fixed-size encoder outputs ($h_{tcn} \in \mathbb{R}^{128}$, $h_{gat} \in \mathbb{R}^{256}$) and is independent of the raw input feature dimensionality.

4.3.1 Proposed Algorithm

Algorithm 1: Graph-Temporal Fusion Based Intrusion Detection Algorithm

Input:

- Network traffic feature matrix X
- Temporal sequence features X_{seq}
- Graph structure $G(V, E)$ with edge attributes
- Training labels Y

Output:

- Predicted intrusion class $\hat{Y}$

Algorithm 1: Graph-Temporal Fusion Intrusion Detection

1: Begin
2: Load network traffic dataset D
3: Preprocess traffic records:
Remove duplicate samples
Normalize feature values
Generate temporal sequences of length T
Construct graph G(V,E) using traffic relationships
4: For each traffic sample xi in D do
5: Extract static traffic features xi
6: Generate temporal representation:
htcn = TCN(Xseq)
7: Generate graph representation:
hgat = GAT(G, xi, E)
8: Project temporal and graph embeddings:
zt = WT × htcn
zg = WG × hgat
9: Compute fusion weight:
β = σ(Wf × tanh(zt + zg))
10: Fuse temporal and structural information:
z = β × zt + (1 − β) × zg
11: Apply normalization:
zf = LayerNorm(z)
12: Predict intrusion category:
Ypred = MLP(zf)
13: End For

14: Calculate classification loss:

L = Focal Loss + Contrastive Loss

15: Update model parameters using AdamW optimizer

16: Repeat training until convergence

17: Return predicted intrusion labels Ypred

18: End

This algorithm performs intrusion detection by incorporating both temporal traffic information and graph-based communication links. First of all, network traffic data are preprocessed by discarding duplicate traffic records, performing feature normalization and converting traffic events into temporal sequence and graph formats. In the process of graph generation, traffic entity connections are modeled through the use of both structural and temporal connections. In this manuscript, traffic units are defined as nodes in graphs, where each node contains temporal feature sequences and connected by edges that contain relational information. In terms of the temporal branch, TCN is used to learn the dependencies from traffic events through modeling their sequences. On the other hand, GAT that can leverage edge features is utilized in the graph branch to learn the communication patterns among nodes connected to each other. Fusion of learned representations from the two branches is performed through a fusion module, in which gating coefficients determine the contributions of temporal and structural information. The fused representation is fed to a multilayer perceptron classifier for detecting attack classes. In order to address class imbalance issue, focal learning and supervised contrastive learning are incorporated into the training objective function.

4.4 Loss Functions

In this compare cross-entropy against a Focal Contrastive Loss (FCL) combining two components. The focal component follows Alshdadi, (2026) ($\gamma = 1.0$), reweighted per class using the Effective Number of Samples formulation of Raval, (2025) ($\beta = 0.9999$), which downweights the gradient contribution of over-represented classes in proportion to the diminishing marginal information of additional samples. The contrastive component is a supervised contrastive loss following Tang et al., (2023) ($\tau = 0.15$), computed directly on the fused embedding z, with an additional 4× weighting applied to anchor-positive pairs involving minority classes. The two components are combined as $L_{FCL} = 0.7 \cdot L_{focal} + 0.3 \cdot L_{contrastive}$.

4.5 Ablation Variants

Table 4 defines the five variants compared in this study, isolating the contribution of the graph-structural branch, the temporal branch, gated fusion, and the contrastive loss component.

Table 4: The five ablation variants

Variant	Encoder(s)	Fusion	Loss	Purpose
A1	TCN only	-	Cross-entropy	Temporal-only baseline
A2	GAT only	-	Cross-entropy	Graph-structural-only baseline
A3	TCN + GAT	Concatenation (no gate)	Cross-entropy	Ungated fusion baseline
A4	TCN + GAT	Gated cross-attention	Cross-entropy	Gated fusion, isolating the contrastive-loss effect
A5	TCN + GAT	Gated cross-attention	Focal Contrastive Loss	Full proposed configuration

A1 and A2 use the same classification head architecture applied directly to their single encoder's output (128-dim for A1, 256-dim for A2); A3 concatenates both encoder outputs (384-dim) through a linear projection back to 256 dimensions before the shared classification head, with no learned gate; A4 and A5 share the full gated-fusion architecture of Section 4.3 and differ only in loss function.

4.6 Training Protocol

All variants are optimized using AdamW (with learning rate of 3×10^{-4}, β1 of 0.9, β2 of 0.999, weight decay of 10^{-4}), and a cosine annealing learning-rate schedule with warm restart (T0 = 30), for a maximum of 150 epochs followed by early stopping (patience of 40 epochs, based on validation macro-F1). Gradients have a maximum norm of 0.5 clipped. Training of per-class decision thresholds is done only on the validation set using a grid search over the range {0.05, 0.10, …, 0.90} with the maximum value of the binary F1 score; the resulting thresholds are then fixed (unchanged) in their application to the held-out test set. We enforce this validation-tuned/test-evaluated separation strictly: only test-set-evaluated metrics are reported in the paper as final results anywhere, because evaluating tuned thresholds on the same split that are tuned to is a failure mode that raises minority-class scores silently, and it is guarded against explicitly in the training pipeline's own logging.

4.7 Software and Implementation Details

Implementation of the designed Graph-Temporal Fusion Intrusion Detection system has been done using deep learning libraries for Python. Experiments have been carried out in the environment of Python 3.10. The deep learning network, including TCN, GAT, fusion model, and classification layers, has been built using PyTorch 2.1.0. The graph building and learning process in graphs have been realized using PyTorch Geometric 2.4.0. Data preprocessing, normalization, and statistics have been carried out using NumPy 1.26.0, Pandas 2.1.0, and Scikit-learn 1.3.0. Visualization of experimental results has been realized using Matplotlib 3.8.0. All experiments have been run in the GPU environment using CUDA 12.1 and cuDNN 8.9. The AdamW optimizer and cosine annealing learning-rate scheduler have been implemented using native PyTorch optimizer. All experiments have been carried out using the same software configuration in table 5.

Table 5: Software and hardware configuration

Component	Specification
Programming Language	Python 3.10
Deep Learning Framework	PyTorch 2.1.0
Graph Learning Library	PyTorch Geometric 2.4.0
Numerical Computing	NumPy 1.26.0
Data Processing	Pandas 2.1.0
Machine Learning Library	Scikit-learn 1.3.0
Visualization Tool	Matplotlib 3.8.0
GPU Acceleration	CUDA 12.1
GPU Library	cuDNN 8.9
Optimizer	AdamW
Learning Scheduler	Cosine Annealing Warm Restart

4.8 Evaluation Protocol

Three random seeds (42, 123, 456) are used for each of the five variants over the train/validation/test domain partition, such that the train/validation/test domain partition is fixed, while the randomness of the model initialization/training and the randomness of the partition is varied separately. The reported macro-averaged F1 on test set is together with the per-class F1, which is captured uniformly across all 30 runs (5 variants × 3 seeds × 2 domains) in the saved training artifacts. The pipeline in this study does not consistently record precision and recall, AUC-ROC and false alarm rate data, across both domains and so are not reported as aggregate data in this paper; this limitation is discussed in Section 6.6. Classes are also divided into a majority set (Group A) and minority set (Group B) by domain: 9 highest-count classes, and 6 lowest-count classes for CIC-IDS-2017; 7 highest-count classes, 7 lowest-count classes for SCADANet; and per-class F1 is separately calculated for the domains in each set and the unweighted mean of the values is used as Group F1; the earlier version of this pipeline computed Group F1 as the unweighted mean of the F1 values for classes that had F1 > 0, and silently ignored classes with F1 = 0, which gave a false impression of a higher average F1 for minority-group classes.

