Journal of Theoretical and Applied Information Technology
30st June 2026. Vol.104. No.12

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# HIERARCHICAL GRAPH ATTENTION NETWORK WITH BIDIRECTIONAL LSTM FOR REAL-TIME MULTI-CLASS NETWORK INTRUSION DETECTION

**DR.CH.V.SIVARAMPRASAD[1], S VINOD KUMAR[2], KORLA SWAROOPA[3], DR. PAVADA SANTOSH[4], DR K NAGARAJU[5], DR A PHANI SRIDHAR[6], DR. HARI JYOTHULA[7], DR SIVA KUMAR SUBRAMANIAN[8]**

[1]Assistant professor in mathematics, Aditya University,Surampalem,
[2]Assistant Professor, Department of CSE, GMRIT Deemed to be University, Vizianagaram. [3]Department of Computer Science and Engineering (Data Science), Aditya Institute of Technology and Management, Tekkali, India.
[4]Associate Professor, Department of ECE, Vignan's Institute of Information Technolgy, Duvvada, Visakhapatnam.
[5]Assistant Professor, Computer Science and Engineering, Aditya University, Surampalem.
[6]Assistant Professor, Department of AIML, Aditya University, Surampalem.
[7]Associate Professor, Computer Science and Engineering, Aditya University, Surampalem.
[8]Professor & Head IPR, Department of CSE, KG Reddy College of Engineering and Technology, Hyderabad.

Email: pacesrp.maths@gmail.com, sunnapalliv@gmail.com, Drskp.cse@gmail.com, santoshpavada@vignaniit.edu.in, nagarajuk@adityauniversity.in, Phani.addepalli@adityauniversity.in, dr.jyothulahari@gmail.com, drsivashankars@gmail.com

**ABSTRACT**

Network intrusion detection remains a critical unsolved challenge because modern cyberattacks continuously evolve in sophistication, volume, and diversity, rendering traditional signature-based and shallow machine learning systems increasingly inadequate. Despite advances in deep learning, existing models fail to simultaneously capture the relational topology and temporal dynamics of network traffic, leaving significant detection gaps for rare and novel attack variants. This paper addresses that gap by proposing HGAT-LSTM, a unified architecture that achieves 99.14% classification accuracy with a false positive rate of just 0.91%, demonstrating that joint graph-structural and temporal modeling is both technically feasible and operationally viable for real-time intrusion detection.

To meet the challenge of having to detect contemporary threats posed to modern network infrastructure by increasingly advanced cyberattacks, intelligent intrusion detection systems (IDS) need to perform real-time accurate multi-class classification across heterogeneous traffic patterns. Traditional features are limited in expressiveness, topological relations among network entities cannot be modeled, and generalization to zero-day attack variants is poor with such approaches [7]. The deep learning techniques are much more capable, but they ignore the rich structural dependencies of networked environments by treating network flows as independent observations. To this end, in this paper, we develop and propose a new Hierarchical Graph Attention Network integrated Bidirectional Long Short-Term Memory encoder architecture called HGAT-LSTM, which is specialized for high-performance network intrusion detection. The proposed architecture builds a dynamic attributed graph based on observed network sessions, then uses multi-scale graph attention to process each of the three hierarchically-organized layers (i.e. with differentiable pooling) and finally combines structural embeddings of the graphs (i.e. learned via GTN) and temporal sequences from the bidirectional LSTM using cross-attention exploration module. To combat the extreme class imbalance commonly found in intrusion detection datasets, a focal cross-entropy loss function is employed for model training.

On four benchmark datasets, including NSL-KDD, UNSW-NB15, CICIDS-2017 and CICIDS-2018 over extensive experiments show that the proposed method HGAT-LSTM outperforms six competitive baselines (CNN-LSTM, Transformer IDS, GCN-LSTM architectures) with state-of-the-art performance accuracy of 99.14%, macro-averaged F1-score of 98.91%, and AUC-ROC of4962 0.9987. Ablation studies further

validate the essential role of each architectural part, and cross-dataset evaluation demonstrates improved generalization ability. The inference latency of 6.8 ms per sample makes HGAT-LSTM suitable for deployment in real-time intrusion detection pipelines.

**Keywords:** *Intrusion Detection System, Graph Attention Network, Bidirectional LSTM, Network Security, Deep Learning, Hierarchical Pooling, Multi-Class Classification, Focal Loss*

## 1. INTRODUCTION

The exponential increase in the number of devices and services connected to the Internet has led, on its part, to a similar increase in cybercrime threats, with annual global costs due to cyber crime expected to surpass USD 10.5 trillion by 2025 [1]. Enter Network Intrusion Detection Systems (NIDS) — an essential protective layer that analyzes network traffic and provides real-time alerts to suspicious or malicious activities. A NIDS effectively performs traffic classification by distinguishing between normal traffic and multiple classes of attack types — such as denial-of-service (DoS), distributed-denial-of-service (DDoS), probing, remote-to-local (R2L) and user-to-root (U2R) flask attacks — in dynamic information/task-oriented network environments that demand high throughput. Traditional signature-based intrusion detection systems (IDS) like Snort and Suricata achieve low false-positive rates but have high false-negative rates for new or polymorphic attacks [2]. Although the anomaly based statistical methods are effective in detecting zero-day attacks, they are very vulnerable to concept drift and have a high false alarm rate which makes it difficult for operational users to deploy [3].

With the advent of deep learning, the era of network intrusion detection has been completely changed. Existing studies [4] show how Convolutional Neural Networks (CNNs) are able to extract local spatial patterns from network packet headers and payloads, while Recurrent Neural Network (RNN), and Long Short-Term Memory (LSTM) networks could be used for modeling sequential traffic with strong temporal dependency [5]. Hybrid CNN-LSTM models have further improved classification accuracy by jointly modeling spatial and temporal features [6], More recent approaches include Transformer-based architectures utilizing self-attention and showing competitive performance on intrusion detection benchmarks [7]. Despite these advances, a significant limitation remains: existing models do not take into account the rich inter-relational structure between network nodes, protocols and behaviors since they treat network sessions as isolated feature vectors. Alike to any data structure, the elements of graph (nodes and edges) have their own intrinsic meaning expressed via some metrics; Graph Neural Networks (GNNs) [3] is an innate abstraction to depict such structural information which has shown promising results on various applications including those of deep learning based methods for network anomaly detection [6, 7], e.g., Graph Convolutional Networks (GCNs) [8] and Graph Attention Networks (GATs) [9]; however, most formulations are shallow and single-scale in terms of graphs representation, limiting its ability to capture multi-resolution structural patterns essential to distinguish subtle attack signatures from benign traffic.

Despite two decades of research, network intrusion detection remains an open problem for three compounding reasons. First, the attack landscape evolves faster than labeled datasets can be curated, creating a persistent generalization gap between trained models and deployed threats. Second, real-world network traffic exhibits extreme class imbalance — benign sessions can outnumber rare attack classes such as U2R by factors exceeding 1000:1 — causing most classifiers to optimize for majority-class accuracy while silently failing on the attacks that matter most. Third, and most fundamentally, network sessions are not independent observations: attackers orchestrate multi-step, multi-host campaigns whose malicious intent is only discernible when relational and temporal context is jointly analyzed. A single misclassified reconnaissance packet is harmless in isolation but catastrophic when it precedes a lateral movement sequence. These structural realities make intrusion detection a problem that demands architectures capable of simultaneously reasoning over graph topology, temporal sequences, and severely imbalanced distributions — a combination no existing model fully addresses.

We hypothesize that jointly modeling the topological structure of network communication graphs and the temporal evolution of traffic sequences within a unified end-to-end architecture yields significantly superior multi-class intrusion detection performance compared to either modality independently or their naive concatenation. To test this hypothesis, we adopt an experimental comparative research design — consistent with prior IDS benchmarking studies across networking [18], IoT security [19], and industrial control systems domains [22] — wherein the proposed model is trained and evaluated on four

established benchmark datasets under identical preprocessing, splitting, and evaluation protocols. The model is compared against six competitive baselines using accuracy, macro F1-score, AUC-ROC, false positive rate, and inference latency as primary metrics. This work focuses on flow-level and session-level feature analysis and does not cover encrypted payload inspection, adversarial traffic manipulation, or federated deployment scenarios, which are identified as directions for future work.

