GNN-Based Network Attack Traffic Classification Through Representation of Session Structure

Seung-Woo Nam
Computer and Information Science
Korea University
Sejong, Korea
nam131119@korea.ac.kr

Gyeong-Min Yu
Computer Science and Software Engineering
Korea University
Sejong, Korea
rudals2710@korea.ac.kr

Yun-Seong Jang
Computer and Information Science
Korea University
Sejong, Korea
brave1094@korea.ac.kr

Ju-Sung Kim
Computer and Information Science
Korea University
Sejong, Korea
jsung0514@korea.ac.kr

Ji-Min Kim
Computer Science and Software Engineering
Korea University
Sejong, Korea
illiard1209@korea.ac.kr

Myung-Sup Kim
Computer Science and Software Engineering
Korea University
Sejong, Korea
tmskim@korea.ac.kr

***Abstract*—Network attack traffic classification is a critical component of modern intrusion detection systems, yet most existing approaches rely on session-level statistical features such as packet counts, byte counts, and inter-arrival times. While effective in capturing aggregate flow behavior, these statistical representations abstract away the protocol semantics embedded within individual packet header fields, limiting their ability to discriminate between attacks that exhibit similar flow-level statistics but differ in their underlying protocol behavior. In this paper, we propose a Graph Neural Network (GNN)-based classification framework that represents network traffic as a host-session graph, where hosts are modeled as nodes and sessions between them as edges. Unlike prior GNN-based approaches that use statistical descriptors as edge features, we construct edge features directly from IP and transport-layer header fields aggregated over each session, without relying on application-layer payload contents. We evaluate three representative GNN architectures—GCN, GAT, and E-GraphSAGE—on three public benchmarks: CIC-IDS-2017, USTC-TFC-2016, and ToN-IoT. Under an identical graph structure and identical model configurations, replacing statistical edge features with our proposed header field-based features improves classification performance across all three backbones on CIC-IDS-2017 and USTC-TFC-2016, with macro-F1 gains of up to 35%, while performing comparably to the baseline on ToN-IoT, whose baseline features are native to the dataset's own connection log format. These results indicate that preserving fine-grained header field information can substantially enhance the discriminative power of GNN-based traffic classifiers, even without access to payload contents, although the magnitude of the benefit depends on how well the attack types of a dataset manifest at the IP and transport layers.**

***Index Terms*—network intrusion detection system, gnn, network session**

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. RS-2025-02219319, Development of Standards for Quantum Cryptography based Zero Trust Secure Network/Service and Control/Management against Quantum Computer Attacks) and by the Technology Innovation Program (TIPS) funded by the Ministry of SMEs and Startups (MSS, Republic of Korea) through the Korea Technology and Information Promotion Agency for SMEs (TIPA) under Grant No. RS-2025-25466990, Development and Standardization of a 5G/6G Network Cross-domain Observability Engineering Orchestrator.

I. Introduction

The rapid expansion of cloud services, Internet-of-Things (IoT) deployments, and 5G-enabled applications has dramatically increased both the volume and diversity of network traffic, making accurate classification of attack traffic a fundamental requirement for modern network management and security operations. To address this, a large body of research has applied machine learning (ML) and deep learning (DL) techniques to network intrusion detection, with most approaches relying on session-level statistical features—such as packet counts, byte counts, flow duration, and inter-arrival times—typically extracted by tools like CICFlowMeter [1]. These statistical descriptors are easy to compute and have driven substantial progress on public benchmarks.

However, statistical features inherently summarize each session into a fixed-length vector of aggregate quantities, which discards the fine-grained protocol semantics carried by individual packet header fields. Two sessions with nearly identical packet counts, byte counts, and timing distributions may still differ substantially in their TCP flag combinations, TTL patterns, or fragmentation behavior—differences that are often decisive for distinguishing benign traffic from attacks such as port scans, SYN floods, or protocol-exploiting intrusions. As a result, statistical-feature-based classifiers tend to struggle with attack variants that mimic benign flow-level statistics while exhibiting anomalous protocol behavior at the header level.

