Consistency-Verified Backdoor Defense for Federated Graph Learning via Cross-Layer Drift

Yanwen Jia[1,*] **Yinuo Zhang**[1,*] **Zitong Shi**[2,*‡] **Yuxin Wu**[3] **Zihan Tan**[1]
Xuankun Rong[1] **Liangtao Zheng**[4] **Wenke Huang**[5,†] **Mang Ye**[1]
[1]Wuhan University [2]National University of Singapore [3]Renmin University of China
[4]University of California, San Diego [5]Nanyang Technological University

Abstract

Federated graph learning (FGL) enables privacy-preserving collaborative training over distributed graph data, yet remains vulnerable to backdoor attacks from malicious clients. In FGL, data and structural heterogeneity make benign clients difficult to distinguish from malicious ones, posing a key challenge to reliable defense. We propose VeriDrift, a backdoor defense framework for FGL based on cross-layer drift and consistency verification. On the client side, VeriDrift derives a client-level risk score from node-level cross-layer drift anomalies during GNN propagation. On the server side, VeriDrift verifies the consistency between the reported risk and the submitted model update, and converts the resulting verified risk into aggregation weights to suppress suspicious updates. Without relying on auxiliary clean data, VeriDrift enables reliable risk assessment under heterogeneous FGL and improves robustness against adaptive attacks. Extensive experiments show that VeriDrift consistently reduces attack success rates across different FGL backdoor settings while maintaining clean accuracy. The code is publicly available at link.

1 Introduction

Federated Learning (FL) enables collaborative training across decentralized clients without sharing raw data [1–5]. With the increasing demand for learning over graph-structured data in real-world scenarios such as social networks, recommendation systems, and healthcare, Federated Graph Learning (FGL) extends this paradigm to graph neural networks (GNNs), allowing clients to jointly train graph models while keeping local graphs private [6–8]. Despite these benefits, FGL is vulnerable to backdoor attacks launched by malicious clients [9, 10]. By injecting trigger patterns into local graphs, attackers can make the global GNN behave normally on clean nodes while producing attacker-specified predictions once triggers appear.

Existing backdoor defenses for FGL can be roughly divided into two categories. The first category transfers conventional FL strategies to FGL, relying on update-space filtering, robust aggregation, trusted references, or server-side purification [11–13]. However, conventional FL typically treats backdoor triggers as fixed patterns injected into independent samples, whereas FGL triggers are embedded in graph structures and may vary in topology, scale, and injection location. Such flexibility leads to non-aligned update behaviors across clients, making it difficult for transferred FL defenses to capture consistent attack patterns. Based on these observations, we raise the following question: *I) How can we reliably identify attack-specific signals in federated graph learning in the presence of both data heterogeneity and graph structural heterogeneity?* [14–16]

The second category consists of FGL-specific defenses that consider graph-structured backdoor behaviors. Empirical defenses, such as FedTGE, identify and reweight suspicious clients through graph-specific distributional signals [17], while server-side purification methods, such as MultiKD,

[*]Equal contribution. [‡]Project Leader. [†]Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

mitigate backdoor effects using clean auxiliary data [18]. Although these methods exploit graph-specific information, they still rely on either empirical filtering or auxiliary data. Certified defenses, such as Yang et al. [19], provide inference-time robustness guarantees, but do not prevent malicious updates from contaminating training-time aggregation and are mainly designed for graph classification. This leads us to consider: *II) How can we build a defense mechanism against adaptive backdoor attacks in federated graph learning?*

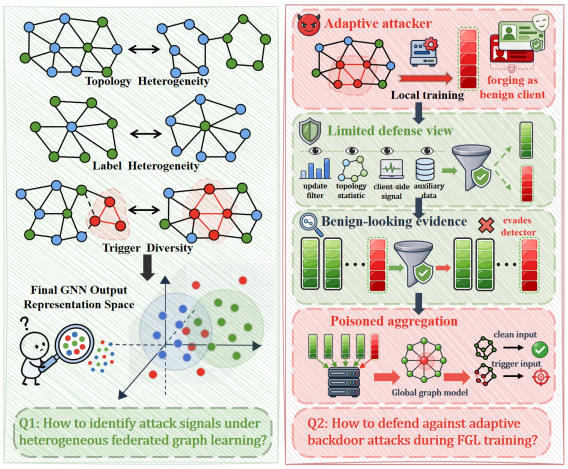

Figure 1: **Problem Illustration.** Two challenges in backdoor defense for heterogeneous FGL.