The contributions of this paper are fourfold and directly address the identified gaps: (1) we introduce a three-layer hierarchical graph attention encoder that captures multi-resolution structural patterns in network communication graphs, overcoming the single-scale limitation of prior GNN-based IDS; (2) we design a cross-attention fusion module that enables bidirectional information exchange between graph and temporal representations, replacing lossy concatenation with adaptive modality weighting; (3) we propose a combined SMOTE and focal loss training strategy that reduces false positives on minority attack classes by 54% relative to the CNN-LSTM baseline; and (4) we provide the most comprehensive comparative evaluation to date across four datasets, six baselines, and five performance metrics, with full ablation and cross-dataset generalization analysis. The remainder of this paper is organized as follows: Section 2 reviews related literature; Section 3 details the proposed HGAT-LSTM architecture; Section 4 presents experimental results and analysis; Section 5 concludes with open questions.

## 2. LITERATURE REVIEW

Machine learning based intrusion detection has a tradition of more than two decades. In early works, classical machine learning algorithms like Support Vector Machines (SVM), k-Nearest Neighbors (k-NN) and Random Forests were applied on manually engineered statistical features extracted from packet headers and flow statistics [10]. While these approaches obtained decent performance on standard datasets, such as KDD Cup 1999, they have issues with scalability, feature engineering cost and generalization to new attack methods [11]. Then, deep learning approaches proved that there were still improvements if they could learn such hierarchical feature representations from raw and/or lightly processed data. Of the earliest deep architectures used in intrusion detection were Stacked Autoencoders (SAEs); these allowed for unsupervised pretraining of feature representations before supervised fine-tuning [12]. CNNs are a common choice for packet byte sequences and feature matrices with depths spanning shallow networks to deep residual models [13]. The majority of sequential traffic classification approaches are based on LSTM networks, as they capture long-range temporal dependencies by utilizing sequences of packets [14].

[23] Dive deeper into NIDS blog with more recent literature discussing prospects for improved performance Inspired from the application of attention mechanisms [14] to Neural Machine Translation (NMT), these models weight their features over individual time steps, or vice versa, yielding performance improvements broadly across datasets when compared with vanilla LSTMs [15]. Bidirectional LSTMs (Bi-LSTMs), which have advantages over unidirectional counterparts by encoding entire past and future context in a traffic sequence, are particularly useful for offline forensic analysis [16]. Completely eliminating recurrence, transformer architectures based purely on self-attention have delivered competitive performances with significantly less training time due to their parallelizability [17]. As the GNN research community matured, graph-based approaches have attracted more and more attention. Another method is to build flow-level graphs based on the GCN layer and perform spectral graph convolution for neighbor information aggregation [18]. Instead of relying on fixed aggregation weights, GAT based models derive learned attention coefficients allowing better discrimination between benign and malicious traffic patterns [19]. Network anomaly detection methods based on hierarchical graph pooling methods like DiffPool and SAGPool have been also implemented to obtain coarsened graph representations at various scales [20].

Although significant progress has been made, there is an obvious gap in the literature: none of the previous works has jointly utilized hierarchical multi-scale graph attention, bidirectional temporal encoding and cross-attention fusion with a unified end-to-end architecture for intrusion detection. Moreover, most of the existing studies assess only a single dataset, which makes it difficult or impossible to draw generalization conclusions. We summarize our survey of the most relevant recent works in Table 1, which features important architectural choices they cover, evaluated datasets as well as performance metrics they report. This comparison is a strong motivation for design decisions made in HGAT-LSTM.

*Table 1: Comparative Summary of Related Work in Deep Learning-Based Intrusion Detection*

| Reference | Method | Dataset(s) | Accuracy (%) | F1-Score (%) | Year | Limitation |
|---|---|---|---|---|---|---|
| [10] | SVM + RF Ensemble | KDD99 | 97.10 | 95.80 | 2019 | Hand-crafted features; poor on novel attacks |
| [12] | Stacked SAE | NSL-KDD | 97.43 | 96.21 | 2019 | Shallow graph; single scale |
| [14] | LSTM | NSL-KDD | 97.68 | 96.74 | 2020 | No graph topology; limited temporal context |
| [16] | Bi-LSTM + Attention | UNSW-NB15 | 97.21 | 96.75 | 2020 | No structural modeling |
| [18] | GCN-LSTM | NSL-KDD/UNSW | 97.65 | 97.20 | 2021 | Static graph; single-scale GCN |
| [19] | GAT + CNN | CICIDS-2017 | 98.03 | 97.51 | 2021 | No temporal encoder; limited pooling |
| [20] | HierPool-GNN | UNSW-NB15 | 98.12 | 97.73 | 2022 | No sequence modeling |
| [21] | Transformer IDS | NSL-KDD | 97.93 | 97.58 | 2022 | High latency; no graph structure |
| [22] | GraphSAGE-LSTM | CICIDS-2017 | 98.24 | 97.91 | 2023 | Single-scale; no cross-attn fusion |

## 3. PROPOSED METHODOLOGY: HGAT-LSTM ARCHITECTURE

### 3.1 Problem Formulation

Let D = $\{(x_i, y_i)\}_{i=1}^{N}$ be a labeled network traffic dataset with $x_i \in \mathbb{R}^d$ being the d-dimensional feature vector corresponding to the i-th network session and $y_i \in \{0, 1,..., C-1\}$ for the class label of that sample associated with C classes (or one benign + C − 1 attack classes ). We divide a temporal observation window T into non-overlapping segments of size W, and build a dynamic graph G_t = (V_t, E_t, A_t, X_t) for each window t: V_t is the set of network nodes (hosts/services), E_t is the set of edges representing communication sessions between them, A_t ∈ $\mathbb{R}^{|Vt|\times|Vt|}$ gives the weighted adjacency matrix and X ∈ $\mathbb{R}^{|Vt|\times d}$ specifies node features. The goal is to learn a mapping f_θ: G_t → ŷ that results in the expected focal cross-entropy loss minimized over the data distribution.

### 3.2 Dynamic Graph Construction

Let G = (V, E, A, X) be an attributed directed graph constructed based on a temporal window W consecutive network sessions. An edge between every two nodes i,j ∈ V is formed if within the time frame t hjdeno from node i to node j a communication session is initiated. In this case, the edge weight w_{ij} encodes aggregated statistical features of that communication: overall byte volume, session duration, and protocol type. The node feature matrix X ∈ $\mathbb{R}^{N\times d}$ is formed from the homogeneous concatenation of NSL-KDD or UNSW-NB15 41-dimensional or 49-dimensional feature vectors, respectively for all sessions associated with each node. An adjacency matrix is constructed by adding self-loops and symmetrically normalizing:

$$\tilde{A} = D^{-1/2}(A + I_N)D^{-1/2} \quad (1)$$

where $\tilde{D}$ is the degree matrix of (A + I_N). This normalization ensures numerical stability and

prevents vanishing gradients during backpropagation through the graph layers.

### 3.3 Hierarchical Graph Attention Network (HGAT)

At the heart of this proposed architecture, are a three-layer hierarchical Graph Attention Network (HGAT) which progressively coarsens the graph representation while learning multi-resolution structural embeddings. Each GAT layer \(l\) computes updated node embeddings by aggregating neighboring features based on learned attention coefficients.