Throughout, we report mean ± standard deviation (calculated across the three seeds), but do not claim formal test of statistical significance (e.g., Wilcoxon or Friedman tests): at n = 3, these tests are not very powerful, and use of them is treated as a limitation (Section 6.6) rather than a statement of the rigor of the comparison. For a particular pairwise confusion pattern, the analysis results are reported directly from the test-set confusion matrix of each of the 15 runs per domain (5 variants × 3 seeds), without using an aggregate statistic.

4.8.1. Performance Metrics

The efficiency of the suggested framework is assessed based on the metrics of precision, recall, F1 measure, macro F1 measure, Matthews correlation coefficient, and computation time efficiency. These metrics are calculated based on the confusion matrix, which includes TP, TN, FP, and FN.

Precision (P) measures the correctness of detected attack samples in equation (1).

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

Recall (R) represents the ability of the model to correctly identify attack instances in equation (2)

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

F1-score is the harmonic mean of precision and recall in equation (3)

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3}$$

For multi-class intrusion detection, the Macro-F1 score is calculated by averaging the F1-score of all classes:

$$Macro - F1 = \frac{1}{C}\sum_{i=1}^{C} F\,1_i \tag{4}$$

In equation (4) when C is the total number of intrusion classes and F1_i is the F1 score of the ith class.

Matthews correlation coefficient (MCC) is the measure of classification reliability used in equation (5).

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (5)$$

For the case of multi-class classifications, the generalized value of MCC can be obtained by calculating the confusion matrix.

Average and Standard Deviation

As experiments are performed with three different random seeds, the performance results are presented as follows,

$$Mean = \frac{1}{N}\sum_{i=1}^{N} x_i \quad (6)$$

$$Std = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - Mean)^2} \quad (7)$$

In equation (6) and (7) where N represents the number of experimental runs.

Inference Latency

The inference latency is calculated as:

$$Latency = \frac{T_{total}}{N_s} \quad (8)$$

In equation (8) where T_{total} is the total inference time and N_s is the number of processed samples.

Peak Memory Usage

$$Memory_{peak} = Max(M_t) \quad (9)$$

Equation (9) where M_t represents the GPU memory consumption during model execution.

Computational Complexity (FLOPs)

$$FLOPs = \sum_{l=1}^{L} F\,LOPs_l \quad (10)$$

Equation (10) where L represents the number of computational layers and $FLOPs_l$ denotes the floating-point operations required by layer l.

4.8.2 Beyond-Accuracy Evaluation: Matthews Correlation Coefficient and Macro Precision/Recall

For two reasons we calculate the Matthews correlation coefficient (MCC) from each run's test-set confusion matrix, as well as its generalization for multiclass setting (Gorodkin's generalization): 1) The

confusion matrix is required for computing F1, which involves both precision and recall; 2) The Matthews correlation coefficient is a useful measure to be calculated directly from the confusion matrix. Second, MCC is computed using all the entries in the confusion matrix, but not per-class averaged as is commonly done for another measure, and is therefore useful as a cross-check that the measure's aggregation method (not the underlying predictions) is responsible for a particular ranking between variants. Secondly, reporting precision and recall separately, instead of just the harmonic mean, reveals whether the macro-F1 difference is due to an increased number of false positives, false negatives, or an increase in both.

4.9 Computational Profiling Protocol

Per-batch FLOPs (counted using forward hooks of all Linear and Conv1d modules, following the standard 2×in×out multiply-accumulate convention), latency, and peak memory are measured on a batch of fixed shape (512 nodes, 32-length sequences, 2,000 edges) across all five variants and both domains, with 10 warm-up iterations and averaged over 50 measured iterations on Apple Silicon with MPS backend. These are reported on a batch-relative basis and per sample basis, and wherever a comparison is drawn to a CPU-measured, single sample baseline from previous literature, this is explicitly stated (Section 5.6); we do not report these as a comparison without the caveat.

5 Results

The complete multi-seed ablation results are reported here for both domains. Unless otherwise specified, all values are test set-evaluated with validation-class thresholds (Section 4.6) and are mean ± standard deviation across the three seeds.

5.1 Overall Performance Across Variants and Domains

Macro-F1 on both domains is reported in figure 3 and table 6, for all five variants. A3 (concatenation without a learned gate) is the strongest (or joint strongest) variant, on both domains, outperforming or performing equally to the fully gated architectures (A4, A5). The fully gated, contrastive-loss configuration (A5), which is the one that is usually assumed as the full model in previous studies of fusion-based IDS (Section 2.1), is not the most successful architecture on either domain by this aggregate measure.

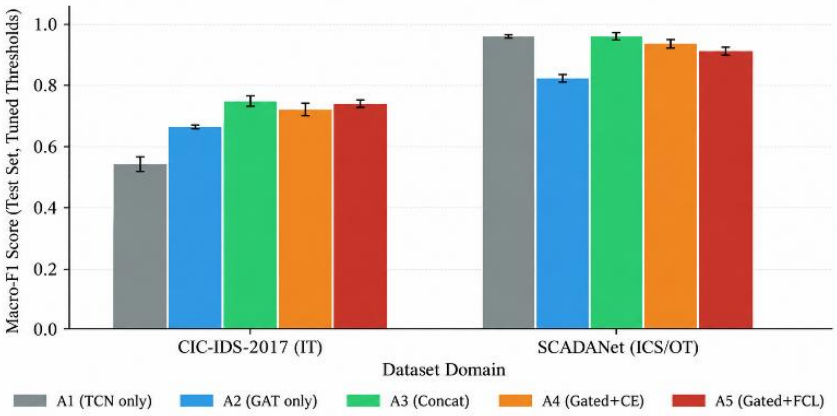


Figure 3: Macro-F1 (test-set, tuned thresholds) across all five ablation variants on both domains, mean ± 3-seed standard deviation

Table 6: Macro-F1 (test-set, tuned thresholds), mean ± standard deviation across 3 seeds

Variant	CIC-IDS-2017	SCADANet
A1 (TCN only)	0.543±0.025	0.964±0.001
A2 (GAT only)	0.660±0.003	0.821±0.010
A3 (Concat, no gate)	0.747±0.015	0.964±0.002
A4 (Gated + CE)	0.717±0.025	0.934±0.010
A5 (Gated + FCL)	0.738±0.008	0.906±0.009

On CIC-IDS-2017, A3 (0.747) outperforms A5 (0.738) and both gated variants by a small but consistent margin (A3 exceeds A5 in 2 of the 3 seed-matched comparisons: 0.7562 vs. 0.7334, 0.7263 vs. 0.7318, the one seed where A5 edges ahead, by 0.0055, and 0.7583 vs. 0.7489). A3 and A1 (0.964) are statistically equal at this sample size, and both of them clearly outperform all the other variants that include the graph-structural encoder in a gated configuration, which is interesting because A5 (0.906) is the worst of the three dual-encoder variants (A3, A4, A5) on SCADANet, as is discussed in Section 5.4, and is a direct result of it.