To address the aforementioned challenges, we propose VeriDrift, a backdoor defense framework for federated graph learning via cross-layer drift and consistency verification. Our design is based on the observation that graph backdoors not only affect final predictions, but also distort how node representations evolve across GNN layers during message passing. VeriDrift therefore uses cross-layer representation drift as client-side evidence for detecting poisoned supervision, and further checks its consistency with server-observable model-update geometry before aggregation. To address I), VeriDrift estimates client risk from node-level cross-layer drift. For each locally supervised node, it measures abnormal late-stage displacement and directional turning across consecutive GNN layers. Since backdoor triggers usually affect only a small subset of nodes, VeriDrift aggregates the most suspicious node-level risks to characterize localized propagation distortion.

To tackle II), VeriDrift introduces a server-side consistency verification mechanism. A malicious client may attempt to evade the defense by forging its reported risk, but implanting an effective backdoor still requires its submitted update to carry attack-relevant information into federated aggregation. The server therefore verifies whether the reported risk is compatible with update-space evidence, including directional deviation from an update center and abnormal update magnitude. A low reported risk with strong update-space abnormality indicates possible risk under-reporting, while a high reported risk supported by update-space evidence is strengthened into a verified risk for robust aggregation. Overall, VeriDrift identifies attack-specific propagation patterns, validates client risks before aggregation, and suppresses suspicious updates while preserving benign structural heterogeneity in FGL. Our main contributions are summarized as follows:

- We show that cross-layer propagation drift provides useful representation-side evidence for FGL backdoor defense, capturing localized representation distortions that remain informative under graph-structured heterogeneity and help separate malicious clients from benign ones.
- We propose VeriDrift, a consistency-verification defense that compares potentially forgeable client-side risk scores with server-observable update-space evidence, without requiring trusted reference data or auxiliary clean datasets.
- We conduct extensive experiments across multiple graph datasets, diverse attack settings, and different FGL tasks, showing that VeriDrift consistently reduces attack success rate while preserving clean-data utility.

2 Related work

Federated graph learning. Federated Graph Learning (FGL) extends federated learning to graph-structured data, enabling clients to collaboratively train GNNs without sharing local graphs. Existing studies have explored FGL from several perspectives, including system-level frameworks and benchmarks [20–22], topology-aware aggregation and cross-client structural collaboration [23–25, 16], personalized, prototype-based, or generalizable federated graph learning [26–28, 14, 15, 29], and sparsification-based efficient FGL [30]. These works show that graph topology and client heterogeneity are central to FGL [31–33], yet training-time backdoor defense in heterogeneous FGL remains underexplored compared with optimization and efficiency. Our work follows this emerging security

direction by suppressing backdoor attacks during federated graph training while preserving benign structural heterogeneity [34].

Backdoor attacks and defenses in FGL. Backdoor attacks implant hidden malicious behaviors while preserving normal performance on clean inputs [9, 35, 10]. In graph learning, such attacks are more flexible, as triggers can be constructed through subgraph injection, feature perturbation, or adaptive structural manipulation [36, 37]. Recent studies extend backdoor attacks to FGL, where malicious clients inject graph triggers during local training and contaminate the global model through aggregation [38, 19]. Existing defenses can be divided into conventional FL defenses and FGL-specific defenses. Conventional FL defenses include Byzantine-robust aggregation and distributed learning, Sybil or client filtering, trust bootstrapping, and model clipping or noise injection [39–41, 11, 42, 12, 13]. Although effective under certain FL assumptions, they are not designed for graph-structured triggers and may confuse malicious drift with benign graph heterogeneity. Recent FGL-specific defenses provide progress: Yang et al. [19] derive certified defenses for graph backdoors, FedTGE detects suspicious clients through topological graph energy [17, 43], and MultiKD purifies the global model through multi-teacher distillation [18]. However, existing defenses still struggle with adaptive attacks and training-time robust aggregation. Our work complements this line by leveraging cross-layer drift signals and developing a consistency-verification mechanism for robust aggregation.