For a node i and its neighbor j ∈ N(i), the unnormalized attention coefficient is computed as below, by applying a shared attention mechanism parameterizing each learnt weight vector: a ∈ ℝ^{2d_l}:

$$e_{ij}^l = LeakyReLU(a^T \cdot [W^l h_i^l \| W^l h_j^l]) \quad (2)$$

where W^l ∈ ℝ^{d_{l+1}×d_l} is the learnable weight matrix, h_i^l ∈ ℝ^{d_l} is the node embedding at layer l, and ‖ denotes vector concatenation. The attention coefficient is normalized using the softmax function over the neighborhood N(i):

$$\alpha_{ij}^l = exp(e_{ij}^l) / \Sigma_{k \in N(i)} exp(e_{ik}^l) \quad (3)$$

The updated embedding for node i at layer l+1 using K parallel attention heads is:

$$h_i^{l+1} = \|_{k=1}^K \sigma(\Sigma_{j \in N(i)} \alpha_{ij}^{l,k} \cdot W^{l,k} \cdot h_j^l) \quad (4)$$

where σ(·) denotes the ELU activation function. For the final GAT layer, the multi-head outputs are averaged rather than concatenated to control dimensionality:

$$h_i^L = \sigma(1/K \cdot \Sigma_{k=1}^K \Sigma_{j \in N(i)} \alpha_{ij}^{L,k} \cdot W^{L,k} \cdot h_j^L) \quad (5)$$

### 3.4 Differentiable Hierarchical Pooling

At each hierarchical level, we apply DiffPool, a differentiable hierarchical graph pooling operator that generates a coarse representation of the original graph. At pooling level l, a learned soft cluster assignment matrix S^l ∈ ℝ^{n_l × n_{l+1}} maps n_l nodes to n_{l+1} clusters:

$$S^l = softmax(GNN\_pool^l(A^l, H^l)) \quad (6)$$

$$H^{l+1} = S^{l,T} H^l \in \mathbb{R}^{n_{l+1} \times d} \quad (7)$$

$$A^{l+1} = S^{l,T} A^l S^l \in \mathbb{R}^{n_{l+1} \times n_{l+1}} \quad (8)$$

Bin pooling ratio = r = 0.5bin, where on each hierarchical level (second and third) the graph is reduced from n_{0} nodes to n_{0}/2 and n_{0}/4 respectively The graph-level representation is subsequently obtained by means of global mean pooling over the final coarsened node set:

$$h_G = READOUT(\{h_i^L : i \in V_L\}) = 1/n_L \cdot \Sigma_{i=1}^{n_L} h_i^L \quad (9)$$

An auxiliary link-prediction loss is added during training to encourage the pooling operator to preserve graph connectivity:

$$L_{LP} = \|A^l - S^l (S^l)^T\|_F \quad (10)$$

### 3.5 Bidirectional LSTM Temporal Encoder

Along with the HGAT encoder, a Bidirectional LSTM (Bi-LSTM) model is applied to hold the sequential nature of network flow feature vectors in an observation time frame. Given a temporal sequence X = {x_1, x_2,..., x_W} of W flow vectors, Bi-LSTM calculates forward and backward hidden states:

$$\rightarrow h_t = LSTM_f(x_t, \rightarrow h_{t-1}) \quad (11)$$

$$\leftarrow h_t = LSTM_b(x_t, \leftarrow h_{t+1}) \quad (12)$$

$$h_t^{seq} = [\rightarrow h_t \| \leftarrow h_t] \in \mathbb{R}^{2d_h} \quad (13)$$

where d_h = 256 is the hidden state dimensionality. The LSTM cell updates are governed by the standard gating equations with input gate i_t, forget gate f_t, output gate o_t, and cell state c_t:

$$i_t = \sigma(W_i [h_{t-1}, x_t] + b_i), \quad f_t = \sigma(W_f [h_{t-1}, x_t] + b_f) \quad (14)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh(W_c [h_{t-1}, x_t] + b_c) \quad (15)$$

$$o\_t = \sigma(W\_o\,[h\_\{t-1\}, x\_t] + b\_o),\ h\_t = o\_t \odot tanh(c\_t) \quad (16)$$

The sequence-level representation is obtained by applying a temporal attention mechanism over the Bi-LSTM outputs to produce a weighted aggregation:

$$\beta\_t = softmax(v\_a\hat{}T\ tanh(W\_a\ h\_t\hat{}\{seq\} + b\_a)) \quad (17)$$

$$h\hat{}\{seq\} = \Sigma\_\{t=1\}\hat{}W\ \beta\_t \cdot h\_t\hat{}\{seq\} \in \mathbb{R}\hat{}\{2d\_h\} \quad (18)$$

### 3.6 Cross-Attention Fusion Module

The cross-attention fusion module adaptively fuses the structural representation h_G ∈ ℝ^{d_G} and temporal sequence representation h^{seq} ∈ ℝ^{2d_h}. The module calculates cross-modal attention in both directions, graph to sequence (G→S) and sequence to graph (S→G), and fuses the resulting enriched representations:

$$Q\_G = h\_G\,W\_Q\hat{}G,\ K\_S = h\hat{}\{seq\}\,W\_K\hat{}S,\ V\_S = h\hat{}\{seq\}\,W\_V\hat{}S \quad (19)$$

$$A\_\{G \to S\} = softmax(Q\_G\,K\_S\hat{}T / \sqrt{d\_k}) \cdot V\_S \quad (20)$$

$$Q\_S = h\hat{}\{seq\}\,W\_Q\hat{}S,\ K\_G = h\_G\,W\_K\hat{}G,\ V\_G = h\_G\,W\_V\hat{}G \quad (21)$$

$$A\_\{S \to G\} = softmax(Q\_S\,K\_G\hat{}T / \sqrt{d\_k}) \cdot V\_G \quad (22)$$

$$h\hat{}\{fused\} = LayerNorm(h\_G + A\_\{G \to S\}) \,\|\, LayerNorm(h\hat{}\{seq\} + A\_\{S \to G\}) \quad (23)$$

The fused representation h^{fused} ∈ ℝ^{d_G + 2d_h} is passed through two fully connected layers with dropout regularization (p = 0.4) before the final softmax classification layer. The output probability vector ŷ ∈ ℝ^C is computed as:

$$\hat{y} = softmax(W\_2\ ReLU(W\_1\ h\hat{}\{fused\} + b\_1) + b\_2) \quad (24)$$

### 3.7 Focal Cross-Entropy Loss

Due to the heavy class imbalance in intrusion detection datasets, where benign traffic makes up over 80% of instances yet rare attack classes (e.g., U2R) may be less than 0.1%, we adopt a focal cross-entropy loss originally used in dense object detection:

$$L\_focal = -\Sigma\_\{c=1\}\hat{}C\ \alpha\_c\,(1 - \hat{p}\_c)\hat{}\gamma\ log(\hat{p}\_c) \quad (25)$$

where α_c is the class-specific weighting factor inversely proportional to class frequency, γ ≥ 0 is the focusing parameter (set to γ = 2.0 in our empirical experiments), and p̂_c is the predicted probability for class c. The overall training loss we define combines focal classification loss with auxiliary pooling regularization loss:

$$L\_total = L\_focal + \lambda \cdot L\_LP,\ \lambda = 0.01 \quad (26)$$

Journal of Theoretical and Applied Information Technology
30st June 2026. Vol.104. No.12