Recently, Graph Neural Networks (GNNs) have emerged as a promising paradigm for traffic classification, as they can jointly model host-level communication structure and session-level attributes. In particular, E-GraphSAGE [2] introduced a host-session graph formulation in which hosts are represented as nodes and sessions between them as edges, enabling edge-level attack classification. Yet, despite this structurally expressive formulation, existing GNN-based approaches still populate edge features with the same statistical descriptors used in conventional ML pipelines, thereby inheriting the very

limitation they could otherwise overcome.

In this paper, we argue that the bottleneck of GNN-based attack traffic classification lies not in the model architecture but in the edge feature representation. We propose to construct edge features directly from IP and transport-layer header fields aggregated over each session, deliberately excluding application-layer payload contents to preserve applicability to encrypted traffic and to maintain privacy-aware deployment. To rigorously isolate the effect of feature design, we conduct controlled experiments in which the graph structure, GNN architecture, and training configuration are held identical, and only the edge feature set is replaced. We evaluate three representative GNN models—GCN[3], GAT[4], and E-GraphSAGE—on three widely used public datasets: CIC-IDS-2017[5], USTC-TFC-2016[6], and ToN-IoT[7]. The contributions of this paper are summarized as follows:

- We identify and empirically characterize the limitations of statistical edge features in GNN-based attack traffic classification, and propose a header field-based edge feature representation that preserves IP- and transport-layer protocol semantics without relying on payload contents.
- We design a controlled evaluation protocol in which graph structure and model configurations are fixed, and only the edge feature set is varied, enabling a clean attribution of performance gains to the proposed feature design.
- Across three public datasets and three GNN architectures, the proposed features improve classification performance on CIC-IDS-2017 and USTC-TFC-2016 and perform comparably on ToN-IoT, whose baseline features are native to the dataset; we analyze why the benefit concentrates on attacks whose signal resides at the IP and transport layers.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 describes the proposed graph construction and header field-based edge feature design. Section 4 presents the experimental setup and results, and Section 5 concludes the paper.

II. Related Work

Research on network attack traffic classification has evolved along several distinct directions, each making different assumptions about how a session should be represented for downstream learning.

A. GNN-based Traffic Classification

Graph Neural Networks have recently emerged as a promising paradigm for traffic classification, as they jointly model host-level communication structure and session-level attributes that flat tabular representations cannot capture. Lo et al. proposed E-GraphSAGE, which formulates network traffic as a host-session graph—hosts as nodes, sessions as edges—and performs edge-level attack classification by extending the GraphSAGE message-passing rule to incorporate edge features. This formulation has since become a de facto standard for GNN-based intrusion detection, and subsequent studies have extended it with attention-weighted aggregation (GAT-based variants [8]), heterogeneous graph designs, and temporal extensions. Despite this architectural progress, almost all existing GNN-based approaches populate edge features with the same session-level statistical descriptors used in conventional ML pipelines, leaving the protocol semantics of individual packet header fields largely unexploited.

B. Statistical Feature-based GNN

Beyond the architectures discussed in Section 2.1, a broader stream of GNN-based intrusion detection research has explored alternative learning paradigms, graph constructions, and training objectives. Self-supervised graph learning approaches [9] introduce contrastive or reconstruction objectives over flow graphs to reduce dependence on labeled attack samples. NetFlow-centric GNN designs [10] adapt the host-session graph to operational flow record formats used by network operators. Despite the diversity of these design choices, the edge feature representation across this body of work has remained remarkably uniform: each edge is encoded as a fixed vector of session-level descriptors—such as packet counts, byte counts, flow duration, inter-arrival time statistics, connection state, and per-protocol transaction attributes from DNS, SSL, or HTTP—typically derived from flow-level exporters such as CICFlowMeter or from connection log formats such as Zeek-based logs adopted by datasets including ToN-IoT.

