A Multi-cycle Spatio-temporal Hypergraph and Dual-attention Residual Hypergraph Neural Network for Saturation Attack Detection in SDN

Sheng Lin[1,2,3], Xianrong Yang[1,2,3], Yi Chen[1,2,3], Chun Guo[1,2,3], Guowei Shen[1,2,3] and Yunhe Cui[1,2,3,✉]

[1] Engineering Research Center of Text Computing and Cognitive Intelligence, Ministry of Education, Guizhou University, Guiyang, 550000, Guizhou
[2] College of Computer Science and Technology, Guizhou University, Guiyang, 550000, Guizhou
[3] Guizhou Provincial Characteristic Key Laboratory of Software Engineering and Information Security, Guizhou University, Guiyang, 550000, Guizhou
yhcui@gzu.edu.cn

Abstract. Saturation attacks in Software-Defined Networking (SDN) have evolved into stealthy attack patterns, especially the Down-to-Up Timeout Probing and Dynamically Flow Table Overflowing (DUDFTO) attack. Current GNN-based detection approaches struggle to accurately model intricate traffic dynamics. Connecting flows via shared attributes easily mixes benign and malicious features, inadvertently mix conflicting information of benign flows and attack flows. Furthermore, the straightforward aggregation of flow node characteristics into hyperedge features in conventional HGNN-based models frequently leads to the loss of high-order correlations and cross-hyperedge information. To overcome the above limitations, this paper proposes ST-HGNN, a saturation attack detection method based on a multi-cycle spatio-temporal hypergraph and a dual-attention residual hypergraph neural network. ST-HGNN constructs ST-FlowGraph, a multi-cycle spatio-temporal hypergraph. Each node represents a network flow. The hypergraph contains three kinds of hyperedges: Source IP-Protocol hyperedges, destination port group hyperedges, and temporal window hyperedges. We further design ST-HGATNet, a dual-attention residual hypergraph neural network model. ST-HGATNet integrates a flow-level node learning module, a group-level hyperedge enhancement module, and a residual fusion and classification module to improve attack detection performance to refine node representations, capture group-level dependencies, and preserve critical and subtle anomalies information. Experimental results verify the detection performance of ST-HGNN, with its detection accuracy surpassing 97%.

Keywords: SDN, DUDFTO Attack, Hypergraph.

1 Introduction

1.1 Research Background

Software-Defined Networking (SDN) decouples the centralized control logic from the underlying data forwarding infrastructure [1], becoming a representative architecture for next-generation networks. SDN enables centralized management and programmable control, thereby improving resource orchestration and operational efficiency while enhancing network flexibility and scalability. However, this architectural design also renders the SDN highly susceptible to saturation attacks.

In OpenFlow-based SDN, switches rely on flow entries to forward packets. Upon receiving a packet that fails to match any active flow rules, the switch dispatches a Packet-in message to the controller to obtain forwarding instructions. Saturation attacks exploit this mechanism by generating a large number of unmatched packets, forcing the controller to install excessive flow entries. As a result, both switch and controller resources are exhausted in terms of computation and storage. This significantly degrades forwarding performance and may disrupt benign traffic.

Saturation attacks have evolved from high-rate flooding to more stealthy and adaptive forms. A representative example is the DUDFTO attack, which employs dynamic timeout probing and adaptive flow table overflowing attack to camouflage malicious behavior [2], substantially reducing the effectiveness of existing attack detection approaches.

1.2 Problems and Challenges

Recently, saturation attack detection approaches leveraging Graph Neural Networks (GNNs) have gained significant traction. However, existing saturation attack detection methods exhibit several critical limitations when detecting emerging saturation attacks such as DUDFTO.

Challenge 1: The graphs of existing GNN-based attack detection methods struggle to represent high-order correlations among traffic flows. Their construction is limited to pairwise edges, where each edge connects exactly two flow nodes. Given the characters of the saturation attack that it simultaneously utilizes many flow to launch attacks, these graphs fail to capture such cross-flow correlations.

Challenge 2: Benign and malicious flows often share attributes, such as spoofed IP addresses. When Most GNN-based attack detection methods aggregate features from these overlapping nodes, they inadvertently mix conflicting information of benign flows and attack flows, thus severely degrading attack detection performance.

Challenge 3: Most HGNN-based attack detection methods focus mainly on spatial correlations, significantly overlooking the temporal evolution of saturation attack flows. Given the significant temporal variations of saturation attack flows, existing hypergraph construction methods fail to capture such patterns.