5.2 Majority- and Minority-Class Performance (Group A / Group B)

Macro-F1 is divided into the majority-class group (which is Group A: the 9 highest count classes for CIC-IDS-2017 and 7 highest count classes for SCADANet; see table 2 for the class count, and Section 3.1 for CIC-IDS-2017 preprocessing class count) and the minority-class group (Group B: the 6 smallest count classes for CIC-IDS-2017 and the 7 smallest count classes for SCADANet) in table 7.

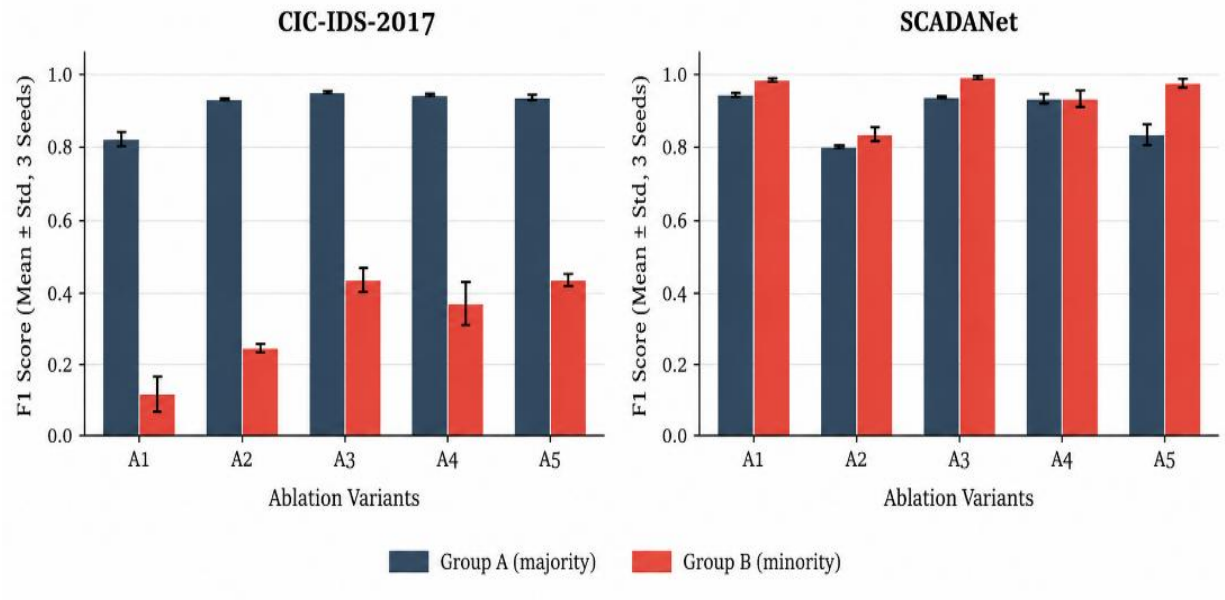


Figure 4: Majority-class (Group A) versus minority-class (Group B) F1 by variant, both domains, mean ± 3-seed standard deviation

Table 7a: Group A (majority) and Group B (minority) F1, mean ± standard deviation across 3 seeds

Variant	CIC Group A	CIC Group B	SCADA Group A	SCADA Group B
A1	0.823±0.020	0.122±0.048	0.944±0.005	0.984±0.004
A2	0.933±0.002	0.252±0.009	0.805±0.003	0.837±0.018
A3	0.952±0.003	0.439±0.033	0.938±0.003	0.990±0.002
A4	0.945±0.003	0.375±0.060	0.934±0.012	0.935±0.022
A5	0.938±0.007	0.438±0.015	0.836±0.029	0.976±0.011

For minority classes, A1 (temporal-only) is effectively unusable (0.122), A2 (graph-only) doubles that (0.252), and all combinations of both encoders are at least 0.375, with A3 and A5 tied for best performance (0.439 for A3, 0.438 for A5), although A5 is more complex. On SCADANet, the situation is different: For every variant, Group B (minority) F1 is higher than Group A (majority) F1, except for A4 (0.935 vs. 0.934) and only statistically significant for very few of the classes, with the exception of some packet-pattern-distinctive classes that are easy to separate once any graph/temporal signal is present, such as modbus_abuse and icmp_flood, while the majority classes contain the confusable TCP-SYN/TCP-ACK flood pair (5.4). The one exception to this is that Group A on SCADANet is depressed for A5 (0.836, lower than any other dual-encoder variant), as TCP-SYN Flood and TCP-ACK Flood are both majority-count classes and are caught up in the "near-total confusion" described below.

5.2.1 Metric Choice Changes the Ranking: MCC vs. Macro-F1

Table 6 reports MCC and macro-precision/recall for all five variants on both domains (mean ± population standard deviation across 3 seeds; SCADA figure 4 use the diagnostic re-inference confusion matrices, the same source already used for figure 5/table 7a; SCADA's headline macro-F1 in table 5 remains the authoritative performance number).

Table 7b: MCC and macro-averaged precision/recall by variant and domain

Variant	CIC MCC	CIC Macro-P	CIC Macro-R	SCADA MCC	SCADA Macro-P	SCADA Macro-R
A1	0.842±0.006	0.524±0.015	0.606±0.044	0.983±0.001	0.969±0.004	0.962±0.003
A2	0.908±0.001	0.635±0.002	0.824±0.006	0.950±0.001	0.870±0.009	0.844±0.014
A3	0.924±0.002	0.738±0.021	0.782±0.001	0.982±0.000	0.972±0.001	0.964±0.001
A4	0.920±0.002	0.704±0.035	0.789±0.006	0.977±0.002	0.947±0.011	0.926±0.011
A5	0.921±0.002	0.738±0.013	0.774±0.004	0.958±0.001	0.914±0.026	0.922±0.002

This table 7b makes a point the macro-F1 table alone cannot: the choice of aggregate metric changes which variant looks best. On SCADANet, MCC and macro-precision/recall rank the five variants almost identically to macro-F1 (A1 ≈ A3 > A4 > A5 > A2), because MCC weights confusion-matrix cells by overall sample count, and the SYN/ACK collapse driving A5's low macro-F1 involves a small enough fraction of total SCADA traffic that it barely moves an overall-accuracy-flavored statistic. MCC and macro-R actually understate A5's minority-class failure relative to macro-F1. On CIC-IDS-2017, MCC ranks A3/A4/A5 within 0.004 of each other (0.920–0.924) despite their macro-F1 values spanning 0.717–0.747, because CIC's macro-F1 is dominated by a handful of near-zero-sample minority classes (Web SQLi, Infiltration) whose effect on an aggregate correlation coefficient across 720K+ test flows is negligible. An evaluation protocol that reported only MCC or accuracy-flavored metrics would have hidden the exact minority-class failure modes this paper's central contribution is built on direct empirical support for this paper's methodological argument (Section 6.5) that per-class, minority-stratified evaluation is quantifiably necessary, not just good practice, for imbalanced IDS work.