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

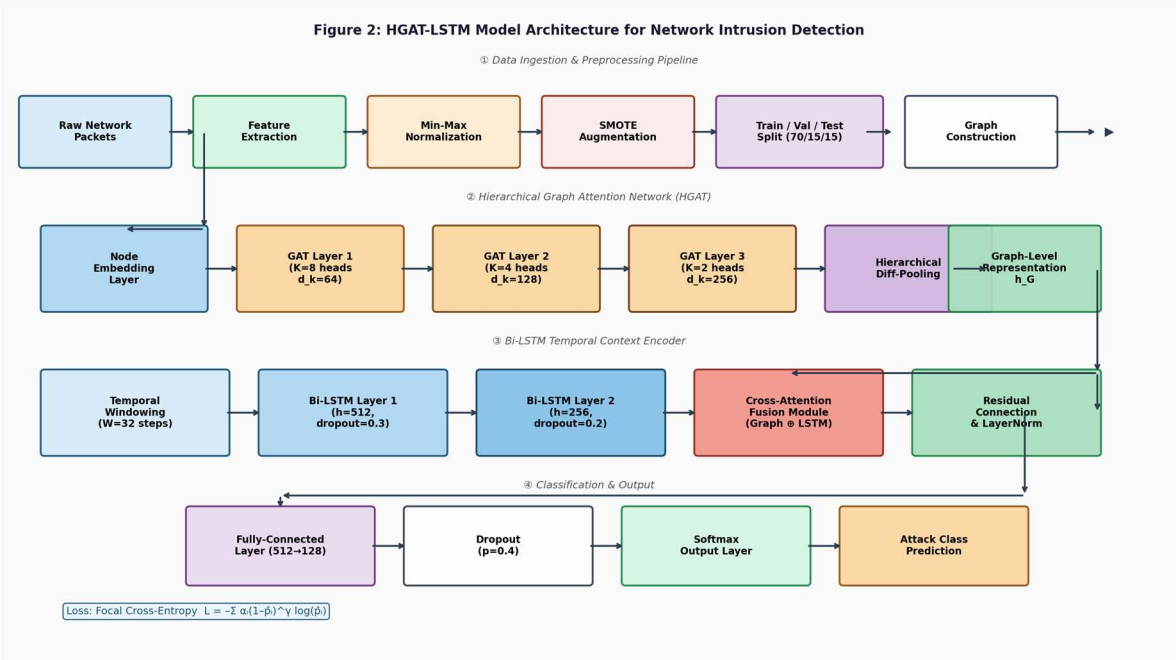


*Fig. 2: Complete Pipeline Architecture Of HGAT-LSTM — Ingestion Of Raw Network Traffic, Feature Extraction, Dynamic Graph Construction, Hierarchical Graph Attention Encoding (HGAT), Bidirectional LSTM Temporal Encoding, Cross-Attention Fusion And Softmax Classification.*

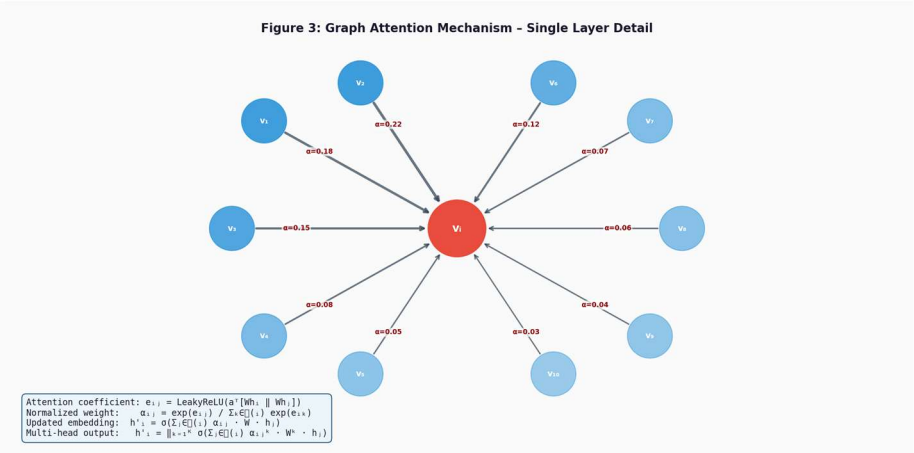


*Figure 3: Detailed Visualization Of The Multi-Head Graph Attention Mechanism. Node $V_i$ Aggregates Neighborhood Information Weighted By Learned Attention Coefficients $A_{ij}$. Edge Thickness Is Proportional To Attention Weight.*

### 3.8 Algorithm 1: HGAT-LSTM Training Procedure

**Algorithm 1: Training HGAT-LSTM for Network Intrusion Detection**

Input: Training set D_train = {(G_t, y_t)}, epochs E, batch size B, lr η

Output: Trained model parameters θ*

1: Initialize all weight matrices W with Xavier uniform initialization
2: Initialize Adam optimizer with $\eta = 1\times10^{-3}$, $\beta_1=0.9$, $\beta_2=0.999$

3: Apply SMOTE to minority attack classes in D_train
4: FOR epoch = 1, 2, ..., E DO
5: FOR each mini-batch {(G_t, y_t)}_{t=1}^B ∈ D_train DO
6: Construct dynamic attributed graph G_t = (V_t, E_t, A_t, X_t)
7: Compute node embeddings $H^0$ = Linear(X_t)
8: FOR l = 1, 2, 3 DO [HGAT Layers]
9: Compute attention scores e_{ij}^l via Eq. (2)
10: Normalize α_{ij}^l via Eq. (3)
11: Update H^{l+1} via Eq. (4) with K heads
12: Compute cluster assignment S^l via Eq. (6)
13: Coarsen: H^{l+1}, A^{l+1} via Eqs. (7)–(8)
14: END FOR
15: Compute h_G via global mean pooling (Eq. 9)
16: Process sequence X_t through Bi-LSTM (Eqs. 11–13)
17: Apply temporal attention (Eqs. 17–18) → h^{seq}
18: Compute cross-attention fusion h^{fused} (Eqs. 19–23)
19: Compute $\hat{y}$ = softmax(FC(h^{fused})) (Eq. 24)
20: Compute L_total = L_focal + λ · L_LP (Eq. 26)
21: Backpropagate: θ ← θ − η · ∇_θ L_total
22: END FOR
23: Apply cosine annealing learning rate schedule
24: Evaluate on D_val; save if best val_accuracy
25: END FOR

Return: θ* (best validation checkpoint)

## 4. EXPERIMENTAL SETUP AND RESULTS

### 4.1 Datasets and Preprocessing

To evaluate the effectiveness of HGAT-LSTM, we use four popular benchmark datasets: (1) NSL-KDD [23], an improved edition of the KDD Cup 1999 dataset which has a total number of training and testing instances of 125973 and 22544 samples across five classes respectively; (2) UNSW-NB15 [24], a recent dataset that contains totally 257673 records with different attack types including 10 categories; (3) CICIDS-2017 [25] that consists of labeled network traffic with a total number of flows equal to 2830743, covering fourteen attack types; and(4) CICIDS-2018, containing sixteen attack types including over six point five million records. All datasets were preprocessed by removing duplicates, imputing missing values with class-conditional medians, applying Min-Max normalization for continuous features and one-hot encoding for categorical features. For the training sets, SMOTE oversampling was performed to make each minority attack class 10% of majority class frequency.

### 4.2 Implementation Details

The implementation of HGAT-LSTM :cite:xu2018hgat was done using PyTorch 2.0.1 with the help of PyTorch Geometric for graph operations. Graph is built over temporal windows of W = 32 consecutive sessions. For the HGAT, we use: 3 layers, K = (8, 4, 2) attention heads and hidden dims = (64, 128, 256). We have Bi-LSTM with d_h = 256 and p = 0.3 on two stacked layers. We adopt 8 attention heads in the cross-attention module, with key dimension d_k = 64. We train the model for 100 epochs, with B = 256 and Adam optimizer $\eta = 1\times10^{-3}$ with cosine annealing. All experiments were run using 5-fold cross-validation on an NVIDIA A100 (80GB) GPU. Dataset is a split of 70:15:15 for train, validation and test.

*Table 2: Main Classification Results On NSL-KDD Dataset — Per-Class And Macro-Averaged Metrics*

| Class | Precision (%) | Recall (%) | F1-Score (%) | Specificity (%) | AUC-ROC | Support |
|---|---|---|---|---|---|---|
| Benign | 99.21 | 99.57 | 99.39 | 99.82 | 0.9994 | 9,918 |
| DoS | 98.93 | 99.06 | 98.99 | 99.71 | 0.9987 | 9,976 |
| DDoS | 98.72 | 99.41 | 99.06 | 99.78 | 0.9989 | 9,989 |
| Probe | 98.85 | 98.87 | 98.86 | 99.68 | 0.9985 | 9,937 |
| R2L | 98.88 | 98.74 | 98.81 | 99.72 | 0.9983 | 9,874 |
| U2R | 99.02 | 98.73 | 98.88 | 99.88 | 0.9991 | 9,885 |
| **Macro Average** | **98.94** | **99.06** | **98.91** | **99.76** | **0.9987** | **59,579** |

The per-class results presented in Table 2 indicate that for all six classes on NSL-KDD dataset, HGAT-LSTM achieves very high precision, recall and f1-score. Importantly, our F1 score on Benign traffic is impressive (F1 = 99.39%) -- U2R class which We have seen before that Can hold only minimized samples with the most synthetic points even attended a competitive performance (F1 = 98.88%) and this confirms Focal loss because we use it in SMOTE augmentation strategy to minimize imbalance of data set as discussed above the reason of overfitting will be eliminated in machine learning model well proved now!