1.3 Main Contributions

To tackle the aforementioned challenges, this paper proposes a saturation attack detection method based on multi-cycle spatio-temporal hypergraph and dual-attention residual HGNN. The primary contributions of this work are outlined below:

Contribution 1: We propose ST-FlowGraph, a multi-cycle spatio-temporal hypergraph. ST-FlowGraph replaces pairwise links with three semantic hyperedges: source IP-protocol clusters, destination port groups, and temporal windows. This structure can explicitly represent complex multi-flow interactions by grouping multiple related flows into single hyperedges.

Contribution 2: We propose ST-HGATNet, a residual HGNN. ST-HGATNet utilizes a flow-level node learning module, a group-level hyperedge enhancement module, and a residual fusion and classification module incorporates temporal feature enhancement across multiple cycles.

Contribution 3: Experiment results on two network topologies evaluate the performance of ST-HGNN. The results show that it achieves over 97% detection accuracy, significantly outperforming baseline methods.

The subsequent structure of this paper is arranged as follows. Section 2 reviews related work on SDN saturation attack detection. Section 3 presents the ST-HGNN method. Section 4 evaluates the performance of ST-HGNN. Section 5 concludes this work.

2 Related Work

2.1 Machine Learning (ML)-based Saturation Attack Detection Methods

ML-based detection methods usually improve detection performance by training classifiers on flow-level features. For instance, Tang et al. [3] proposed a hybrid model combining LightGBM and Logistic Regression to detect saturation attacks. Kong et al. [4] applied conventional ML algorithms to network attack detection. Mudgal et al. [5] introduced additional features, such as packet arrival frequency and content correlation score, together with the CatBoost algorithm to detect saturation attacks. Nevertheless, these methods mainly depend on traffic features from a single frame and fail to fully exploit the temporal and topological characteristics of network traffic.

2.2 Graph Neural Network (GNN)-based Saturation Attack Detection methods

GNN-based detection methods can utilize dependencies among network entities to detect network attacks. Early work like GCN-TC maps traffic to graphs [6], but its static graphs cannot represent the high-order spatio-temporal correlations. Ji et al. [7] designed multi edge communication flow graphs to model flow-device associations and detect saturation attacks, and Ran et al. [8] designed hypergraphs based on device hyperedges to represent higher order relationships among flows and devices. However,

they still lack explicit multi-cycle modeling and overlook the temporal evolution of saturation attack flows. This motivates ST-HGNN, which explicitly models both spatial and temporal dimensions while mitigating attribute-induced interference.

3 ST-HGNN

3.1 Overview of ST-HGNN

This paper proposes ST-HGNN to detect saturation attacks in SDN. ST-HGNN consists of two key components: a multi-cycle spatio-temporal hypergraph named ST-FlowGraph and a dual-attention residual HGNN model named ST-HGATNet. The overall framework is presented in Fig. 1, and the workflow is summarized as follows:

Step 1: Graph Data Collection. ST-HGNN collects and preprocesses network flow data from the SDN control and data planes, including data such as flow entry statistics,

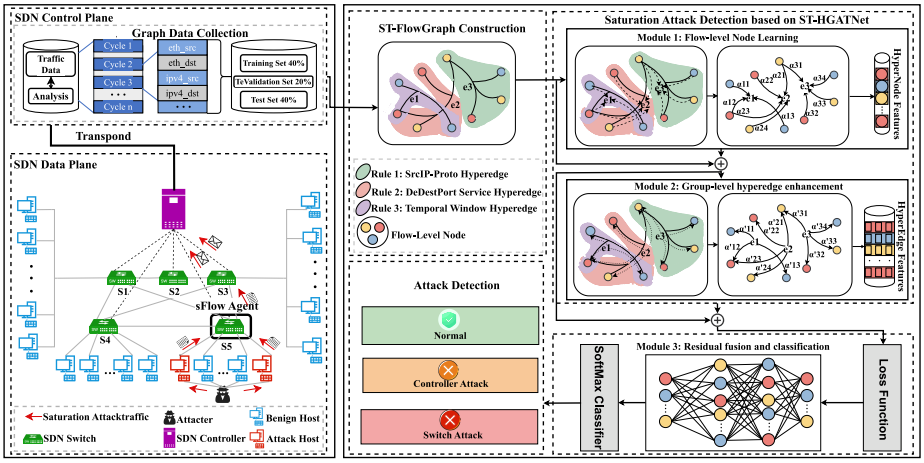


Fig. 1. Overview of ST-HGNN

port counters, and control messages. The preprocessed data is then sent to Step 2 for hypergraph construction.