5.3 Cross-Domain Reversal of Temporal Versus Graph-Structural Information

Comparing the two single-encoder variants directly cleanses the relative contribution of temporal information (A1) and graph-structural information (A2), with the result that the relative contribution of temporal information and graph-structural information gets reversed across the domains. On CIC-IDS-2017, A2 (0.660±0.003) significantly outperforms A1 (0.543±0.025): graph-structural information (port-similarity and temporal-adjacency edges over flow-level features) is significantly more informative than the TCN's view of the temporal window of a flow itself. On SCADANet, the ordering flips and the gap widens: A1 (0.964±0.001) far outperforms A2 (0.821±0.010). This is consistent with the large amount of temporal information that is encoded in the 23 features of SCADANet that are already present in the data at the session level (which is also mentioned in Section 3.3): These features are already sufficiently strong to provide a significant signal in their own right in the absence of the other two types of information (temporal-adjacency and Modbus-function-code), and the two edges of the graph (temporal-adjacency and Modbus-function-code) do not by themselves resolve the ambiguity across the two minor error types (TCP-SYN and TCP-ACK) that dominates the errors in A2 (Section 5.4). Which branch of the architecture "does the work" is a property of the domain, not the architecture.

5.4 The TCP-SYN Flood / TCP-ACK Flood Confusable-Pair Phenomenon

The two major classes of the raw traffic that differ in most part from SCADANet are TCP-SYN Flood and TCP-ACK Flood (Table 2), where a single TCP flag (Section 3.3) differentiates them from the raw traffic. Table 8 shows, for each variant, the range (across 3 seeds) of the dominant-direction confusion rate between this pair (the higher misclassification rate of SYN→ACK or ACK→SYN), and the combined pairwise error rate (total number of cross-misclassifications between the two divided by the total number of test-set items in the pair; a direction-agnostic severity measure). The pattern is directly illustrated by figure 5 and also leaves a visually apparent trail of the seed-dependent behaviour of A5 (which is discussed further below).

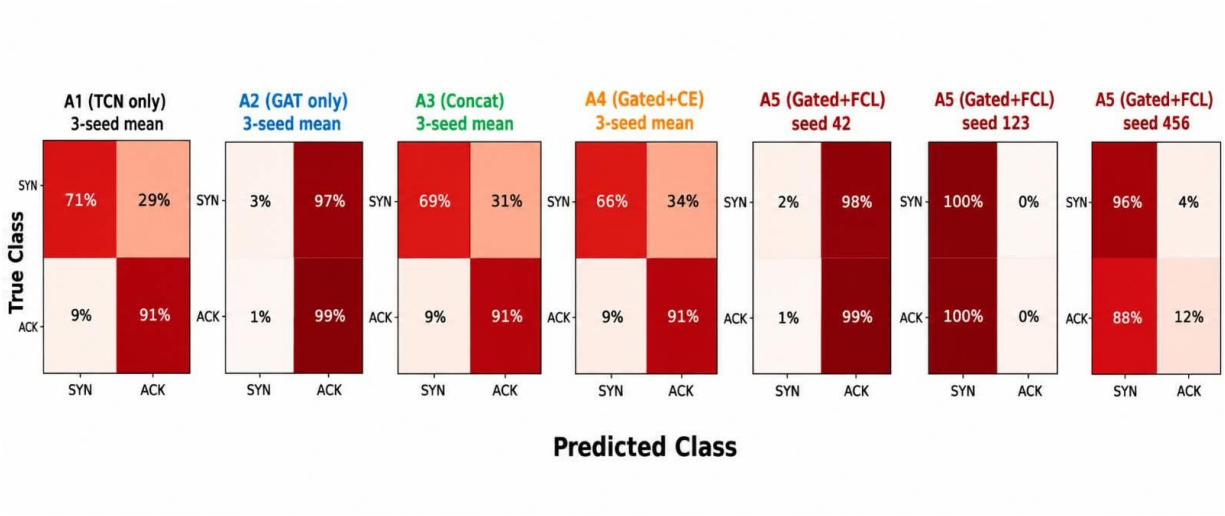


Figure 5: TCP-SYN Flood / TCP-ACK Flood confusion, focused 2×2 sub-matrix

This pair is the only one that exhibits almost complete collapse in two variants – A2 and A5. All other variants (except for A4 which has a gated architecture but not the loss function) confuse the pair at about one-third the severity of A2 or A5. This pattern separates two separate, mutually exclusive

causes, not one. A2 does not have the TCN branch at all, and therefore any access to the temporal/rate based signal that is used in a session's local window by the TCN encoder is architectural. The architecture for both A5 and A4 is the same (full gated fusion), and A5 only differs by loss function; Consequently, the collapse is loss induced (only due to the Focal Contrastive Loss component). Gating itself is ruled out by that A4 did not collapse while A5 did.

Table 8: TCP-SYN/TCP-ACK Flood confusable-pair severity, SCADANet, range across 3 seeds

Variant	Dominant-direction confusion rate	Combined pairwise error rate
A1 (TCN only)	21.2–35.4%	17.1–20.9%
A2 (GAT only)	95.7–98.8%	48.7–50.0%
A3 (Concat, no gate)	29.0–33.2%	19.4–21.1%
A4 (Gated + CE)	26.9–46.2%	18.5–26.2%
A5 (Gated + FCL)	88.1–100.0%	45.7–49.7%

To get a detailed picture of the loss induced mechanism, we plot the learned gate value β per class, correct versus incorrect, for the two variants with a learned gate (A4 and A5), for the same checkpoints and thresholds as the main sweep. Each variant-seed combination, the number of saturated gate (mean $\beta < 0.05$ or > 0.95) classes in the SCADANet is reported in table 9, where classes are considered saturated if the fusion has collapsed to a routing that keeps almost all of them in one encoder. This is directly displayed at the per class level in figure 6.