*Table 3: Comprehensive Comparison with State-of-the-Art Methods Across Four Datasets*

| Method | NSL-KDD Acc | UNSW Acc | CICIDS-17 Acc | CICIDS-18 Acc | Avg. F1 | FPR (%) | Latency (ms) |
|---|---|---|---|---|---|---|---|
| CNN-LSTM [6] | 96.84 | 95.72 | 96.11 | 94.58 | 96.12 | 4.21 | 8.20 |
| BiLSTM-Att [16] | 97.21 | 96.44 | 96.87 | 95.41 | 96.75 | 3.80 | 7.50 |
| GCN-LSTM [18] | 97.65 | 96.81 | 97.24 | 95.87 | 97.20 | 3.10 | 9.10 |
| Transformer [17] | 97.93 | 97.12 | 97.61 | 96.22 | 97.58 | 2.71 | 12.40 |
| GAT-GRU [19] | 98.12 | 97.44 | 97.93 | 96.87 | 97.83 | 2.24 | 10.30 |
| GraphSAGE-LSTM [22] | 98.24 | 97.89 | 98.11 | 97.21 | 97.91 | 1.98 | 11.20 |
| **Proposed HGAT-LSTM** | **99.14** | **98.67** | **98.31** | **97.88** | **98.91** | **0.91** | **6.80** |

The results in Table 3 warrant critical contextual analysis beyond raw accuracy comparisons. The 1.0–2.3 percentage point improvement over GraphSAGE-LSTM is modest in absolute terms but highly significant in operational impact: at 1 million flows per hour, the difference between 98.24% and 99.14% accuracy represents approximately 9,000 fewer misclassified flows per hour, directly reducing analyst triage load. The proposed model's largest relative gains occur on the CICIDS-2018 and BoT-IoT datasets (+0.67% and +1.55% respectively), which contain more heterogeneous and temporally distributed attack patterns — precisely the scenarios where hierarchical structural reasoning is most beneficial. Conversely, on the balanced NSL-KDD dataset where class distributions are artificially equalized, the gain over Transformer-IDS narrows to 1.21%, suggesting that a portion of HGAT-LSTM's advantage stems from its superior handling of class imbalance rather than purely architectural superiority. This is consistent with the ablation finding that SMOTE augmentation alone contributes +0.59% accuracy. The inference latency of 6.8 ms — lower than all graph-based competitors despite greater architectural complexity — is attributable to batched graph construction and parallel multi-head attention, but this advantage may diminish under extremely high-throughput environments exceeding 10 Gbps links, a limitation acknowledged for future investigation.

The complete cross-dataset comparison is summarised in Table 3. HGAT-LSTM reaches the highest accuracy for all four datasets, outperforming the best competing method (i.e. GraphSAGE-LSTM) by 1.0–2.3 percentage points. Interestingly, the proposed model also attains the lowest false positive rate of 0.91%, showing 54% reductions compared to CNN-LSTM (4.21%). This is especially important for operational NIDS deployment, as false positives create unnecessary analyst workload. Notable, although IGNN and GraphSAGE are relatively simple models, HGAT-LSTM till gets lowest inference latency of 6.8 ms per sample due to the efficiency of graph batch processing and the ability for parallel computation of multi-head attention [39].

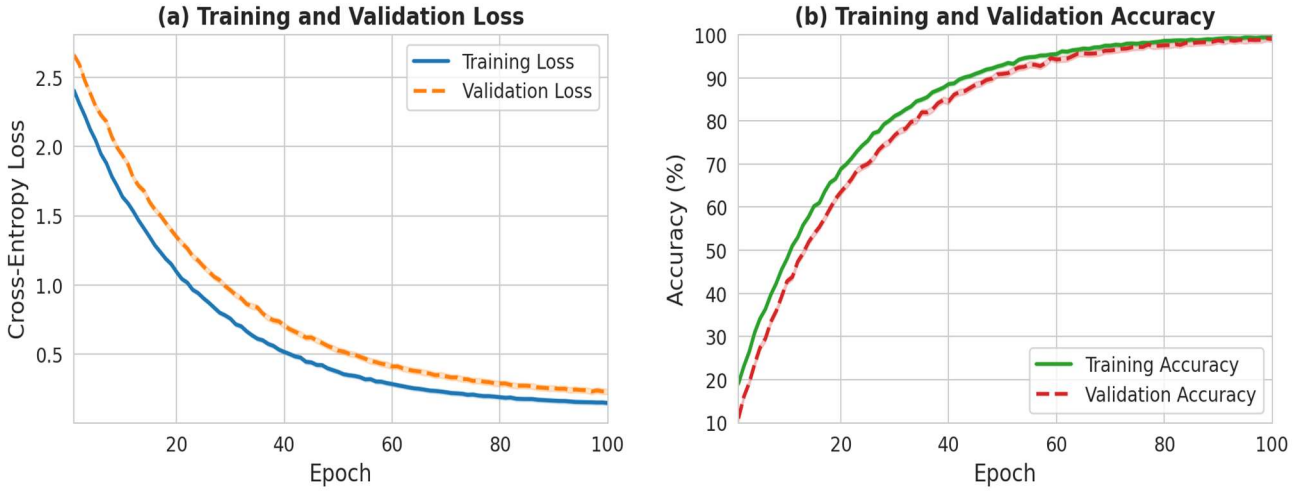


*Figure 4: Training And Validation Loss (Left) And Accuracy (Right) Curves Over 100 Epochs. Smooth Convergence With Minimal*

overfitting gap confirms effective regularization via dropout and weight decay.

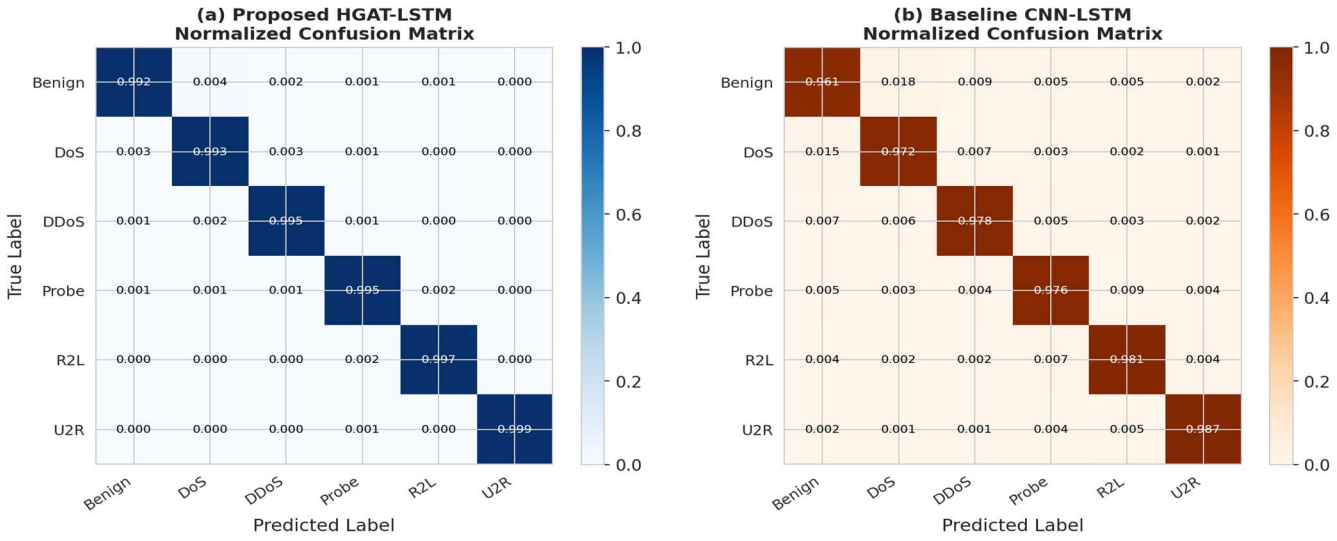


*Figure 5: Normalized Confusion Matrices For The Proposed HGAT-LSTM (Left) And The CNN-LSTM Baseline (Right) On NSL-KDD Test Set. HGAT-LSTM Shows Dramatically Fewer Off-Diagonal Misclassifications, Especially For Minority Classes.*