III. Proposed Method

This section presents a packet field-based GNN framework for attack traffic classification. It provides a detailed description of the criteria used to construct graphs from network traffic sessions.

A. Overall Framework

Figure 1 illustrates the proposed pipeline. Taking a set of pre-segmented sessions as input, the framework proceeds in three stages: (i) a host-session graph is constructed in which hosts are nodes and sessions are undirected edges; (ii) for each session, 17 IP and transport-layer header fields are extracted from every constituent packet and aggregated into a 68-dimensional edge feature vector via four session-level statistics; and (iii) the resulting graph is passed to a GNN backbone (GCN, GAT, or E-GraphSAGE) that performs edge-level attack classification.

B. Host-Session Graph Construction

Let $S = \{s_1, s_2, ..., s_M\}$ denote the set of input sessions, each consisting of an ordered sequence of packets exchanged between two hosts and characterized by its 5-tuple. Treating each session as a bidirectional unit between its two endpoint hosts, we represent the collection of sessions as an undirected graph $G = (V, E, X_E)$ as follows:

- Nodes V : the set of distinct host IP addresses observed as session endpoints in S. Each node $v \in V$ represents one host.

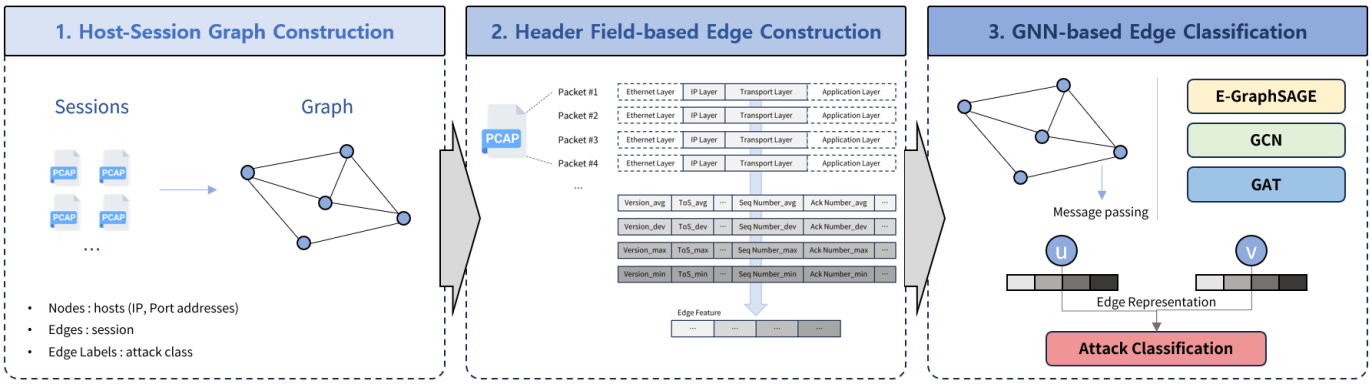

Fig. 1. Proposed pipeline

- Edges E : each session $s \in E$ between two hosts $u, v \in V$ induces an undirected edge $e_{uv} \in E$. Multiple sessions between the same host pair are represented as multi-edges, ensuring that no session-level information is lost through edge collapsing.
- Edge features $X_E \in \mathbb{R}^{|E| \times 68}$: each edge carries a feature vector derived from the header fields of the packets in the corresponding session, as defined in Section 3.3.

No node features are assigned a priori; node representations are learned by the GNN purely through message passing over the edge features, so that any performance difference between the two feature settings is attributable to the edge feature alone.

C. Header Field-based Edge Feature

The central contribution of this paper is the construction of edge features that preserve fine-grained protocol semantics from the IP and transport layers, in place of the session-level statistical descriptors used by prior GNN-based classifiers. For each packet p_i belonging to a session s, we extract the values of 17 header fields, drawn entirely from the IP and Transport Layers:

- IP layer (10 fields): Version, IHL (Internet Header Length), ToS (Type of Service), Total Length, Identification, Flags, Fragment Offset, TTL (Time-To-Live), Protocol, and Header Checksum.
- Transport layer (7 fields): Sequence Number, Acknowledgment Number, Header Length, Flags, Window Size, Checksum, and Urgent Pointer.