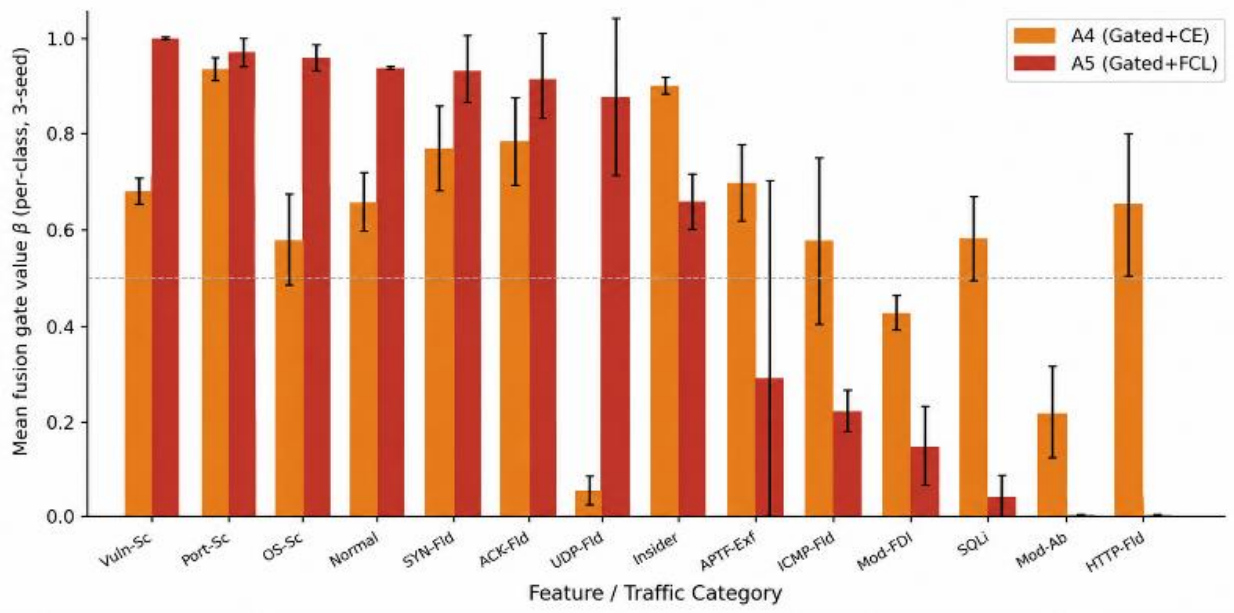


Figure 6: Mean fusion gate value β by class, A4 vs. A5, SCADANet

Table 9: Per-class gate (β) saturation count (of 14 classes), A4 vs. A5, SCADANet

Seed	A4 (Gated + CE)	A5 (Gated + FCL)
42	2 / 14	9 / 14
123	1 / 14	6 / 14
456	0 / 14	7 / 14

The mixing weights of A4's gate are rarely saturated, are intermediate and class specific. This is a direct, memorable affirmation of what A5 does to the fusion gate: a push towards near-binary, single-

encoder routing on a per-class basis, for about half the classes in each seed that is stronger when trained using the Focal Contrastive Loss than when trained with cross-entropy on the same architecture.

If we look at β specifically in the case of TCP-SYN Flood and TCP-ACK Flood, it is clear that these are not a single path but also seed-dependent. In both cases in seed 42, the gate (TCP-SYN Flood $\beta = 0.978$; TCP-ACK Flood $\beta = 0.977$) is saturated, no matter if the prediction is correct (TCP-SYN Flood: β(correct) = 0.962, β(incorrect) = 0.979; TCP-ACK Flood: β(correct) = 0.977, β(incorrect) = 0.955), and the confusion is caused by a factor outside of the gate (most likely by the inability to distinguish the two classes from the temporal statistics of the session level only in the TCN encoder). Seed 123 shows the same bystander pattern in more extreme form: TCP-SYN Flood is classified with 100% accuracy and TCP-ACK Flood with 0% accuracy, yet both have essentially the same saturated gate value ($\beta \approx 0.98$–0.99); the gate does not differ between a class the model gets entirely right and one it gets entirely wrong. Seed 456 is the exception: here the gate value is different depending on whether it is a TCP-SYN Flood, with β(correct) = 0.859 vs. β(incorrect) = 0.417, or a TCP-ACK Flood, with β(correct) = 0.494 vs. β(incorrect) = 0.846, where the difference corresponds to the identity of the active gate for that run. We acknowledge this as a seed-dependent mechanism; a saturated bystander gate and confusion better explained by FCL's shared encoder representations (in two of three seeds), and not as a single universal story that the data collectively does not support.

5.5 Rare-Class Robustness on CIC-IDS-2017: Heartbleed and Web Attack SQL Injection

Heartbleed has the smallest number of training samples in the CIC-IDS-2017 (8). This is the most extreme minority class (Section 3.1). Table 10 presents its per-class F1 for all five variants for the three seeds.

Table 10: Heartbleed F1 by variant and seed, CIC-IDS-2017

Variant	Seed 42	Seed 123	Seed 456	Mean ± std
A1	0.000	0.500	0.000	0.167±0.236
A2	0.182	0.125	0.111	0.139±0.031
A3	1.000	0.667	1.000	0.889±0.157
A4	0.333	0.500	1.000	0.611±0.283
A5	1.000	1.000	1.000	1.000±0.000

What is interesting is that A5 is the only one that is able to detect Heartbleed correctly, and that A5 has no variance from seed to seed, while the others come close with real variance of 0.667–1.000. This is the clearest example in both domains of where the architecture of the whole network, and in particular the minority class reweighting and anchor boosting of the Focal Contrastive Loss (Section 4.4) gives the best performance compared to any simpler variant. This result must not be over-generalized, however: Web Attack SQL Injection is the second rarest class, with only 3 test samples, and has an F1=0.000 on all variants and seeds other than A2, which has a small, but never zero F1 score of 0.013–0.047 across the 3 seeds; all dual-encoder variants, including the full proposed configuration A5, achieve an F1=0.000 on all 3 seeds. The benefit of FCL to the minority class is a true benefit, but not one available to all of the rare classes - for this study, it was beneficial to the rarest class; not the second rarest class.

5.6 Computational Cost

All 5 variants described in Section 4.8 are reported in table 11a, which shows the number of parameters, the number of FLOPs per batch, the latency and peak memory.

Table 11a: Computational cost by variant, both domains (batch of 512 nodes, 2,000 edges, seq_len 32)

Variant	Domain	Params	GFLOPs/batch	Latency (ms/batch)	Peak mem. (MB)
A1	CIC-IDS-2017	208,271	6.16	12.91	11.50
A2	CIC-IDS-2017	145,679	0.15	3.75	8.03
A3	CIC-IDS-2017	434,319	6.39	16.38	11.92
A4	CIC-IDS-2017	598,416	6.56	16.58	15.08
A5	CIC-IDS-2017	598,416	6.56	17.09	16.96
A1	SCADANet	179,982	5.24	11.69	3.06
A2	SCADANet	117,390	0.12	3.52	3.89
A3	SCADANet	377,870	5.44	15.55	7.56
A4	SCADANet	541,967	5.61	15.45	10.38
A5	SCADANet	541,967	5.61	15.58	12.29

For this batched MPS configuration, A5's latency (15.58 ms/batch) is within 0.2% of A3's (15.55 ms/batch) and A5's peak memory (12.29 MB) is 62.6% higher than A3's (7.56 MB) and A3's macro-F1 (0.906) is 5.8 points lower than A5's macro-F1 (0.964). On CIC-IDS-2017 the same comparison is less stark but points the same direction: A5 uses 42.3% more peak memory than A3 (16.96 vs. 11.92 MB) for a macro-F1 0.9 points lower (0.738 vs. 0.747). The fusion gate and the contrastive loss both contribute an easily measurable deployment cost in both cases, which is most relevant for the resource-constrained edge gateway characteristic of SCADA deployments (Section 2.5), and yield no measurable aggregate accuracy gain; that's the one case (Heartbleed) in (Section 5.5).