Step 2: Hypergraph Construction. ST-HGNN builds a multi-cycle spatio-temporal hypergraph, ST-FlowGraph. In ST-FlowGraph, nodes correspond to traffic flows, and hyperedges constructed utilizing three semantic rules to capture spatio-temporal correlations among flows. The constructed hypergraph is then fed into ST-HGATNet for attack detection.

Step 3: Saturation Attack Detection based on ST-HGATNet. ST-HGNN designs a residual-enhanced cascaded model, ST-HGATNet. ST-HGATNet conducts flow-level classification for the detection of saturation attacks.

3.2 Graph Data Collection

Different with benign traffic flows, flows of emerging saturation attack such as DUDFTO exhibit distinct spatio-temporal characteristics. For attack flows target at SDN switches, their corresponding flow entries display low-rate but persistent characteristics, producing distinct temporal variations in flow duration and packet interarrival time. For attack flows target at the controller, they generate a large number of Packet-In messages with fluctuating rates. Meanwhile, DUDFTO also dynamically changes spoofed source IP addresses, which distorts MAC-IP mappings, and causes spatial overlap between attack and benign flows.

To capture the spatio-temporal behaviors of emerging saturation attacks, ST-HGNN uses OpenFlow messages to collect network data from SDN controllers and switches, including flow table statistics, port counters, and control messages. As shown in Table 1, ST-HGNN extracts statistical features from these data and performs data cleaning and standardization. This process prepares high-quality inputs for the subsequent spatio-temporal hypergraph construction.

Table 1. Feature Description

ID	Feature Name	Description
1	Duration	Total duration of a flow.
2	Packet Count	Total packet count of a single flow.
3	Byte Count	Total byte count of a single flow.
4	Packet Rate	Packet transmission rate.
5	Byte Rate	Byte transmission rate.
6	Mean Packet Size	Average bytes per packet in a flow.
7	Mean Interarrival Time	Average packet inter-arrival time.
8	Protocol	Protocol at the transport layer (e.g., TCP, UDP, ICMP).
9	Src IP Entropy	Entropy of source IP addresses within a flow group.
10	Src MacIp Ratio	Ratio of occurrences between source MAC and source IP.
11	Flow Table Match Count	Number of packets matched by the flow table.
12	Flow Table Lookup Count	Number of packets looked up in the flow table.
13	Active Entry Count	Number of active flow entries in the switch.
14	Match Ratio	Packet match ratio of the switch.
15	Available Space	Available flow table capacity.
16	Packet In Rate	Arrival frequency of Packet-In messages at the controller.
17	Flow Mod Rate	Transmission rate of Flow-Mod messages issued by the controller.
18	Packet In Rate Switch	Rate of Packet-In messages generated by a switch.
19	New Flow Install Rate	Rate of new flow entries installed in the switch.

3.3 Graph Data Collection

In SDN, the emerging saturation attack exhibits coordinated multi-flow behaviors that evolve over time. Most graphs constructed by existing GNN-based attack detection methods only capture pairwise relationships and fail to represent high-order relations among multiple flows. Moreover, the attributes of benign nodes may be affected by shared attributes with attack nodes. To resolve these deficiencies, we propose ST-FlowGraph, a multi-cycle spatio-temporal hypergraph. Key notations and definitions for ST-HGNN are as follows.

Definition 1. ST-FlowGraph. A ST-FlowGraph is a sequence of hypergraphs denoted as $\{\mathcal{G}^{(1)}, \mathcal{G}^{(2)}, \dots, \mathcal{G}^{(\mathcal{T})}\}$. For cycle t, $\mathcal{G}^{(t)} = \left(\mathcal{V}^{(t)}, \mathcal{E}^{(t)}, \mathrm{H}^{(\mathrm{t})}\right)$, where $\mathcal{V}^{(t)} = \{v_1, v_2, \dots, v_N\}$ is a node set. v_i is a node generated for the i-th network flow observed in cycle t. $\mathcal{E}^{(t)} = \{e_1, e_2, \dots, e_M\}$ denotes a hyperedge set. A hyperedge e_j is generated according to semantic rules. $\mathrm{H}^{(\mathrm{t})} \in \{0,1\}^{|\mathcal{V}^{(t)}| \times |\mathcal{E}^{(t)}|}$ is the incidence matrix. If a flow node v_i belongs to a hyperedge e_j, $\mathrm{H}_{\mathrm{ij}}^{(\mathrm{t})} = 1$. Otherwise, $\mathrm{H}_{\mathrm{ij}}^{(\mathrm{t})} = 0$.