These fields are deliberately restricted to the IP and transport layers. We exclude Ethernet (link-layer) fields, which carry little information for cross-network attack classification, and application-layer payload contents, which are unavailable on encrypted traffic and raise privacy concerns in operational deployment.

For UDP sessions, the fields available in the UDP header (Header Length and Checksum) are mapped onto their corresponding feature slots, and the remaining TCP-specific fields are set to zero; since the IP-layer Protocol field is part of the feature vector, the model can distinguish such imputed zeros from genuine TCP values. Sessions of other transport protocols, for which the 17-field extraction is undefined, are excluded during preprocessing.

Let $f_k(p_i)$ denote the value of the k-th header field of packet p_i, for $k = 1, ..., 17$. For each field k, we compute four session-level statistics over the packets belonging to s using Equations (1)-(4):

$$\mu_k = \frac{1}{|s|} \sum_{p_i \in s} f_k(p_i) \tag{1}$$

$$\sigma_k = \sqrt{\frac{1}{|s|} \sum_{p_i \in s} (f_k(p_i) - \mu_k)^2} \tag{2}$$

$$\max_k = \max_{p_i \in s} f_k(p_i) \tag{3}$$

$$\min_k = \min_{p_i \in s} f_k(p_i) \tag{4}$$

The resulting edge feature is the concatenation of these statistics across all 17 fields, yielding a fixed-dimensional vector x_e as shown in Equation (5).

$$x_e = [\mu_1, \sigma_1, \max_1, \min_1, \ldots, \mu_{17}, \sigma_{17}, \max_{17}, \min_{17}] \in \mathbb{R}^{68}. \tag{5}$$

The proposed feature and the conventional statistical baseline (e.g., CICFlowMeter) both involve statistical aggregation, and it is therefore important to make clear how they differ. The two representations aggregate fundamentally different underlying quantities. Conventional statistical features compute statistics over session-level behavioral quantities—such as the total number of packets, total byte count, flow duration, inter-arrival time, and aggregate TCP flag counts—which describe the macroscopic shape of a flow but contain no information about the value of any individual protocol field.

In contrast, the proposed feature computes statistics over per-packet protocol field values—such as the distribution of TTL values, TCP flag combinations, window sizes, and IP identification numbers across the packets of a session—thereby preserving the protocol-level behavior of the session within a comparably sized vector. As a result, two sessions whose CICFlowMeter vectors are nearly indistinguishable may still differ substantially in the proposed representation, provided that their underlying protocol behavior differs.

D. GNN-based Classification

To evaluate the proposed edge feature across architecturally diverse settings, we adopt three representative GNN backbones, all configured for edge-level classification with identical hyperparameters across the proposed and baseline feature settings.

- GCN: performs symmetrically normalized neighbor aggregation derived from the spectral graph convolution framework. Since GCN in its original form operates on node features only, edge features are incorporated by first projecting each edge feature vector and adding it to the messages exchanged between incident nodes prior to neighbor aggregation.
- GAT: replaces the fixed normalization of GCN with learned attention coefficients, allowing each node to weight its incident edges according to their relative importance. Edge features are incorporated into the attention score computation, so that the model can attend more strongly to sessions whose header field patterns are informative for classification.
- E-GraphSAGE: is designed natively for edge-featured graphs and serves as the most direct architectural match for our setting. Its message-passing rule aggregates (h_v, x_e)pairs—the neighboring node embedding and the connecting edge feature—from each node's neighborhood, allowing edge features to participate in message construction without ad hoc projection.

After message passing, the representation of an edge e_{uv} is obtained by concatenating the final embeddings of its two incident nodes $z_{uv} = [h_u||h_v]$, which is then fed into a classification head that outputs class probabilities. Depending on the desired model capacity, the head can be instantiated as a simple linear layer followed by a softmax function, or as a multi-layer perceptron (MLP) when richer non-linear interactions between the two endpoint embeddings are required. In either case, the edge feature x_e does not bypass the GNN at the classification stage; it influences the final prediction only through the node embeddings h_u and h_v produced by message passing.