Representation dynamics in graph neural networks. Node representations in GNNs evolve through message passing across layers, and such layer-wise dynamics have been studied to understand Laplacian smoothing, oversmoothing, and deep GNN training [44–46]. For FGL backdoor defense, representation dynamics are particularly relevant because graph triggers can affect not only final predictions, but also how node representations are displaced and redirected during propagation. While recent graph-aware defenses exploit energy distributions or intermediate representations [17, 18], these signals mainly provide static distributional or representation evidence and do not directly characterize the intrinsic propagation anomalies induced by graph triggers during GNN message passing. Our work fills this gap by treating cross-layer representation dynamics as primary evidence for identifying backdoor-induced propagation distortion in heterogeneous FGL.

3 Preliminaries

3.1 Federated graph learning with layer-wise propagation

Following standard federated graph learning, we consider a system with K clients $\mathcal{C} = \{1, \dots, K\}$ and a central server. Each client k owns a private graph $\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k, \mathbf{X}_k, \mathbf{Y}_k)$, where $\mathcal{V}_k^{\mathrm{tr}} \subseteq \mathcal{V}_k$ denotes the locally supervised nodes. At communication round t, the server broadcasts the global model $\mathbf{w}^{(t)}$, and each selected client performs local GNN training and returns an update $\Delta\mathbf{w}_k^{(t)} = \mathbf{w}_k^{(t)} - \mathbf{w}^{(t)}$. For an N-layer GNN, we denote the representation of node $v \in \mathcal{V}_k^{\mathrm{tr}}$ at layer l by $\mathbf{h}_{k,v}^{(l)}$, where $l = 1, \dots, N$. The sequence $\{\mathbf{h}_{k,v}^{(l)}\}_{l=1}^{N}$ forms the layer-wise propagation trajectory used by our drift analysis. Raw graphs, labels, and node-level representations remain local to clients.

3.2 Backdoor-induced propagation drift

Unlike fixed triggers on independent samples, graph triggers are embedded into local topology, node attributes, or both. In a GNN, node representations are updated by repeatedly aggregating information from neighboring nodes [47, 6, 7]. A generic message-passing layer can be written as

$$\mathbf{h}_v^{(l)} = \mathrm{UPDATE}^{(l)}\left(\mathbf{h}_v^{(l-1)}, \mathrm{AGG}^{(l)}\left(\left\{\mathbf{h}_u^{(l-1)} : u \in \mathcal{N}(v)\right\}\right)\right), \tag{1}$$

where $\mathcal{N}(v)$ denotes the neighborhood of node v. Since each layer mixes information from the neighborhood, a trigger that perturbs local structure or features can affect not only the triggered node itself, but also the way its representation evolves across layers.

Conceptually, after a trigger changes the neighborhood from $\mathcal{N}(v)$ to $\widetilde{\mathcal{N}}(v)$ and the input features from $\mathbf{X}$ to $\widetilde{\mathbf{X}}$, the layer-wise trajectory of node v may deviate from its clean propagation path:

$$\mathcal{T}_v = \left(\mathbf{h}_v^{(1)}, \mathbf{h}_v^{(2)}, \dots, \mathbf{h}_v^{(N)}\right), \quad \widetilde{\mathcal{T}}_v = \left(\widetilde{\mathbf{h}}_v^{(1)}, \widetilde{\mathbf{h}}_v^{(2)}, \dots, \widetilde{\mathbf{h}}_v^{(N)}\right). \tag{2}$$