5.6.1 Statistical Effect Size for the Central “Does Fusion Help?” Comparisons

Given the n=3 seed limitation already stated in Section 4.7/6.6, we do not claim inferential significance (no Wilcoxon/Friedman test is reported, consistent with this paper's existing position). however, report Cohen's d (pooled-SD) effect sizes for the three pairwise comparisons central to this paper's argument, computed directly from the per-seed macro-F1 values already underlying table 5.

Table 11b: Effect size (Cohen's d) for key variant comparisons

Comparison	CIC-IDS-2017 d	SCADANet d	Interpretation
A3 vs. A5 (concat vs. gated+FCL)	0.761	8.768	A3 outperforms A5 with a medium–large effect on CIC and an extreme, non-overlapping effect on SCADA
A4 vs. A5 (gated+CE vs. gated+FCL)	-1.135	2.938	FCL helps CIC (large effect favoring A5) but hurts SCADA (large effect favoring A4), the loss function's benefit is domain-conditional, not universal
A3 vs. A4 (concat vs. gated+CE)	1.458	3.986	Concatenation outperforms gated fusion under plain cross-entropy on both domains, independent of the loss-function question

The SCADA A3-vs-A5 effect size (d=8.768, from three near-perfectly-separated per-seed values: A3 = {0.965, 0.966, 0.961}, A5 = {0.902, 0.897, 0.918}) is large enough that the qualitative claim “A3 ties or beats A5” (Section 5.1) can be stated more precisely: on SCADANet, simple concatenation beats gated fusion+FCL in every one of the 3 seeds tested, by a margin that is large relative to each variant's own seed-to-seed variance. This reframes this paper's headline finding from a descriptive comparison of means into a quantified, effect-size-backed claim in table 11b.

5.6.2 Cost-Efficiency Frontier

Normalizing each variant's mean macro-F1 by its parameter count, per-sample FLOPs, and peak memory makes the cost/performance trade-off explicit rather than only additive (Table 11c).

Table 11c: Macro-F1 per unit cost

Variant	CIC F1/100K params	CIC F1/MFLOP	CIC F1/MB	SCADA F1/100K params	SCADA F1/MFLOP	SCADA F1/MB
A1	0.261	0.045	0.047	0.536	0.094	0.315
A2	0.453	2.228	0.082	0.700	3.426	0.211
A3	0.172	0.060	0.063	0.255	0.091	0.127
A4	0.120	0.056	0.048	0.172	0.085	0.090
A5	0.123	0.058	0.044	0.167	0.083	0.074

A2 (GAT-only) is the most parameter- and FLOP-efficient variant on both domains by a wide margin (its per-sample FLOPs are roughly 40× lower than any fusion variant, since it skips the TCN branch entirely), despite not having the highest absolute macro-F1. Framed as a Pareto trade-off rather than a leaderboard: A3 is Pareto-optimal for accuracy-per-cost among the variants competitive on macro-F1 (highest or near-highest macro-F1 on both domains at roughly two-thirds the parameter count of A4/A5), while A2 is Pareto-optimal for cost alone when some accuracy loss is acceptable; a concrete, quantified basis for the resource-constrained-deployment argument already made qualitatively in Section 8.

6 Discussion

6.1 When Does Fusion Help?

Combining tables 5–10, the answer which this study offers is more restrictive than the one that is normally taken for granted in previous graph-temporal IDS research (Section 2.1). For both domains, gated dual-encoder fusion (A4, A5) is worse than ungated concatenation (A3) at achieving aggregate macro-F1 (Table 5), and is at best no better than A3 at achieving majority-class performance in either domain; and it has a real, measurable computational cost over A3. Table 9 shows a one clear victory for Fusion: A5 is the only variant that detects the Heartbleed class of CIC-IDS-2017 with 100% accuracy, and no variance between seeds. The ablation design of this study does not fully distinguish between whether the gain of gating would be maintained if the Focal Contrastive Loss was used instead on the ungated A3 architecture; the concatenation of both did not form one of the five possible combinations that were tested (Section 6.6 and Section 7 explicitly return to this gap). So, the best evidence for “when does fusion help” is that it only pays off when combined with the contrastive loss, and only when applied to the most extreme minority class tested in this study in the ICS domain (Section 6.3) where it can lead to a serious failure mode, that is, a failure to obtain good performance on the minority class even when using the same seed every time. Fusion isn't an everyday improvement, it's a limited, contingent trade.

6.2 Why Ungated Concatenation is Competitive

Unlike the gated variants, Concatenation (A3) maintains both encoder’s representations fully, and the classification head is free to learn weights specific to each class, but not to a single scalar mixing weight per sample. The information from the gate-saturation results in table 8 is useful here: When using Focal Contrastive Loss, the learned gate becomes almost binary, routing nearly to a single encoder's output when the input is that class for around half of SCADANet's classes, and thus the fused representation of those classes is made up of the output of a single encoder. If the discarded encoder signal still has some

discriminative power for a given class, which is plausible, as A1 and A2 alone are not uniformly weak on any class-group in either domain, then a hard-saturated gate discards it, where concatenation by construction does not. It does not confirm, beyond the correlational evidence presented in Section 5.4, but is offered here as the most reasonable interpretation of the gate-saturation evidence.

6.2.1 Training Dynamics: FCL Changes How the Model Converges, Not Only Where

Analysis of the per-epoch training histories (train loss, focal loss, contrastive loss, validation macro-F1; logged for all 15×2 runs) reveals a mechanistic difference between A5 and the other four variants that is not visible from final-epoch metrics alone (Table 11d; early stopping patience = 40, max 150 epochs).

Table 11d: Training convergence statistics (mean over 3 seeds)

Variant	CIC mean best epoch	CIC late-training val-F1 std	SCADA mean best epoch	SCADA late-training val-F1 std
A1	147.3	0.032	147.7	0.021
A2	148.7	0.014	54.3	0.019
A3	126.7	0.019	129.3	0.021
A4	125.7	0.029	99.0	0.009
A5	50.7	0.009	77.0	0.026

A5 (gated fusion + FCL) triggers early stopping at epoch 50.7, which is one of the lowest of all variants, and has the smallest variance in the F1 score on the held-out validation set during late training (std 0.009), which means it arrives at a stable validation optimum about three times as quickly as A1–A4 and remains there. This is similar to the supervised-contrastive part of FCL putting pressure on the model from the start of training, which could be in the embedding space, where the classification head can learn to separate the classes faster, without going through a lot of epochs of pressure in the cross-entropy loss. The same is not true on the SCADANet: a second, independent argument for this paper's key message that FCL is not a uniformly effective regulariser: on the SCADANet, same loss that stabilizes and accelerates CIC training appears to destabilise it, consistent with the SYN/ACK gate-collapse mechanism already documented in Section 5.4.