IV. Experiments

This section evaluates the proposed header field-based edge feature against the conventional statistical baseline under the controlled comparison.

A. Datasets

We evaluate the proposed framework on three publicly available network attack datasets: CIC-IDS-2017, USTC-TFC-2016, and ToN-IoT.

- CIC-IDS-2017 is an intrusion detection benchmark released by the Canadian Institute for Cybersecurity. It contains five days of captured traffic covering benign activity and a range of attacks including DoS/DDoS, brute force, web attacks, infiltration, and port scans.
- USTC-TFC-2016 consists of 10 classes of benign application traffic and 10 classes of malware traffic, originally provided as per-class packet capture files.
- ToN-IoT is a dataset focused on Internet-of-Things environments, containing attacks such as DDoS, scanning, injection, and ransomware.

Table I lists the class labels used for each dataset; we use all labels provided by each dataset without exclusion. The three datasets adopt different labeling taxonomies—attack techniques (CIC-IDS-2017), applications and malware families (USTC-TFC-2016), and high-level attack categories (ToN-IoT)—so the number of classes and degree of class imbalance differ, which should be kept in mind when comparing absolute Macro-F1 values across datasets.

TABLE I
Class Labels Used in Each Dataset

Dataset	Classes
CIC-IDS-2017	Benign, DoS Hulk, DoS GoldenEye, DoS Slowloris, DoS Slowhttptest, DDoS, PortScan, FTP-Patator, SSH-Patator, Bot, Web Attack–Brute Force, Web Attack–XSS, Web Attack–SQL Injection, Infiltration, Heartbleed
USTC-TFC-2016	*Benign (10):* BitTorrent, Facetime, FTP, Gmail, MySQL, Outlook, Skype, SMB, Weibo, WorldOfWarcraft; *Malware (10):* Cridex, Geodo, Htbot, Miuref, Neris, Nsis-ay, Shifu, Tinba, Virut, Zeus
ToN-IoT	Normal, Backdoor, DDoS, DoS, Injection, MITM, Password, Ransomware, Scanning, XSS

The same preprocessing pipeline is applied to all three datasets. Packets are pre-segmented into sessions using the standard 5-tuple key, as assumed in Section 3. Each session is assigned to a single class label according to its dataset-provided annotation, and an undirected host-session graph is constructed per the procedure in Section 3.2. Sessions belonging to incomplete or non-IP protocols, for which the proposed 17-field extraction is undefined, are discarded; this filtering is applied identically to both the proposed and baseline feature settings to ensure that the two settings operate on the same underlying session population.

B. Experimental Setup

For each session, we construct a baseline feature vector composed of 39 features extracted following the feature list of the ToN-IoT dataset. These features cover connection metadata, DNS, SSL, HTTP, and weird-event attributes commonly recorded in network connection logs; the complete list is provided in Table II. The baseline vector replaces x_e at the edge feature slot of the framework; all other components of the pipeline are held identical to the proposed setting.

We note that the two feature settings are not information-equivalent: they differ in dimensionality (68 vs. 39) and in information source, as the baseline includes application-layer transaction attributes (DNS, SSL, HTTP) that the proposed feature deliberately excludes. Our protocol therefore compares two complete feature extraction pipelines under an otherwise

identical framework—session population, graph structure, architecture, and training are all held fixed—so performance differences should be attributed to the choice of pipeline as a whole rather than to dimensionality alone.