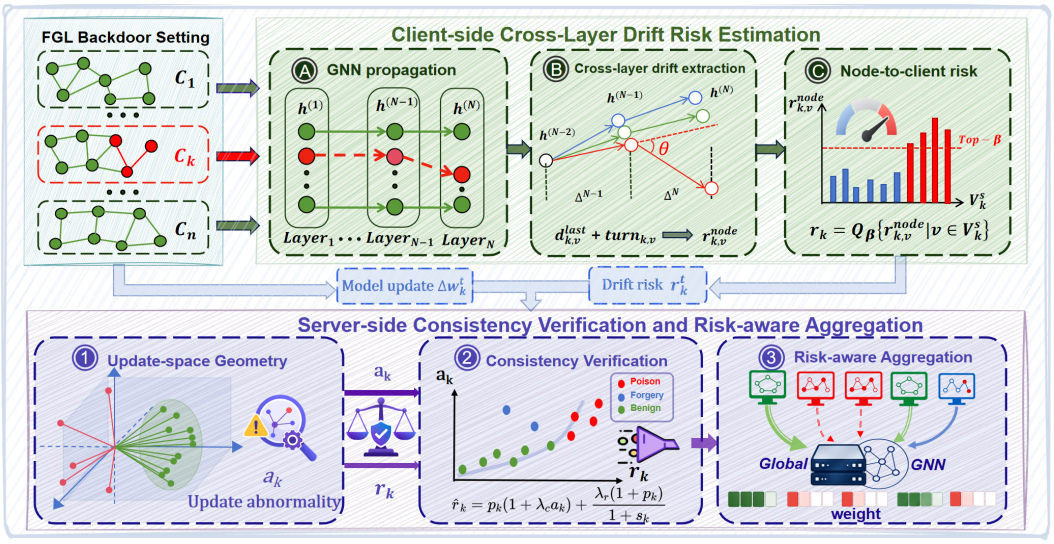

Figure 2: Overview of VeriDrift. Each client estimates a risk score from abnormal cross-layer drift of locally supervised nodes. The server verifies the risk using the geometric behavior of the corresponding client model update and converts the verified risk into aggregation weights.

We refer to such trigger-induced deviation in the layer-wise propagation process as *backdoor-induced propagation drift*. This notion is used to characterize how poisoned supervision may alter internal GNN dynamics before the final prediction is produced. In practice, our defense does not require observing paired clean and poisoned trajectories; instead, it estimates suspicious drift from abnormal patterns within the observed layer-wise trajectory.

3.3 Threat model

Malicious clients can poison their local graphs, modify local training, submit adversarial model updates, and manipulate the reported client-side risk score. Their goal is to implant a targeted backdoor into the global GNN such that clean nodes are classified normally, while trigger-attached test nodes are predicted as an attacker-specified target label. The server cannot access raw graphs, node features, labels, and does not rely on auxiliary clean data. It only observes the submitted model updates and client-level risk scores. The defender aims to suppress backdoor contributions during aggregation while preserving benign graph heterogeneity.

4 Method

4.1 Overview

Figure 2 illustrates the overall framework of VeriDrift. The method consists of three stages. First, each client estimates a risk score from abnormal cross-layer drift of locally supervised nodes. Second, the server verifies this risk using the geometric behavior of the corresponding client model update, thereby assessing whether the client-side risk is consistent with server-observable update evidence. Third, the verified risk is converted into continuous aggregation weights to suppress suspicious updates while preserving benign heterogeneity.

4.2 Client-side risk computation from cross-layer drift

Motivation. Graph backdoors not only alter final predictions but can also reshape how malicious supervision propagates through GNN layers. Existing graph-aware defenses often rely on static hidden representations or graph-level distributional statistics, but they do not explicitly capture how trigger-induced effects evolve during message passing. We therefore construct a client-side risk signal from cross-layer representation dynamics, aiming to characterize whether locally supervised nodes exhibit abnormal displacement or irregular directional changes in deeper propagation stages.

Cross-layer drift indicators. For client k, let $\mathcal{V}_k^{\mathrm{tr}}$ denote the locally supervised nodes and let $\mathbf{h}_{k,v}^{(l)}$ be the representation of node v at layer l of an N-layer GNN. We focus on the last two transitions:

$$\Delta_{k,v}^{(N-1)} = \mathbf{h}_{k,v}^{(N-1)} - \mathbf{h}_{k,v}^{(N-2)}, \qquad \Delta_{k,v}^{(N)} = \mathbf{h}_{k,v}^{(N)} - \mathbf{h}_{k,v}^{(N-1)}. \tag{3}$$