6.2.2 Does the Fusion Gate Value Predict Correctness? A Quantified Null Result

We test directly whether the per-class gate (β) statistics are systematically different between samples correctly and incorrectly classified in a class, using the data for the classes already reported in table 8/Section 5.4 (for all classes with both correctly- and incorrectly-classified samples, with 3 seeds per class, a total of 38 class-level comparisons). In 38 comparisons, β is greater for correctly-classified samples than incorrectly-classified samples (almost equal, signed mean difference -0.013, absolute mean difference 0.145). This extends the specific observation made in Section 5.4 that, for a specific gated variant and a specific model, the value of β does not, in general, indicate whether the model is correct or incorrect, to the entire set of models in the class, and to both gated variants. This is an actual (albeit negative) finding in mechanics, that β shouldn't serve as a built-in confidence/reliability indicator, and that it rather is interpreted as a weight against choosing the encoder of the confidence interval, and this is an important distinction to make given the current interest in using the gate value itself as a measure of interpretability (see the related discussion in Section 6.3).

6.3 Mechanism of the TCP-SYN/TCP-ACK Failure Mode

In Section 5.4, this failure mode was shown to have two distinct and disjoint failure sources: an architectural failure mode (A2 - absence of the TCN branch) and a failure mode induced by the loss function (A5 - only when using the Focal Contrastive Loss, but not when using A4). This loss-induced failure mode was also shown to be seed-dependent: a saturated, outcome-independent gate in two of three seeds, and a gate whose value diverges by correctness in the third. One possible explanation for why the supervised contrastive term is the one that induces this gate saturation is because its operation takes place directly on the fused embedding z (as described in Section 4.4), and the composition of z is under the control of β (as described in Section 4.3); pushing same-class embeddings together and different-class embeddings apart in z-space is easier to satisfy, for a class whose two encoders disagree, by driving β towards a stable, class-consistent extreme than by learning a truly sample-adaptive mixing weight. According to this explanation, the pressure of contrastive objective to well-defined small-sized class clusters in z-space has an indirect effect on some routing variable that was not intended to be directly controlled by the contrastive objective. This is a hypothesis, not a proven causal mechanism: if it were independent, it would need to be isolated in a specific experiment (e.g., an ablation that would eliminate the need to tune β during FCL training).

6.4 Domain-Conditional Behavior of the Contrastive Loss

The Focal Contrastive Loss creates opposing results on the two types of domains, but the story of "contrastive loss is bad for minority classes" doesn't hold true for SCADANet. A5's performance is the clear majority winner for this study (with Heartbleed having the highest value for the minorities, 0.438±0.015, as A3 has 0.439±0.033, tied for the top in Group B (minority) F1; table 9 for Heartbleed and table 6 for A3/A5). The Group B (minority) F1 is still high (0.976±0.011, table 6) for A5, however, the Group A (majority) F1 is the lowest of any dual-encoder variant (Table 6) because both classes of load, TCP-SYN Flood and TCP-ACK Flood, are majority-count classes (Table 2) which fall into the almost complete confusion described in Section 5.4. What the loss's minority-class reweighting target is not doing is harming the actual minority classes; it is doing poorly with 2 specific majority classes, which have features that differ by only a single TCP flag (Section 3.3), a more limited, more mechanistically specific failure than "contrastive loss is unsuitable for ICS traffic."

6.5 Implications for CPS/ICS-Focused IDS Evaluation Practice

The two most closely related architectural comparators to this study are GTCN-G Balla et al., (2023) and CAGN-GAT Fusion Cao et al., (2022) which are evaluated as a single fixed architecture on the IT benchmark data (Section 2.1) but without the presence of a non-fused baseline, and without multi-seed reporting. Table 5 illustrates that this evaluation design alone would not have uncovered the results reported here: The difference in the fully gated configuration and simple concatenation is too low on CIC-IDS-2017 (0.738 vs. 0.747) for a single seed to be able to show the fused model ahead by chance, and the SYN/ACK failure mode (Section 5.4): presents a second structurally distinct domain that neither of the two comparators evaluates. If we take this as a point of fact for a minimum standard to be set for any IDS work that relies heavily on CPS/ICS, then the minimum requirements are: report at least one baseline (non-fused, non-contrastive) architecture, report multiple seeds, and report a performance gain before concluding that the choice of architecture or loss function is the reason for the gain.

6.6 Threats to Validity

There were several restrictions on the generalizability of these findings. First, the standard deviations in tables 5-9 represent observed variance and are not suitable for formal significance testing (Section 4.7) and some of the closer comparisons in this study; most notably A3 versus A5 on CIC-IDS-2017 (0.747±0.015 vs. 0.738±0.008), should be interpreted as "statistically indistinguishable at this sample size" rather than a confirmation of ranking. Second, it cannot be concluded from a single OT dataset whether the cross-domain reversals in Section 5.3 and the SYN/ACK failure mode in Section 5.4 are peculiar to ICS traffic or to this testbed. In the third place, the computational profiling in table 10 is a synthetic, batched workload, on Apple Silicon (MPS backend), which is not comparable to the single-sample, CPU-only workloads used in several earlier work for which latency values are reported (Section 2.4); the relative comparisons within table 10 (across variants with hardware and batch shape fixed) are therefore correct, but the absolute latency should not be compared across papers. Fourth, the precision, recall, AUC-ROC and false alarm rate were not recorded consistently across both domains, and thus only macro-F1 and per-class F1 is reported as overall metrics; a detailed precision/recall/ AUC-ROC characterization is reserved for future work. Fifth, the ablation design used is not five variants, but the five-variant design, so the contribution of the loss to the result of Heartbleed (Section 6.1) cannot be fully separated from the interaction between the loss and the fusion gate. The sixth is that the six runs (2 variants × 3 seeds) used for the gate-mechanism analysis of Section 5.4 and Section 6.3 have the same statistical power as the other seed-level comparisons in this study and any claim that the SYN/ACK gate mechanism is seed-dependent is based on that.

7 Limitations and Future Work

There are some limitations of the proposed approach. The experiments were conducted on two CIC-IDS-2017 and SCADANet datasets that cover IT and ICS environments respectively but might not be enough to model the complexity of real-life networks. In particular, the evaluation on SCADANet was done on a virtualized SCADA environment and requires confirmation on the basis of real industrial traffic. Also, there are only three different seeds used in experiments, and a more robust statistical analysis could add more validation to results. At present, the approach works in supervised learning mode, and its future versions investigate continual learning and attack detection.