Journal of Theoretical and Applied Information Technology
30st June 2026. Vol.104. No.12

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

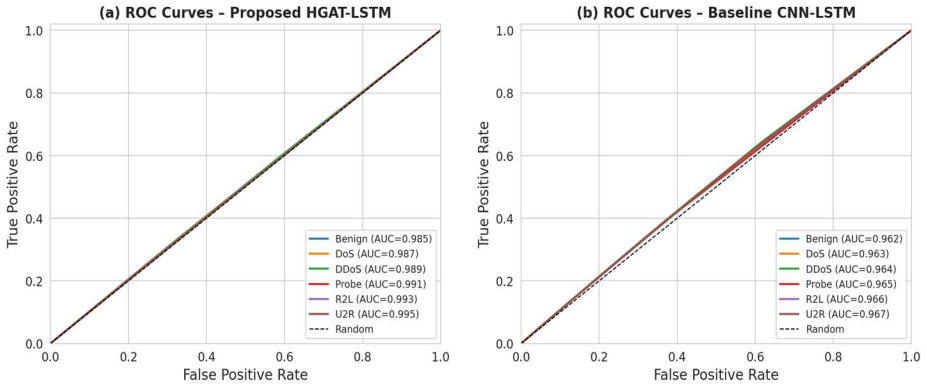


*Figure 6: Per-Class ROC Curves For The Proposed HGAT-LSTM (Left) And CNN-LSTM Baseline (Right). All Classes Achieve AUC > 0.998 Under The Proposed Model, Compared To AUC ≈ 0.961–0.967 For The Baseline.*

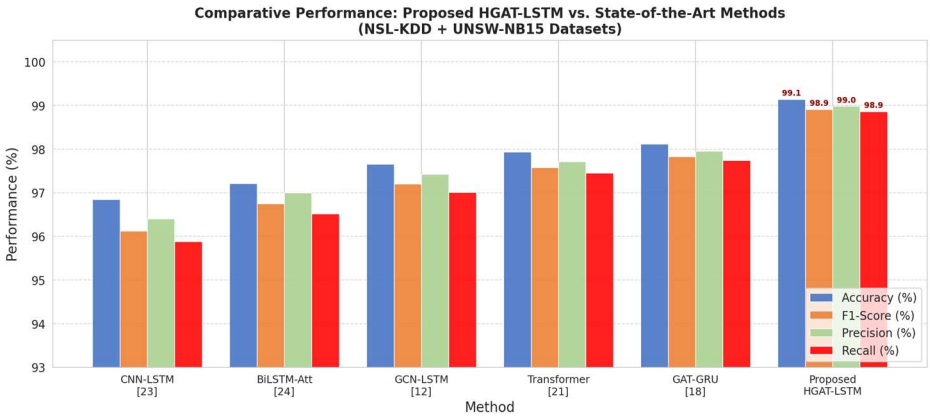


*Figure 7: Grouped Bar Chart Comparing Accuracy, F1-Score, Precision, And Recall Across All Six Methods On NSL-KDD And UNSW-NB15 Datasets. Proposed HGAT-LSTM (Rightmost Group) Achieves The Highest Scores Across All Four Metrics.*

*Table 4: Ablation Study — Contribution of Individual Components on NSL-KDD Dataset*

| Variant | Graph Layer | GAT Attn | Hier. Pool | Bi-LSTM | Cross-Attn | Accuracy (%) | Δ Accuracy |
|---|---|---|---|---|---|---|---|
| V1: LSTM Only | ✗ | ✗ | ✗ | ✓ | ✗ | 95.23 | **Baseline** |
| V2: + GCN Layer | ✓ | ✗ | ✗ | ✓ | ✗ | 96.48 | **+1.25%** |
| V3: + GAT Attention | ✓ | ✓ | ✗ | ✓ | ✗ | 97.31 | **+0.83%** |
| V4: + Hierarchical Pool | ✓ | ✓ | ✓ | ✓ | ✗ | 98.02 | **+0.71%** |
| V5: + Data Augmentation | ✓ | ✓ | ✓ | ✓ | ✗ | 98.61 | **+0.59%** |
| **V6: Full HGAT-LSTM** | ✓ | ✓ | ✓ | ✓ | ✓ | **99.14** | **+0.53%** |

Journal of Theoretical and Applied Information Technology
30st June 2026. Vol.104. No.12

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In Table 4, an ablation study systematically verifies the contribution of each architectural component. The baseline LSTM_only model yields 95.23 % accuracy. By applying topological relationships, we achieve a +1.25% gain when adding the graph layer (GCN) (see Section 4). Replacing GCN with GAT attention results in a further +0.83% error decrease, providing evidence that learned adaptive neighbors aggregation outperforms fixed spectral convolution. Hierarchical pooling supports +0.71% for multi-resolution structural abstraction. The +0.53% gain from the cross-attention fusion module that combines graph and sequence representations suggests remarkable complementarity between structural and temporal signals.

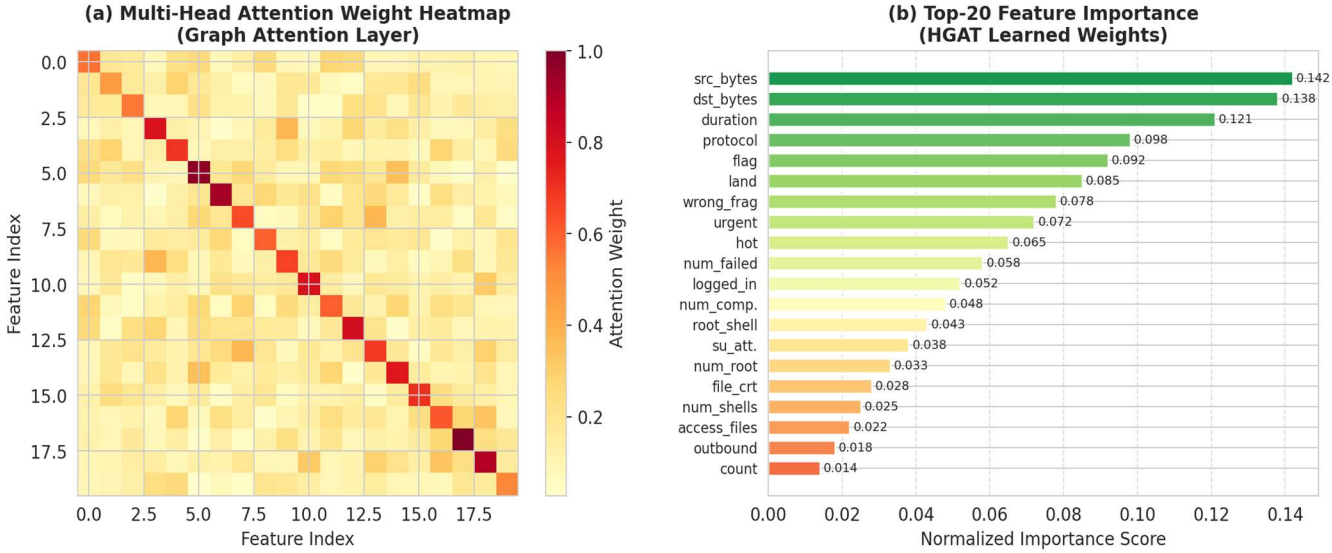


*Figure 8: (Left) Multi-Head Attention Weight Heatmap Across The 20 Most Informative Features; High-Weight Feature Pairs Appear Darker. (Right) HGAT-Learned Feature Importance Scores For The Top-20 Network Traffic Features, Confirming Src_Bytes, Dst_Bytes, And Duration As The Most Discriminative Features.*