Here, $\Delta_{k,v}^{(N-1)}$ records the transition ending at the penultimate layer, while $\Delta_{k,v}^{(N)}$ records the final transition before prediction. Comparing them allows us to examine whether the representation trajectory remains smoothly refined or becomes abruptly distorted at the final propagation stage. Based on these transitions, we define two complementary drift indicators:

$$d_{k,v}^{\mathrm{last}} = \left\| \Delta_{k,v}^{(N)} \right\|_2, \qquad \mathrm{turn}_{k,v} = 1 - \frac{\left\langle \Delta_{k,v}^{(N-1)}, \Delta_{k,v}^{(N)} \right\rangle}{\left\| \Delta_{k,v}^{(N-1)} \right\|_2 \left\| \Delta_{k,v}^{(N)} \right\|_2 + \epsilon}. \tag{4}$$

The displacement $d_{k,v}^{\mathrm{last}}$ measures how strongly node v is moved in the final representation transition, while $\mathrm{turn}_{k,v}$ measures the directional change between the two consecutive transitions. Together, they capture both the magnitude and the geometric irregularity of late-stage propagation drift.

Node-level risk scoring. We combine the two indicators after robust positive-tail normalization:

$$r_{k,v}^{\mathrm{node}} = \frac{1}{2}\left[\phi\big(d_{k,v}^{\mathrm{last}}\big) + \phi(\mathrm{turn}_{k,v})\right]. \tag{5}$$

Here, $\phi(\cdot)$ denotes a median/MAD-based normalization that suppresses ordinary local fluctuations and retains anomalously large positive deviations. Thus, $r_{k,v}^{\mathrm{node}}$ assigns higher risk to supervised nodes whose late-stage propagation exhibits unusually strong displacement, irregular directional turning, or both, relative to the local node population.

Client-level risk consolidation. Backdoor effects usually influence only a small subset of supervised nodes. If all node risks are averaged directly, the attack signal may be diluted by many normal nodes. We therefore summarize each client using an upper-quantile statistic of its node-level risks:

$$q_k = Q_\beta\big(\{r_{k,v}^{\mathrm{node}} \mid v \in \mathcal{V}_k^{\mathrm{tr}}\}\big), \qquad r_k = q_k. \tag{6}$$

Here, β is the risk quantile, set to 0.90 by default. The quantile score q_k captures the localized high-risk level among the most affected supervised nodes, rather than the average behavior of the entire local node set. This makes the client-level risk more suitable for graph backdoor settings, where poisoned supervision is typically sparse but can induce strong drift on a small group of nodes.

Submission to the server. After local training, each client submits its model update Δw_k together with the client-level risk score r_k. The score r_k summarizes representation-side evidence from cross-layer propagation dynamics. The server then verifies whether this client-side risk is consistent with the corresponding model-update geometry before using the verified risk for aggregation.

4.3 Server-side consistency verification

Motivation. The client-side risk score reflects representation-side evidence from cross-layer drift, but it is still reported by clients and may be manipulated by malicious participants. However, an effective backdoor must ultimately influence the global model through the submitted model update. This creates a cross-view constraint: the reported risk should be compatible with the update-space behavior observed by the server. We therefore introduce a server-side consistency verification module to adjust the submitted client-side risk before aggregation.

Cross-view consistency principle. The verification module compares two views of each client: the submitted drift-based risk and the server-observable update-space abnormality. A high submitted risk supported by abnormal update geometry is strengthened, while a low submitted risk coupled with strong update abnormality indicates possible risk under-reporting. Conversely, a high submitted risk with benign update geometry may reflect heterogeneity-induced drift and should not be over-amplified without update-space support.

Update-space abnormality. At communication round t, let

$$u_k^{(t)} = \mathrm{vec}\Big(w_k^{(t)} - w^{(t)}\Big), \qquad \bar{u}_k^{(t)} = \frac{u_k^{(t)}}{\|u_k^{(t)}\|_2 + \epsilon} \tag{7}$$

denote the submitted update and its normalized direction. The server maintains a risk memory

$$M_j^{(t-1)} = \zeta M_j^{(t-2)} + (1-\zeta)\hat{r}_j^{(t-1)}, \qquad \eta_{j|-k}^{(t)} = \exp\Big(-M_j^{(t-1)}\Big), \tag{8}$$

where $\zeta \in [0,1)$ controls temporal smoothing and $M_j^{(0)} = 0$. For target client k, the server constructs a leave-one-out low-risk update center:

$$\mathbf{c}_{-k}^{(t)} = \frac{\sum_{j\neq k} \eta_{j|-k}^{(t)} \bar{u}_j^{(t)}}{\sum_{j\neq k} \eta_{j|-k}^{(t)}}. \tag{9}$$