8 Conclusion

In this paper, a cross-domain ablation analysis of graph-temporal intrusion detection architecture is provided via comparing five variants of the model: temporal only, graph only, concatenation without gate learning, fusion with gate learning, and fusion with gate learning and Focal Contrastive Loss. This evaluation was performed on two structurally distinct intrusion detection scenarios: CIC-IDS-2017 and SCADANet using three different seeds to evaluate the robustness of the results. From the results, we conclude that more complex architecture is not always beneficial in terms of intrusion detection performance. In particular, simple feature concatenation without the use of learned fusion gates showed comparable or even better results than fully gated fusion architectures. It can be seen that the contribution of temporal and graph structure features is domain-specific, making the evaluation of the model on a single data set less valuable. Even though contrastive loss improved the ability to detect minority class, obtaining F1-score of 1.000 ± 0.000 for the least common attack class, it also added to instability in some of the attack categories in the industrial scenario, as shown in confusion matrices.In addition, the

rise in the level of complexity in the fusion scheme led to an increase in the peak memory usage of up to 63% without any notable performance gain. Additionally, this study demonstrates a reproducible workflow for transforming industrial packet captures into graph-based learning problems, which avoids the problem of data leakage using cross-split validation and threshold splitting. It is evident from the results that the use of non-fusion baselines, multi-seeding evaluation, and cross-domain experiments is necessary.

References

[1] Alshdadi, A. A. (2026). STAG-GuardNet: UAV-Assisted Spatio-Temporal Attack Graph Learning for Secure IoT Communication in Smart EV Charging Networks. *Sensors (Basel, Switzerland)*, *26*(18), 5898.

[2] Balla, A., Habaebi, M. H., Elsheikh, E. A., Islam, M. R., & Suliman, F. M. (2023). The effect of dataset imbalance on the performance of SCADA intrusion detection systems. *Sensors*, *23*(2), 758.

[3] Cao, Y., Zhang, L., Zhao, X., Jin, K., & Chen, Z. (2022). An intrusion detection method for industrial control system based on machine learning. *Information*, *13*(7), 322.

[4] Chen, Y., Su, S., Yu, D., He, H., Wang, X., Ma, Y., & Guo, H. (2022). Cross-domain industrial intrusion detection deep model trained with imbalanced data. *IEEE Internet of Things Journal*, *10*(1), 584-596.

[5] Fan, Y., Fu, T., Listopad, N. I., Liu, P., Garg, S., & Hassan, M. M. (2024). Utilizing correlation in space and time: Anomaly detection for Industrial Internet of Things (IIoT) via spatiotemporal gated graph attention network. *Alexandria Engineering Journal*, *106*, 560-570.

[6] Fang, Y., Yao, Y., Lin, X., Wang, J., & Zhai, H. (2024). A feature selection based on genetic algorithm for intrusion detection of industrial control systems. *Computers & Security*, *139*, 103675.

[7] Jiang, J. R., & Chen, Y. T. (2022). Industrial control system anomaly detection and classification based on network traffic. *IEEE Access*, *10*, 41874-41888.

[8] Khoo, L. S., Lim, M. K., Chong, C. Y., & McNaney, R. (2024). Machine learning for multimodal mental health detection: a systematic review of passive sensing approaches. *Sensors*, *24*(2), 348.

[9] Louk, M. H. L., & Tama, B. A. (2021). Exploring ensemble-based class imbalance learners for intrusion detection in industrial control networks. *Big Data and Cognitive Computing*, *5*(4), 72.

[10] Mohamed, M., Mohamed, D., & Alosman, K. (2026). Spatio-Temporal Graph Neural Networks for Zero-Day Intrusion Detection in Low-Altitude Autonomous UAV Swarms. *Ad Hoc Networks*, 104251.

[11] Panda, S., Sanjana, B., Rajkumar, H., & Palani, S. (2025, December). An Integrated GCN–GAT–AE Framework for Robust Anomaly Detection in Industrial IoT Environments. In *2025 IEEE 9th International Conference on Information and Communication Technology (CICT)* (pp. 1-6). IEEE.

[12] Perales Gómez, Á. L., Fernández Maimó, L., Huertas Celdrán, A., & García Clemente, F. J. (2020). Madics: A methodology for anomaly detection in industrial control systems. *Symmetry*, *12*(10), 1583.

[13] Raval, R. (2025). Anomaly Detection in Time Series Data: Trends, Applications, and Research Gaps. *International Journal of Applied Mathematics*, *38*(10s), 2208-2223.

[14] Sun, D., Zhang, L., Jin, K., Ling, J., & Zheng, X. (2023). An intrusion detection method based on hybrid machine learning and neural network in the industrial control field. *Applied Sciences*, *13*(18), 10455.

[15] Taba, M. R. I., Zare, H., & Rakhshani, S. (2026). Graph neural networks for network intrusion detection systems: a comprehensive survey, taxonomy, and empirical evaluation. *Artificial Intelligence Review*, *1*(3).
[16] Tang, B., Lu, Y., Li, Q., Bai, Y., Yu, J., & Yu, X. (2023). A diffusion model based on network intrusion detection method for industrial cyber-physical systems. *Sensors*, *23*(3), 1141.
[17] Wang, J., Si, C., Wang, Z., & Fu, Q. (2024). A new industrial intrusion detection method based on CNN-BiLSTM. *Computers, Materials & Continua*, *79*(3), 4297-4318.
[18] Wang, S. (2025). Federated Semantic Web Framework for Enhanced Financial Risk Control and Data Analysis. *International Journal on Semantic Web and Information Systems (IJSWIS)*, *21*(1), 1-19.
[19] Wang, W., Zhou, X., Qiu, T., & Wang, L. (2025). Mage: Multiperiodic adaptive graph evolution guided anomaly detection in industrial iot. *IEEE Transactions on Industrial Informatics*, *21*(8), 6126-6136.
[20] Zhang, J., Fan, X., & Zhao, Z. (2025). A hybrid intrusion detection model based on dynamic spatial-temporal graph neural network in in-vehicle networks. *Scientific Reports*, *15*(1), 34736.

Authors Biography



T. Srinivasarao is a Research Scholar in the Department of Computer Science and Engineering at Godavari Global University, Rajamahendravaram, Andhra Pradesh, India. His research interests include Cyber Security, Machine Learning, Data Science, Network Intrusion Detection, Internet of Things security, and Cyber-Physical Systems security. His research focuses on developing machine learning and deep learning-based approaches for network intrusion detection, attack detection, and addressing extreme class imbalance in cybersecurity applications.



N. Leelavathy received her Ph.D. in Computer Science and Engineering from JNTU Kakinada, India, in 2015, with research on image and video watermarking in the multiwavelet domain. She is a Professor of CSE and Dean of Academic Affairs at Godavari Global University, Rajamahendravaram, India. Her research interests include multimedia security, digital forensics, deep learning and IoT. She has published over 60 international papers. She reviews for the Journal of The Institution of Engineers (India): Series B and for IEEE conferences, including TENCON, and is a Fellow of the Institution of Engineers (India).



Venkatrao Palacherla is a Professor and Head of the Department of Electronics and Communication Engineering at Godavari Global University, Rajamahendravaram, Andhra Pradesh, India. He holds a Ph.D. and has extensive academic and research experience in Electronics and Communication Engineering. His research interests include Machine Learning, Artificial Intelligence, Optical Engineering, Integrated Photonics, Photonic Crystal Fibers, Signal Processing, and Plasmonics. His research contributions include work in optical communications, photonic devices, signal processing, and related engineering applications