TABLE II
LIST OF 39 BASELINE FEATURES

Category	Features
Connection	`proto`, `service`, `duration`, `src_bytes`, `dst_bytes`, `conn_state`, `missed_bytes`, `src_pkts`, `src_ip_bytes`, `dst_pkts`, `dst_ip_bytes`
DNS	`dns_query`, `dns_qclass`, `dns_qtype`, `dns_rcode`, `dns_AA`, `dns_RD`, `dns_RA`, `dns_rejected`
SSL	`ssl_version`, `ssl_cipher`, `ssl_resumed`, `ssl_established`, `ssl_subject`, `ssl_issuer`
HTTP	`http_trans_depth`, `http_method`, `http_uri`, `http_referrer`, `http_version`, `http_request_body_len`, `http_response_body_len`, `http_status_code`, `http_user_agent`, `http_orig_mime_types`, `http_resp_mime_types`
Weird	`weird_name`, `weird_addl`, `weird_notice`

For each dataset, each of the three GNN backbones (GCN, GAT, E-GraphSAGE) is trained twice—once with the proposed 68-dimensional header field edge feature and once with the 39-dimensional baseline feature—yielding 3 datasets × 3 backbones × 2 features = 18 runs in total. Within each dataset–backbone pair, the graph construction, model architecture, hyperparameters, and optimization settings are held identical across the proposed and baseline runs; only the edge feature vector differs. Hyperparameters (number of GNN layers, hidden dimension, dropout rate, learning rate, batch size, training epochs) are fixed in advance and applied uniformly to both feature settings.

Within each dataset, sessions are partitioned into a 70%/30% train/test split. The split is stratified by class label so that the test set preserves the class distribution of the dataset.

C. Results

Table III reports the experimental results corresponding to the setup defined above. Each cell shows the test-set performance of one of the three GNN backbones (E-GraphSAGE, GCN, GAT) on one of the three datasets, evaluated under both the baseline and proposed edge feature settings. Five metrics are reported: Accuracy, Precision, Recall, Macro-F1, and Weighted-F1.

Focusing on Macro-F1 as the primary metric, the proposed feature outperforms the baseline on all three GNN backbones for CIC-IDS-2017 and USTC-TFC-2016, with substantial gains observed on USTC-TFC-2016. On ToN-IoT, the proposed feature improves over the baseline with E-GraphSAGE but performs slightly below the baseline with GCN and GAT.

On CIC-IDS-2017, the proposed feature improves Macro-F1 from 0.30 to 0.49 with E-GraphSAGE, from 0.42 to 0.50 with GCN, and from 0.50 to 0.53 with GAT, while Accuracy improves from 0.94–0.95 to 0.98 across all three backbones. The largest improvement is observed on USTC-TFC-2016, where Macro-F1 increases from approximately 0.5 to 0.83–0.92 across the three backbones, with the other metrics showing comparable gains. On ToN-IoT, the two feature settings are more closely matched: the proposed feature improves Macro-F1 over the baseline with E-GraphSAGE (0.50 to 0.53), but falls slightly below it with GCN (0.56 vs. 0.59) and GAT (0.48 vs. 0.56). In terms of Accuracy, the proposed feature improves over the baseline with GCN (0.78 to 0.91) and shows a marginal gain with E-GraphSAGE (0.85 to 0.90), indicating that the two feature settings yield different trade-offs between Accuracy and Macro-F1 on this dataset. Overall, the proposed feature is most effective on CIC-IDS-2017 and USTC-TFC-2016, where it yields consistent improvements across all three GNN backbones, and is particularly strong on USTC-TFC-2016.

These improvements reflect what the two representations preserve. The attack types that benefit most express their defining behavior directly in header field values while remaining nearly invisible to flow-level statistics: port scans yield SYN-dominated flag distributions and minimal window sizes, and SYN floods exhibit degenerate flag and sequence-number patterns—all captured by the per-field statistics of Equation (5), yet collapsed into unremarkable packet and byte counts by the baseline. The large gains on USTC-TFC-2016 are likewise consistent with malware families exhibiting characteristic TTL, window size, and IP identification patterns induced by their host systems, whereas application-layer-borne attacks (e.g., web attacks) benefit less, as also reflected in the ToN-IoT results below.