Definition 2. Hyperedge Feature. For each hyperedge $e_j \in \mathcal{E}^{(t)}$, we define a feature vector $f_j \in R^{d_e}$. It summarizes the statistical properties of its member flows.

Based on the above definitions, three semantic rules are formulated to build the ST-FlowGraph.

Rule 1. If the source IP and protocol of the i-th network flow F_i are the same, and the number of distinct destination IPs in this group exceeds a threshold θ_{scan}, the corresponding flow node v_i is attached to the source IP-protocol hyperedge e_j. Thus, $H_{ij} = 1$.

Rule 2. If flow F_i is transmitted by the j-th destination port, and the total number of flows toward that port exceeds a threshold $\theta_{\mathrm{service}}$, the flow node v_i will be assigned to the destination port hyperedge e_j. Thus, $H_{ij} = 1$.

Rule 3. If F_i occurs in the l-th time window of cycle t, and the total number of flows N_l exceeds a threshold $\theta_{\mathrm{flow\_cnt}}$, the corresponding flow node v_i will be assigned to the temporal hyperedge e_j. Thus, $H_{ij} = 1$.

ST-FlowGraph groups related flows into shared hyperedges. This mechanism explicitly models complex interactions across protocol, service, and temporal dimensions. Thus, ST-FlowGraph helps capture high-order correlations. Additionally, the multi-cycle design preserves temporal continuity and avoids the complexity of fully dynamic graphs. Subsequently, ST-HGATNet leverages these structural correlations to learn discriminative flow representations.

3.4 Saturation Attack Detection based on ST-HGATNet

This section proposes ST-HGATNet, a dual-attention residual HGNN model. ST-HGATNet consists of three modules: (1) a flow-level node learning module, (2) a group-level hyperedge enhancement module, and (3) a residual fusion and classification module.

Module 1: Flow-level node learning. In ST-HGATNet, each hyperedge groups flows by protocol, service, and temporal relations. This module captures node contri-

butions within groups, and also captures the influence of groups on individual nodes. It uses the following two hypergraph attention mechanisms:

Node-to-hyperedge aggregation attention. This layer forms hyperedge representations from nodes, emphasizing nodes with greater contribution to group behaviors. The attention coefficient between node v_i and hyperedge e_j is defined as

$$\alpha_{ij} = \frac{\exp\left(\sigma\left(a_1^{\top}\left[W_1 x_i \oplus U_1 f_j \oplus V_1 t_j\right]\right)\right)}{\sum_{k\in\mathcal{N}_j} exp\left(\sigma\left(a_1^{\top}\left[W_1 x_k \oplus U_1 f_j \oplus V_1 t_j\right]\right)\right)}, \tag{1}$$

where x_i is the node feature. f_j is the hyperedge initial feature. t_j is a learnable type embedding. W_1, U_1, V_1 are projection matrices for the node, hyperedge, and type features, respectively. σ is LeakyReLU. a_1 is a learnable attention weight vector, and $\mathcal{N}_j$ is the node set in hyperedge e_j. The updated hyperedge representation is

$$h_j = \sum_{i\in\mathcal{N}_j} \alpha_{ij} W_1 x_i. \tag{2}$$

Hyperedge-to-node propagation attention. Subsequently, we propagate hyperedge information back to nodes. We employ a second attention mechanism to weigh the influence of each hyperedge on a node. The attention coefficient from hyperedge e_j to node v_i is calculated as

$$\beta_{ij} = \frac{\exp\left(\sigma\left(a_2^{\top}\left[W_2 x_i \oplus U_2 h_j \oplus (x_i \odot h_j)\right]\right)\right)}{\sum_{k\in\mathcal{M}_i} \exp\left(\sigma\left(a_2^{\top}\left[W_2 x_i \oplus U_2 h_k \oplus (x_i \odot h_k)\right]\right)\right)}, \tag{3}$$

$\odot$ denotes the element-wise product and $\oplus$ denotes concatenation. W_2, U_2 are projection matrices for node and hyperedge features. a_2 is another learnable attention weight vector, and $\mathcal{M}_i$ is the set of hyperedges containing v_i. The updated node feature is

$$x_i' = \sum_{j\in\mathcal{M}_i} \beta_{ij} U_2 h_j. \tag{4}$$

Using this attention mechanism, we select group-level information for each node while suppressing interference from shared attributes.