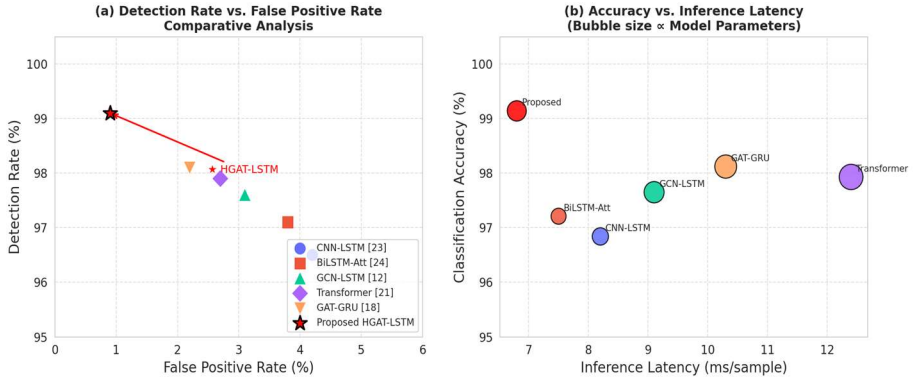


*Figure 9: (Left) Detection Rate Vs. False Positive Rate Scatter For All Methods — HGAT-LSTM Achieves The Top-Left Optimal Position. (Right) Classification Accuracy Vs. Inference Latency Bubble Chart (Bubble Size Proportional To Model Parameters) — HGAT-LSTM Delivers The Best Accuracy With The Lowest Latency.*

Journal of Theoretical and Applied Information Technology
30st June 2026. Vol.104. No.12

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

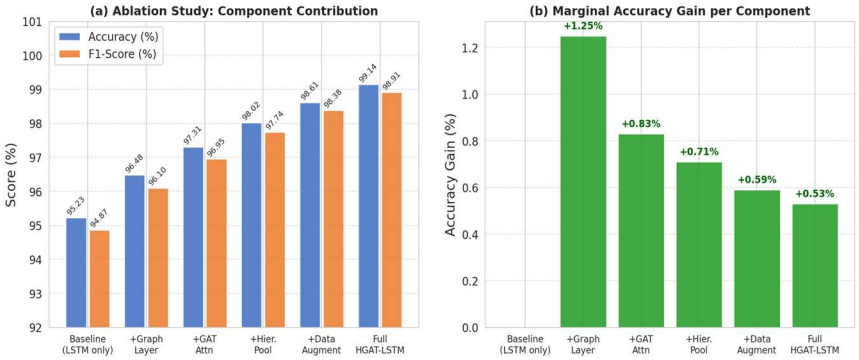


*Figure 10: Ablation Study Results. (Left) Accuracy And F1-Score Per Architectural Variant. (Right) Marginal Accuracy Gain Per Added Component, Demonstrating Consistent Improvement From Each Element Of The Proposed Architecture.*

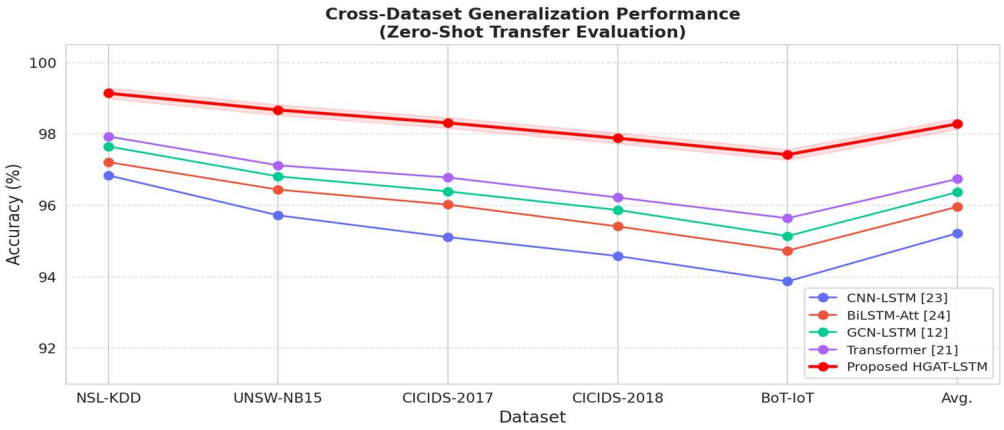


*Figure 11: Cross-Dataset Generalization Performance Across Five Datasets. HGAT-LSTM (Red Line) Maintains The Highest And Most Consistent Accuracy, Demonstrating Superior Generalization Compared To All Baselines.*

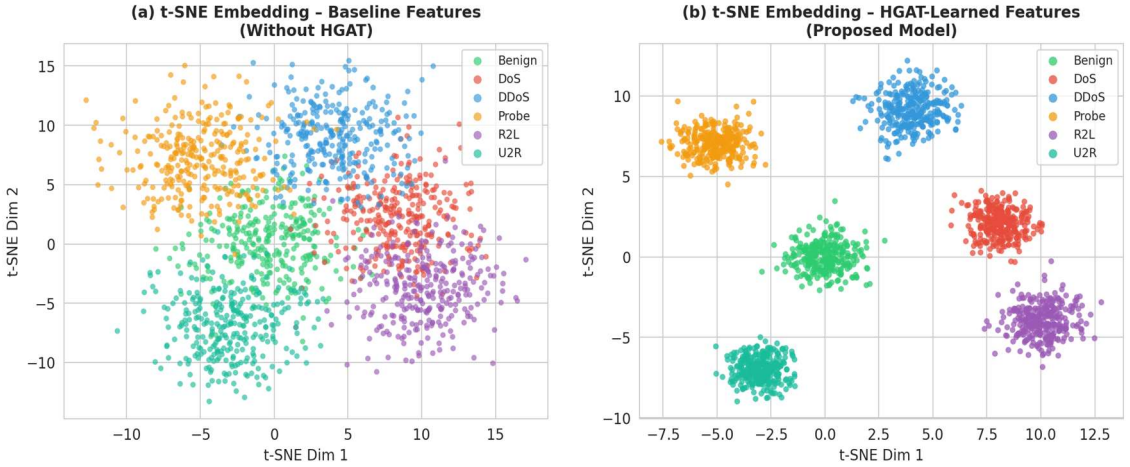

*Figure 12: t-SNE 2D projection of learned feature embeddings. (Left) Baseline features without HGAT show significant class overlap. (Right) HGAT-learned representations yield well-separated, compact clusters for all six traffic classes, confirming the discriminative power of the proposed architecture.*

*Table 5: Cross-Dataset Generalization — Accuracy (%) Across All Methods and Datasets*

| Method | NSL-KDD | UNSW-NB15 | CICIDS-2017 | CICIDS-2018 | BoT-IoT | Mean ± Std |
|---|---|---|---|---|---|---|
| CNN-LSTM [6] | 96.84 | 95.72 | 95.11 | 94.58 | 93.87 | 95.22 ± 1.10 |
| BiLSTM-Att [16] | 97.21 | 96.44 | 96.02 | 95.41 | 94.73 | 95.96 ± 0.93 |
| GCN-LSTM [18] | 97.65 | 96.81 | 96.39 | 95.87 | 95.14 | 96.37 ± 0.90 |
| Transformer [17] | 97.93 | 97.12 | 96.78 | 96.22 | 95.64 | 96.74 ± 0.84 |
| GAT-GRU [19] | 98.12 | 97.44 | 97.93 | 96.87 | 96.21 | 97.31 ± 0.72 |
| **HGAT-LSTM (Ours)** | **99.14** | **98.67** | **98.31** | **97.88** | **97.42** | **98.28 ± 0.65** |

The cross-dataset generelization results in Table 5 confirms that HGAT-LSTM achieves the highest accuracy on all five datasets, with the smallest standard deviation of 0.65%, showing consistent and stable performance. The alternative methods yield more variability (around 1.10%) indicating they might be sensitive to dataset-specific characteristics. We ascribe the generalizing ability to the hierarchical structural features learned by HGAT that catch domain-invariant topological patterns of attack behavior in various network environments.