The weaker results on ToN-IoT with GCN and GAT admit a natural explanation. The 39 baseline features are derived from ToN-IoT's own Zeek-based log format and are thus dataset-native, and its dominant attacks—injection, XSS, and password—manifest primarily at the application layer, where DNS, SSL, and HTTP attributes carry signal that header fields cannot capture. The opposite movements of Accuracy and Macro-F1 are consistent with class imbalance: the proposed feature gains on majority classes while ceding some minority, application-layer-centric ones. ToN-IoT thus delineates the boundary condition of the proposed feature: header information is most beneficial when the attack signal resides at the IP and transport layers, and application-layer attributes remain complementary.

Table IV reports the results of an ablation that examines how the IP-layer and transport-layer fields contribute individually. The edge feature is restricted to (i) the 10 IP-layer fields only (40-dimensional) or (ii) the 7 transport-layer fields only (28-dimensional). This ablation is conducted with E-GraphSAGE, which is the GNN backbone most directly aligned with our edge-featured graph setting.

The relative contribution of the two layers differs by dataset. On CIC-IDS-2017, the IP-only and transport-only configurations perform comparably, with Macro-F1 values of 0.41 and 0.43, respectively. On USTC-TFC-2016, the transport-only configuration is clearly more informative than IP-only

TABLE III
PERFORMANCE COMPARISON BETWEEN BASELINE AND PROPOSED EDGE FEATURES

Dataset	Feature	E-GraphSAGE					GCN					GAT				
		Acc.	**Prec.**	**Rec.**	**M-F1**	**W-F1**	**Acc.**	**Prec.**	**Rec.**	**M-F1**	**W-F1**	**Acc.**	**Prec.**	**Rec.**	**M-F1**	**W-F1**
CIC-IDS-2017	Baseline	0.94	0.42	0.30	0.30	0.92	0.94	0.44	0.42	0.42	0.93	0.95	0.55	0.50	0.50	0.94
	Proposed	**0.98**	**0.54**	**0.50**	**0.49**	**0.98**	**0.98**	**0.53**	**0.51**	**0.50**	**0.98**	**0.98**	**0.57**	**0.52**	**0.53**	**0.98**
USTC-TFC-2016	Baseline	0.74	0.60	0.57	0.57	0.72	0.74	0.57	0.55	0.55	0.72	0.70	0.55	0.49	0.49	0.65
	Proposed	**0.92**	**0.93**	**0.91**	**0.92**	**0.92**	**0.89**	**0.89**	**0.87**	**0.88**	**0.89**	**0.87**	**0.86**	**0.83**	**0.83**	**0.87**
ToN-IoT	Baseline	0.85	**0.71**	0.48	0.50	0.83	0.78	**0.77**	**0.59**	**0.59**	0.74	0.80	**0.69**	**0.55**	**0.56**	0.79
	Proposed	**0.90**	0.56	**0.54**	**0.53**	**0.90**	**0.91**	0.60	0.56	0.56	**0.90**	**0.88**	0.55	0.49	0.48	**0.87**

TABLE IV
FIELD-LAYER ABLATION ON E-GRAPHSAGE

Dataset	Layer		Acc.	Prec.	Rec.	M-F1	W-F1
	IP	**Trans.**					
CIC-IDS-2017	O	O	**0.98**	**0.54**	**0.50**	**0.49**	**0.98**
	O	X	0.97	0.46	0.40	0.41	0.97
	X	O	**0.98**	0.45	0.44	0.43	0.97
USTC-TFC-2016	O	O	**0.92**	**0.93**	**0.91**	**0.92**	**0.92**
	O	X	0.78	0.75	0.68	0.68	0.76
	X	O	0.88	0.77	0.76	0.76	0.87
ToN-IoT	O	O	**0.90**	**0.56**	**0.54**	**0.53**	**0.90**
	O	X	0.89	0.55	0.52	0.51	0.88
	X	O	0.87	0.49	0.40	0.41	0.85

(Macro-F1 0.76 vs. 0.68). On ToN-IoT, the pattern is reversed: IP-only outperforms transport-only (Macro-F1 0.51 vs. 0.41). These results indicate that neither layer alone dominates across datasets, and that the relative informativeness of the IP and transport layers depends on the characteristics of the dataset.