Module 2: Group-level hyperedge enhancement. DUDFTO varies its attack behavior across cycles and its flow patterns changes accordingly. Hyperedge features h_j from Module 1 capture only intra-cycle patterns and lack temporal context. To address these limitations, we connect identical hyperedges across consecutive cycles to aggregate temporal features. For a hyperedge e_j in cycle t, we identify its counterparts in cycles $t-1$ and $t+1$, then combine their features $h_j^{(t-1)}$ and $h_j^{(t+1)}$ via a lightweight temporal aggregation:

$$\widetilde{h_j^{(t)}} = h_j + \gamma \cdot \left(h_j^{(t-1)} + h_j^{(t+1)}\right), \tag{5}$$

where $\widetilde{h_j^{(t)}}$ is the enhanced feature of hyperedge e_j in cycle t. γ is a learnable scalar controlling the contribution of adjacent cycles. This linear fusion adds temporal con-

text while preserving original semantics, enabling the model to track the evolution of attack patterns.

Module 3: Residual fusion and classification. To preserve original flow information and stabilize training, ST-HGATNet introduces two residual connections.

First residual connection. The original node features x_i are obtained from ST-FlowGraph. The hyperedge representations h_j are obtained from the first layer of Module 1. These two are combined to form the fused representation $\widetilde{x_i}$:

$$\widetilde{x_i} = x_i + \frac{1}{|\mathcal{M}_i|}\sum_{j\in\mathcal{M}_i} h_j \,, \tag{6}$$

where $\mathcal{M}_i$ is the set of hyperedges containing node v_i. $\widetilde{x_i}$ serves as the input node feature for the second layer of Module 1.

Second residual connection. Node representations x_i'' are derived from the output of the second layer in Module 1, then they are combined with the augmented representation $\widetilde{x_i}$:

$$z_i = x_i'' + \widetilde{x_i}. \tag{7}$$

z_i fuses original flow features with group-level context and is used for final classification. The final node embeddings z_i are fed into a two-layer perceptron for classification:

$$\hat{Y} = \mathrm{softmax}(W_c\,\mathrm{ReLU}(W_b Z + b_b) + b_c), \tag{8}$$

where Z is the matrix stacking all z_i. W_b and W_c are weight matrices, b_b and b_c are biases. The output $\hat{Y}$ is a probability distribution over three classes: normal flows, saturation attack flows toward controllers, and saturation attack flows target at switches. To mitigate the class imbalance problem, we employ a weighted cross-entropy loss:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{3} w_c\; y_{i,c}\log\widehat{y_{i,c}} \,, \tag{9}$$

where N is the number of flow nodes. w_c is the weight for class c. $y_{i,c}$ is the ground-truth indicator, and $\widehat{y_{i,c}}$ is the predicted probability.

4 Experimental Evaluation

4.1 Experiment Setup

(1) Network Topology: We build two topologies using Ryu and Mininet. Topo1 is a Fat-tree ($k = 4$) with 4 core, 8 aggregation, and 8 edge switches. Each edge switch connects two hosts. Topo2 comes from the Topology Zoo dataset, representing a 2011 U.S. backbone segment with 24 switches and 28 links. Each switch attaches to one host.

(2) Dataset: We use packets of IMC-10 as background packets and replay these packets on two topologies. Switch-targeted saturation attacks are launched using DUDFTO, while Controller-targeted attacks are generated using Scapy. For Topo1, we collect 20905 flows across three categories: normal flows, switch-targeted saturation

attack flows, and controller-targeted saturation attack flows. For Topo2, we collect 15933 flows. We split the dataset into training dataset (40%), validation dataset (20%), and testing dataset (40%).

(3) Baselines: In order to comprehensively evaluate ST-HGNN, we de-sign three experimental evaluation, including attack detection performance of ST-HGNN and ablation studies. For evaluating attack detection performance of ST-HGNN, we compare ST-HGNN with FT-Master [3], GCN-TC [6], NER-SAGE [7], HINT [8], FT-GCN [9], SFTO-Guard [10], and BRS-CNN [11]. For ablation studies, we construct several model variants by removing or replacing in-dividual components. ST-HGNN performs three-class classification. We mainly evaluate ST-HGNN using Accuracy (Acc), Macro-Precision (MP), Macro-Recall (MR), Macro-F1 (MF1), Weighted Precision (WP), Weighted Recall (WR), and Weighted F1 (WF1).

4.2 Attack Detection Performance of ST-HGNN

This section evaluates the overall detection performance of ST-HGNN. We compare ST-HGNN with representative machine learning and graph neural network-based attack detection methods.