The leave-one-out design prevents the target update from shaping its own reference direction, while the memory weights assign larger influence to clients with lower historical risk. We first compute the update-norm score of client k as

$$\ell_k^{(t)} = \left\| u_k^{(t)} \right\|_2, \qquad z_{k,\mathrm{norm}}^{(t)} = \frac{\ell_k^{(t)} - \mathrm{median}_j\{\ell_j^{(t)}\}}{\mathrm{MAD}_j\{\ell_j^{(t)}\} + \epsilon}. \tag{10}$$

The update-space abnormality of client k is then defined as

$$a_k^{(t)} = \left(1 - \cos\left(\bar{u}_k^{(t)}, \mathbf{c}_{-k}^{(t)}\right)\right) + \left[z_{k,\mathrm{norm}}^{(t)}\right]_+, \tag{11}$$

where $[x]_+ = \max(x, 0)$. The first term measures directional deviation from the leave-one-out low-risk update center $\mathbf{c}_{-k}^{(t)}$, while the second term captures unusually large update magnitude. Since the center is determined by historical risk memory rather than the current submitted risk, reporting a low risk in the current round cannot directly reduce $a_k^{(t)}$.

Consistency verification. The server robustly standardizes the submitted drift risks r_k as $z_{k,\mathrm{risk}}^{(t)}$, and converts them into nonnegative risk pressures $p_k^{(t)} = \mathrm{softplus}(z_{k,\mathrm{risk}}^{(t)})$. To verify whether this risk is consistent with update-space behavior, we fit a leave-one-out isotonic predictor $g_{-k}^{(t)}(\cdot)$ from all clients except k, mapping risk pressure to expected update abnormality. The residual and consistency score are defined as

$$\rho_k^{(t)} = \left| a_k^{(t)} - g_{-k}^{(t)}\left(p_k^{(t)}\right) \right|, \quad \tilde{\rho}_k^{(t)} = \frac{\rho_k^{(t)} - \mathrm{median}_j\{\rho_j^{(t)}\}}{\mathrm{MAD}_j\{\rho_j^{(t)}\} + \epsilon}, \quad s_k^{(t)} = \exp\Big(-[\tilde{\rho}_k^{(t)}]_+\Big). \tag{12}$$

A large $s_k^{(t)}$ indicates that the submitted risk is compatible with update-space evidence, while a small $s_k^{(t)}$ reveals an abnormal cross-view mismatch.

Definition 4.3. (verified risk score). For client k at round t, we define its verified risk as

$$\hat{r}_k^{(t)} = p_k^{(t)}\big(1 + \lambda_c s_k^{(t)}\big) + \lambda_c \frac{1 - s_k^{(t)}}{1 + p_k^{(t)}}. \tag{13}$$

The first term strengthens the submitted risk when it is supported by update-space evidence. The second term penalizes cross-view inconsistency, and becomes relatively stronger when the submitted risk is small. Thus, a client reporting low risk but exhibiting abnormal update geometry still receives an increased verified risk, while unsupported high risk is not further amplified. The verified risk $\hat{r}_k^{(t)}$ is then used for risk-aware aggregation.

4.4 Risk-aware aggregation with verified risk

Aggregation rule. Given the verified risk score $\hat{r}_k^{(t)}$, the server performs continuous risk-aware aggregation instead of hard client rejection. Importantly, aggregation is based on the verified risk rather than the raw submitted risk, so a client cannot obtain a large weight merely by reporting a small risk value if its update geometry is inconsistent with that report. At round t, we define the risk penalty coefficient and the unnormalized aggregation weight as

$$\gamma_t = \alpha\big(1 - \exp(-t/\tau)\big), \qquad \tilde{\omega}_k^{(t)} = n_k \exp\Big(-\gamma_t \hat{r}_k^{(t)}\Big), \tag{14}$$

where α controls the maximum penalty strength, τ controls the warm-up speed, and n_k is the number of supervised nodes on client k. The factor n_k preserves the standard sample-size weighting in FedAvg, while the exponential term down-weights clients with larger verified risks. When verified risks are small or similar, the rule naturally reduces to sample-size-weighted averaging.