V. CONCLUSION

In this paper, we examined the question of where the principal performance gap in GNN-based attack traffic classification arises. Departing from prior work that has advanced the GNN architecture while retaining session-level statistical descriptors as edge features, we proposed a header field-based edge feature constructed by aggregating 17 IP and transport-layer header fields—each summarized by four session-level statistics—into a 68-dimensional vector. Under a controlled comparison in which the host-session graph, GNN backbone, hyperparameters, and training configuration were held identical, only the edge feature was varied between the proposed and baseline settings.

Across 3 datasets and 3 GNN backbones (GCN, GAT, E-GraphSAGE), the proposed feature consistently outperformed the baseline on CIC-IDS-2017 and yielded substantially larger improvements on USTC-TFC-2016, where Macro-F1 increased from approximately 0.5 to 0.83–0.92 across all three backbones. On ToN-IoT, whose dataset-native baseline features and application-layer-centric attacks favor the baseline, the two feature settings were comparable, with GCN and GAT favoring the baseline in Macro-F1. Our analysis indicated that the gains of the proposed feature concentrate on attack types whose discriminative signal resides in IP- and transport-layer field values, such as scanning and flooding behavior. An ablation on E-GraphSAGE further showed that neither the IP-layer nor the transport-layer fields alone dominate across datasets, and that the relative informativeness of the two layers depends on the dataset characteristics. Taken together, these findings indicate that the edge feature representation is a meaningful axis of improvement for GNN-based traffic classifiers, and that protocol-level header information can be a viable alternative to session-level descriptors—particularly given that the proposed feature relies on no application-layer payload information and thus remains applicable in encrypted and privacy-sensitive settings.

Several directions remain for future work. First, the dataset-specific alignment observed on ToN-IoT calls for further investigation into the interaction between feature design and dataset-native feature sets. Second, we plan to extend the proposed feature with additional fields and alternative aggregation schemes beyond the four statistics used in this work. Third, we will evaluate the proposed framework on encrypted traffic, where the payload-free property of the feature is essential.

REFERENCES

[1] A. H. Lashkari, G. Draper-Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of Tor traffic using time based features," in *Proc. 3rd Int. Conf. Information Systems Security and Privacy (ICISSP)*, 2017, pp. 253–262.

[2] W. W. Lo, S. Layeghy, M. Sarhan, M. Gallagher, and M. Portmann, "E-GraphSAGE: A graph neural network based intrusion detection system for IoT," in *Proc. IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2022, pp. 1–9.

[3] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," arXiv preprint arXiv:1609.02907, 2016.

[4] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," arXiv preprint arXiv:1710.10903, 2017.

[5] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Information Systems Security and Privacy (ICISSP)*, 2018, pp. 108–116.

[6] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *Proc. Int. Conf. Information Networking (ICOIN)*, 2017, pp. 712–717.

[7] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," *Sustainable Cities and Society*, vol. 72, p. 102994, 2021.

[8] W. W. Lo, G. Kulatilleke, M. Sarhan, S. Layeghy, and M. Portmann, "XG-BoT: An explainable deep graph neural network for botnet detection and forensics," *Internet of Things*, vol. 22, p. 100747, 2023.

[9] E. Caville, W. W. Lo, S. Layeghy, and M. Portmann, "Anomal-E: A self-supervised network intrusion detection system based on graph neural networks," *Knowledge-Based Systems*, vol. 258, p. 110030, 2022.

[10] D. Pujol-Perich, J. Suárez-Varela, A. Cabellos-Aparicio, and P. Barlet-Ros, "Unveiling the potential of graph neural networks for robust intrusion detection," *ACM SIGMETRICS Performance Evaluation Review*, vol. 49, no. 4, pp. 111–117, 2022.