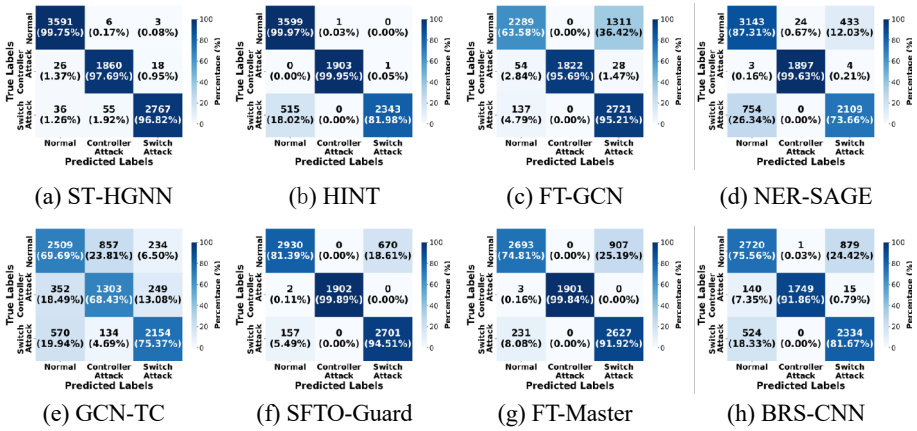
As shown in Tables 2 and 3, ST-HGNN achieves the best performance on both topologies. On Topo1, ST-HGNN reaches 0.9828 ACC, 0.9810 Macro-F1, and 0.9828 Weighted-F1. On Topo2, ST-HGNN achieves 0.9812 ACC, 0.9797Macro-F1, and 0.9812 Weighted-F1. In comparison, the best-performing baseline method HINT achieves a Macro-F1 score of 0.9445 on Topo1 and 0.9207 onTopo2. These results are significantly lower than ST-HGNN. This indicates that ST-HGNN maintains stable detection performance across different topologies.

Figures 2 and 3 present confusion matrices for different flow categories. For normal

Table 2. Detection Performance Comparison in Topo1

Mode	ACC	MP	MR	MF1	WP	WR	WF1
ST-HGNN	**0.9828**	**0.9813**	**0.9809**	**0.9810**	**0.9829**	**0.9828**	**0.9828**
SFTO-Guard	0.9009	0.9166	0.9193	0.9143	0.9099	0.9009	0.9011
HINT	<u>0.9382</u>	<u>0.9580</u>	<u>0.9397</u>	<u>0.9445</u>	<u>0.9458</u>	<u>0.9382</u>	<u>0.9372</u>
FT-Maste	0.8635	0.8878	0.8886	0.8821	0.8779	0.8635	0.8637
FT-GCN	0.8170	0.8644	0.8483	0.8392	0.8541	0.8170	0.8157
NER-SAGE	0.8544	0.8739	0.8687	0.8699	0.8549	0.8544	0.8532
BRS-CNN	0.8136	0.8421	0.8303	0.8344	0.8207	0.8136	0.8155
GCN-TC	0.7135	0.7054	0.7117	0.7062	0.7233	0.7135	0.7166

The optimal results are highlighted in bold, and the suboptimal results are underlined.



(a) ST-HGNN (b) HINT (c) FT-GCN (d) NER-SAGE

(e) GCN-TC (f) SFTO-Guard (g) FT-Master (h) BRS-CNN

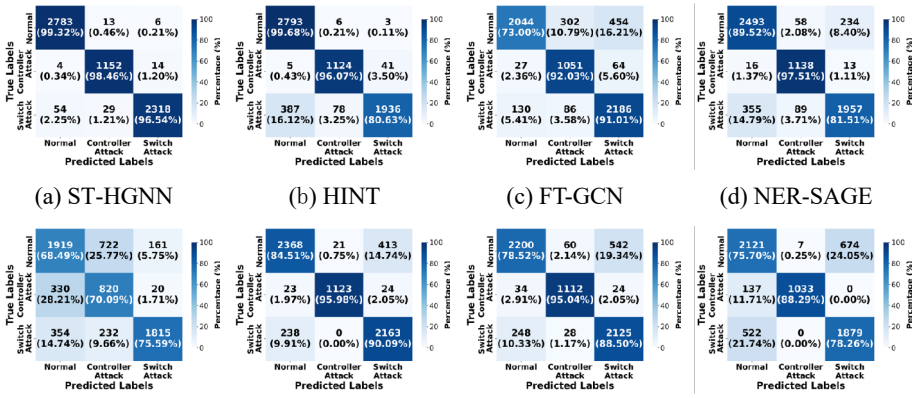
Fig. 2. Confusion matrices of different detection methods on Topo1.