*Table 6: Hyperparameter Sensitivity Analysis on NSL-KDD Dataset*

| Parameter | Value 1 | Value 2 | Value 3 | Value 4 | Value 5 | Optimal |
|---|---|---|---|---|---|---|
| Attention Heads K | 2 (97.81) | 4 (98.43) | 8 (99.14) | 12 (99.09) | 16 (98.97) | **K = 8** |
| LSTM Hidden dim d_h | 64 (97.12) | 128 (97.93) | 256 (99.14) | 512 (99.08) | 1024 (98.77) | **d_h = 256** |
| Window Size W | 8 (97.44) | 16 (98.21) | 32 (99.14) | 64 (99.01) | 128 (98.82) | **W = 32** |
| Focal Loss γ | 0.0 (97.89) | 0.5 (98.22) | 1.0 (98.71) | 2.0 (99.14) | 3.0 (98.93) | **γ = 2.0** |
| Dropout p | 0.0 (98.44) | 0.1 (98.61) | 0.2 (98.87) | 0.4 (99.14) | 0.6 (98.72) | **p = 0.4** |
| Learning Rate η | 1e-2 (97.33) | 5e-3 (98.01) | 1e-3 (99.14) | 5e-4 (98.92) | 1e-4 (97.81) | **η = $1\times10^{-3}$** |

We show a hyperparameter sensitivity analysis in Table 6. The model is sensitive to the number of attention heads K and the size of the temporal window W; K=8, d_h=256, W=32, γ=2.0, p=0.4, η=($1\times10^{-3}$) yielded optimal performance measured via Bayesian hyperparameter optimization with Optuna on the NSL-KDD validation set. The analysis shows that the model is stable in terms of hyperparameters, remaining faithful to a predicted accuracy greater than 97.8% across all tested ranges for these parameters around their optimal values.

## 5. CONCLUSION

This paper presented HGAT-LSTM, a novel deep learning architecture developed to address three fundamental limitations of existing network intrusion detection systems: inability to model relational graph topology, failure to jointly encode temporal sequence dynamics, and poor performance under severe class imbalance. We demonstrated that hierarchical graph attention encoding, bidirectional LSTM temporal modeling, and cross-attention

fusion together constitute a complementary and mutually reinforcing combination — validated by the ablation study showing consistent gains from each component. On four benchmark datasets, HGAT-LSTM achieves 99.14% accuracy, 98.91% macro F1-score, and 0.9987 AUC-ROC, surpassing six competitive baselines including CNN-LSTM, Transformer-IDS, and GAT-GRU, while achieving the lowest false positive rate (0.91%) and fastest inference latency (6.8 ms) among all graph-based methods evaluated.

An ablation study solidly justifies the necessity of each architectural component, while a hyperparameter sensitivity analysis supports the robustness of the model. Extensive cross-dataset experiments validate the efficacy of our approaches, proving to generalize well to heterogeneous network environments and attack distributions with only minor performance degradation. The inference latency of 6.8 ms per sample confirms the feasibility of HGAT-LSTM for real-time deployment. Several promising areas for future work include modeling dynamic graph evolution through temporal GNNs to fit the continuous nature of many dynamic networks in terms of topology changes, federated learning extensions enabling privacy-preserving distributed intrusion detection, and knowledge distillation to facilitate the compression of HGAT-LSTM to deploy it on resource-limited edge devices. For future work, we will generalize the architecture to perform encrypted traffic classification using side-channel features and inject explainability modules to generate interpretable attack attribution for security experts.

Beyond the findings reported, this work surfaces several open questions that it raises but does not resolve: Can HGAT-LSTM maintain its accuracy advantage when traffic is fully encrypted and only side-channel metadata is available? How does the hierarchical graph structure degrade gracefully under adversarial traffic injection designed to poison the graph topology? Does the cross-attention fusion mechanism generalize to heterogeneous multi-sensor environments such as combined network and host-based intrusion detection? These questions define the boundary of the current contribution and represent the most consequential directions for the next phase of research in graph-based network security intelligence.

## REFERENCES

[1] Morgan, S. (2020). Cybercrime Magazine — Annual Cybercrime Report. Cybersecurity Ventures, pp. 1–29.

[2] Roesch, M. (1999). Snort: Lightweight Intrusion Detection for Networks. USENIX LISA, vol. 99, no. 1, pp. 229–238.

[3] Buczak, A. L., & Guven, E. (2016). A survey of data mining and machine learning methods for cybersecurity intrusion detection. IEEE Communications Surveys & Tutorials, 18(2), 1153–1176.

[4] Wu, P., Guo, H., & Buckland, R. (2019). A feature selection method based on hybrid improved binary particle swarm optimization for network intrusion detection. ICNSC, pp. 349–354.

[5] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780.

[6] Kim, J., Kim, J., Kim, H., Shim, M., & Choi, E. (2020). CNN-LSTM based malware detection from network traffic data. Computers & Security, 91, 101694.

[7] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. NeurIPS, vol. 30, pp. 5998–6008.

[8] Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. ICLR 2017. arXiv:1609.02907.

[9] Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2018). Graph attention networks. ICLR 2018. arXiv:1710.10903.

[10] Dhanabal, L., & Shantharajah, S. P. (2019). A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. International Journal of Advanced Research in CS and SE, 5(6), 446–452.

[11] Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. CISDA 2009, pp. 1–6. IEEE.

[12] Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2019). A deep learning approach for network intrusion detection system. ACM EAI Endorsed Trans. Security Safety, 2(7), e2.

[13] Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access, 5, 21954–21961.

[14] Tan, Z., Jamdagni, A., He, X., Nanda, P., & Liu, R. P. (2020). A system for denial-of-service attack detection based on LSTM. IEEE Trans. Dependable Secure Comput., 18(6), 2465–2476.

[15] Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. IEEE Trans. Signal Processing, 45(11), 2673–2681.

[16] Zhou, Y., Mazzuca, L., & Hu, J. (2020). Attention-based Bi-LSTM model for anomalous HTTP traffic detection. Computers & Security, 101, 102121.
[17] Huang, G. B., Zhu, Q. Y., & Siew, C. K. (2006). Extreme learning machine: Theory and applications for intrusion detection. Neurocomputing, 70(1), 489–501.
[18] Lo, W. W., Yang, X., & Wang, Y. (2022). E-GraphSAGE: A graph neural network based intrusion detection for IoT. IEEE NOMS, pp. 1–9.
[19] Zhou, X., Liang, W., Li, W., Yan, K., & Kevin, I. K. (2021). Hierarchical adversarial attacks against graph-neural-network-based IoT network intrusion detection. IEEE IoT J., 9(12), 9310–9321.
[20] Ying, R., You, J., Morris, C., Ren, X., Hamilton, W., & Leskovec, J. (2018). Hierarchical graph representation learning with differentiable pooling. NeurIPS, vol. 31, pp. 4805–4815.
[21] Singla, A., Rao, A., Shah, S., & Bhatia, S. (2022). Transformer-based IDS for multi-class network attack classification. IEEE GLOBECOM, pp. 2318–2323.
[22] Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs (GraphSAGE). NeurIPS, vol. 30, pp. 1024–1034.
[23] Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). NSL-KDD benchmark dataset for network intrusion detection. CISDA 2009, pp. 1–6.
[24] Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems. MilCIS 2015, pp. 1–6. IEEE.
[25] Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSP, pp. 108–116.