The normalized aggregation coefficient and global update are

$$\omega_k^{(t)} = \frac{\tilde{\omega}_k^{(t)}}{\sum_j \tilde{\omega}_j^{(t)}}, \qquad w^{(t+1)} = w^{(t)} + \sum_{k=1}^{K} \omega_k^{(t)} \Delta w_k^{(t)}. \tag{15}$$

Since $\exp(-\gamma_t \hat{r}_k^{(t)})$ decreases monotonically with the verified risk, suspicious clients contribute less to the global model. The warm-up schedule keeps early aggregation close to FedAvg while the risk memory and update reference are still unstable, and gradually strengthens the risk penalty in later rounds. Compared with hard filtering, this soft weighting is more suitable for non-IID FGL because it suppresses suspicious updates without discarding benign clients whose updates may deviate due to structural heterogeneity.

Discussion and limitations. VeriDrift combines cross-layer drift and update-space geometry, using the former as representation-side evidence and the latter for server-side verification. This avoids relying on a single view and better suits heterogeneous FGL, where benign structural differences may affect local risks or model updates. However, this verification is conditional: it detects score-update inconsistency when forged low risks are not supported by update-space behavior, as discussed in Appendix D. Extremely high malicious-client ratios or stronger adaptive attackers that jointly suppress drift and align update geometry may weaken this signal, which we leave for future study.

5 Experiments

We conducted node-classification experiments under both IID and Non-IID-Louvain [21, 48] settings on six benchmark graphs, and evaluated task-level generality on three graph-classification datasets.

5.1 Experimental setup

Datasets. To evaluate node-level backdoor defense in FGL, we conduct experiments on six benchmark graphs across three scenarios: Citation Network [49], Co-authorship [50], and Amz-purchase [51]. Specifically, the datasets include Cora, PubMed, CiteSeer, Coauthor-CS, Coauthor-Phy, and Amz-Photo. Dataset details are provided in Appendix A.

Table 1: Comparison with representative backdoor defense baselines over six datasets under two settings. Topo. indicates whether the defense explicitly exploits graph-structured or propagation-aware information beyond generic update statistics; Ref. indicates whether the method is reference-free, i.e., does not require proxy data, a validation set, or trusted clean server data; Veri. indicates whether client risk is checked against update-space evidence before aggregation.

Methods	Topo.	Ref.	Veri.	Cora			PubMed			CiteSeer			Coauthor-CS			Coauthor-Phy			Amz-Photo		
				$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
IID with a malicious proportion of $\Upsilon = 0.3$ and a trigger type of Renyi																					
Vanilla [AISTATS17]	×	✓	×	82.89	1.37	42.13	81.27	1.67	41.47	77.34	0.00	38.67	90.69	1.11	45.90	95.54	0.24	47.89	93.80	1.06	47.43
FLTrust [NDSS21]	×	×	×	83.76	24.48	54.12	82.03	18.57	50.30	76.39	37.38	56.89	87.94	70.85	79.40	94.59	44.03	69.31	93.20	51.34	72.27
RLR [AAAI21]	×	✓	×	81.38	18.58	49.98	80.40	38.41	59.41	75.59	33.09	54.34	90.13	2.51	46.32	95.20	0.79	48.00	93.79	4.21	49.00
RFA [TSP22]	×	✓	×	84.91	4.41	44.66	80.51	8.49	44.50	76.98	6.29	41.64	90.41	11.28	50.85	95.43	16.86	56.15	92.74	41.30	67.02
DnC [NDSS21]	×	✓	×	82.59	38.59	60.59	82.27	30.71	56.49	74.97	43.22	59.10	89.15	15.33	52.24	94.41	3.03	48.72	93.18	25.47	59.33
FLAME [USENIX22]	×	✓	×	80.80	66.49	73.65	80.67	60.13	70.40	74.63	96.39	85.51	87.78	88.68	88.23	93.61	58.25	75.93	91.10	95.84	93.47
FedCPA [ICCV23]	×	✓	×	84.22	18.32	51.27	81.91	16.83	49.37	76.29	34.51	55.40	87.58	61.33	74.46	94.45	42.37	68.41	92.87	59.23	76.05
FedTGE [ICLR25]	✓	✓	×	84.56	64.48	74.52	81.56	60.24	70.90	76.14	66.99	71.57	89.31	77.09	83.20	94.73	60.50	77.62	93.08	90.82	91.95
VeriDrift [Ours]	✓	✓	✓	81.21	89.95	**85.58**	85.50	65.56	**75.53**	75.75	96.48	**86.12**	87.92	96.26	**92.09**	94.72	88.96	**91.84**	92.67	97.22	**94.95**