flows, HINT achieves 99.97% and 99.68% accuracy on Topo1 and Topo2, slightly higher than ST-HGNN's 99.75% and 99.32%. This marginal degradation is a trade-off for enhanced sensitivity. ST-HGNN's fine-grained spatio-temporal modeling occasionally misclassifies bursty benign flows that temporally resemble camouflaged attacks. For controller attacks, ST-HGNN achieves 97.69% accuracy on Topo1, which is slightly lower than SFTO-Guard's 99.89%. ST-HGNN reaches98.76% and outperforming all other methods on Topo2. More importantly, for switch-related saturation attacks, ST-HGNN achieves 96.82% and 96.54% accuracies, significantly outperforming HINT (81.8% and 80.63%).

Table 3. Detection Performance Comparison in Topo2

Mode	ACC	MP	MR	MF1	WP	WR	WF1
ST-HGNN	**0.9812**	**0.9786**	**0.9811**	**0.9797**	**0.9813**	**0.9812**	**0.9812**
SFTO-Guard	0.8872	0.9048	0.9019	0.9026	0.8897	0.8872	0.8931
HINT	<u>0.9184</u>	<u>0.9250</u>	<u>0.9213</u>	<u>0.9207</u>	<u>0.9246</u>	<u>0.9182</u>	<u>0.9167</u>
FT-Maste	0.8531	0.8676	0.8735	0.8686	0.8573	0.8531	0.8528
FT-GCN	0.8324	0.8225	0.8535	0.8294	0.8471	0.8322	0.8316
NER-SAGE	0.8796	0.8813	0.8951	0.8869	0.8798	0.8794	0.8782
BRS-CNN	0.7897	0.8301	0.8075	0.8178	0.7951	0.7897	0.7915
GCN-TC	0.7146	0.7029	0.7139	0.6975	0.7518	0.7143	0.7254

The optimal results are highlighted in bold, and the suboptimal results are underlined.



(a) ST-HGNN (b) HINT (c) FT-GCN (d) NER-SAGE

(e) GCN-TC (f) SFTO-Guard (g) FT-Master (h) BRS-CNN

Fig. 3. Confusion matrices of different detection methods on Topo2.

4.3 Ablation Study

We conducted an ablation study to assess the contribution of each component within ST-HGNN. Four variants were compared to the full model: (1) replacing HyperCon with GCN (w/o STConv), (2) replacing Hypergraph Attention with mean aggregation (w/o STAttn), (3) removing Residual connections (w/o STResLink).

As shown in Table 3, the full model achieves 0.9914 accuracy and 0.9903 Macro-F1 on Topo1. Replacing STConv with GCN causes a significant drop of 2.94% in accuracy and 2.99% in Macro-F1, confirming that STConv plays an essential role in capturing high-order correlations among flows. Replacing STAttn with mean aggregation reduces the accuracy by 1.83% and Macro-F1 by 1.83%, indicating that the attention

Table 4. Ablation study results in Topo1 and Topo2

Topology	Metric	Full Model	w/o STConv	w/o STAttn	w/o TResLink
Topo1	ACC	**0.9914**	0.9620	0.9731	0.9787
	MP	**0.9905**	0.9666	0.9723	0.9794
	MR	**0.9902**	0.9591	0.9710	0.9778
	MF1	**0.9903**	0.9604	0.9720	0.9791
	WP	**0.9910**	0.9609	0.9722	0.9798
	WR	**0.9904**	0.9612	0.9708	0.9795
	WF1	**0.9903**	0.9608	0.9719	0.9793
Topo2	ACC	**0.9869**	0.9535	0.9658	0.9724
	MP	**0.9771**	0.9548	0.9671	0.9736
	MR	**0.9891**	0.9524	0.9654	0.9720
	MF1	**0.9829**	0.9535	0.9661	0.9724
	WP	**0.9872**	0.9545	0.9673	0.9736
	WR	**0.9869**	0.9543	0.9655	0.9745
	WF1	**0.9862**	0.9549	0.9664	0.9758

The optimal results are highlighted in bold, and the suboptimal results are underlined.

mechanism effectively mitigates the negative transfer effect caused by shared attributes between benign and attack flows. Removing the residual connections leads to smaller but still noticeable declines of 1.27% in accuracy and 1.12% in Macro-F1, demonstrating that residual connections contribute to training stability and feature preservation. Similar trends are observed on Topo2. These results demonstrate that STConv is essential for capturing spatiotemporal dependencies among flows, while STAttn effectively mitigates negative transfer. In addition, STResLink contributes to training stability.

5 Experimental Evaluation

In this paper, we propose ST-HGNN, a saturation attack detection method. We design a new multi-cycle spatio-temporal hypergraph called ST-FlowGraph to represent high-order flow correlations.

Furthermore, we also design a dual attention residual hypergraph neural network named ST-HGATNet, which uses dual attention to suppress negative transfer and captures temporal evolution for discriminative flow representation learning. The experimental results demonstrate that ST-FlowGraph can effectively represent high-order relationships among flows. Meanwhile, ST-HGATNet successfully captures spatio-temporal depen-dencies and alleviates negative transfer. These advantages enable ST-HGNN to detect novel SDN saturation attacks efficiently, especially showing strong capability in detecting saturation attacks.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (NSFC) under the Grant No. 62462010, the Guizhou Provincial Science and Technology Plan under the Grant No. Qian Ke He Zhongda Zhuanxiang Zi[2024]014, the Science and Technology Program of Guizhou Province, China under Grant No. ZK[2023] Zhongdian 011, the Big Data Security and Network Security Innovation Team of Guizhou Provincial High Education Institution, China under Grant No. [2023]052, and the Science and Technology Program Project of Guiyang National High-tech Zone ZhuGaoXinJieBang [2025]12.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Open Networking Foundation, “Software-defined networking: The new norm for networks,” ONF White Paper, 2012.
2. J. Li, Y. Cui, Y. Chen, G. Shen, C. Guo, and Q. Qian, “The DUDFTO Attack: Towards Down-to-up timeout probing and dynamically flow table overflowing in SDN,” *IEEE Trans. Netw. Service Manag.*, 2025, doi: https://doi.org/10.1109/TNSM.2025.3574260.
3. D.Tang, C. Gao, W. Liang, J. Zhang, and K. Li, “FTMaster: A detection and mitigation system of low-rate flow table overflow attacks via SDN,” *IEEE Trans. Netw. Service Manag.*, vol. 20, no. 4, pp. 5073–5084, 2023, doi: https://doi.org/10.1109/TNSM.2023.3290597.

4. D. Kong, X. Chen, C. Wu, Y. Xiang, and K. Li, "RDefender: A lightweight and robust defense against flow table overflow attacks in SDN," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 9436–9451, 2024, doi: https://doi.org/10.1109/TIFS.2024.3472477.
5. A. Mudgal, A. Verma, M. Singh, and S. Behal, "FloRa: Flow table low-rate overflow reconnaissance and detection in SDN," *IEEE Trans. Netw. Service Manag.*, vol. 21, no. 6, pp. 6670–6683, 2024, doi: https://doi.org/10.1109/TNSM.2024.3446178.
6. J. Zheng and D. Li, "GCN-TC: Combining trace graph with statistical features for network traffic classification," in *Proc. 2019 IEEE Int. Conf. Commun. (ICC)*, IEEE, 2019, pp. 1–6, doi: https://doi.org/10.1109/ICC.2019.8761337.
7. Z. Ji, Y. Cui, Y. Guo, et al., "Towards saturation attack detection in SDN: A multi-edge representation learning-based method," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 37, p. 138, 2025, doi: https://doi.org/10.1016/j.jksuci.2025.102345.
8. L. Ran et al., "Hint: Graph and flow-device hypergraph neural network-based method," in *Proc. Int. Conf. Intell. Comput. (ICIC)*, Springer Nature Singapore, 2025, pp. 50–62, doi: https://doi.org/10.1007/978-981-97-5615-5_5.
9. X. Deng, J. Zhu, X. Pei, et al., "Flow topology-based graph convolutional network for intrusion detection in label-limited IoT networks," *IEEE Trans. Netw. Service Manag.*, vol. 20, no. 1, pp. 684–696, 2022, doi: https://doi.org/10.1109/TNSM.2022.3212749.
10. D. Tang, D. Zhang, Z. Qin, et al., "SFTO-Guard: Real-time detection and mitigation system for slow-rate flow table overflow attacks," *J. Netw. Comput. Appl.*, vol. 213, p. 103597, 2023, doi: https://doi.org/10.1016/j.jnca.2023.103597.
11. A. A. Najar and S. M. Naik, "Cyber-secure SDN: A CNN-based approach for efficient detection and mitigation of DDoS attacks," *Comput. Secur.*, vol. 139, p. 103716, 2024, doi: https://doi.org/10.1016/j.cose.2024.103716.