Methods	Topo.	Ref.	Veri.	Cora			PubMed			CiteSeer			Coauthor-CS			Coauthor-Phy			Amz-Photo		
				$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
Non-IID-Louvain with a malicious proportion of $\Upsilon = 0.3$ and a trigger type of Renyi																					
Vanilla [AISTATS17]	×	✓	×	76.90	3.61	40.26	86.04	5.90	45.97	74.25	0.51	37.38	83.84	5.47	44.66	92.14	2.63	47.39	88.82	27.29	58.06
FLTrust [NDSS21]	×	×	×	82.71	16.22	49.47	81.02	30.97	55.99	75.29	28.10	51.70	82.43	45.07	63.75	90.94	86.12	88.53	88.36	42.38	65.37
RLR [AAAI21]	×	✓	×	72.40	55.90	64.15	75.62	66.78	71.20	72.53	71.80	72.17	83.38	34.43	58.91	88.76	44.58	66.67	81.83	81.12	81.48
RFA [TSP22]	×	✓	×	83.11	3.33	43.22	79.06	5.44	42.25	76.75	7.09	41.92	84.07	19.86	51.97	92.51	40.37	66.44	88.19	38.25	63.22
DnC [NDSS21]	×	✓	×	78.48	44.32	61.40	82.37	30.63	56.50	73.34	49.64	61.49	79.90	29.06	54.48	91.97	19.84	55.91	84.36	72.64	78.50
FLAME [USENIX22]	×	✓	×	68.07	58.65	63.36	80.10	86.35	83.23	74.34	79.15	76.75	80.13	68.18	74.16	90.05	78.40	84.23	85.87	79.34	82.61
FedCPA [ICCV23]	×	✓	×	82.34	14.20	48.27	80.63	17.03	48.83	77.00	24.47	50.74	82.73	44.26	63.50	91.48	60.00	75.74	86.07	49.92	67.99
FedTGE [ICLR25]	✓	✓	×	77.79	54.13	65.96	82.67	68.66	75.67	75.47	72.71	74.09	81.40	76.31	78.86	92.91	68.78	80.85	86.36	66.92	76.64
VeriDrift [Ours]	✓	✓	✓	80.42	84.89	**82.66**	80.80	93.30	**87.05**	76.11	96.08	**86.10**	82.00	76.31	**79.16**	89.83	89.02	**89.43**	83.77	86.20	**84.99**

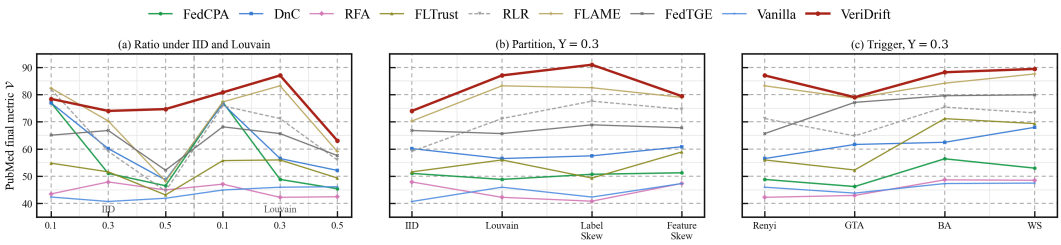

Figure 3: Sensitivity analysis on PubMed under different settings.

Comparison methods. For the main node-classification setting, we compare our method with representative traditional FL and FGL defenses: (1) FedAvg [52]; (2) DnC [53]; (3) FLAME [13]; (4) FLTrust [12]; (5) RFA [11]; (6) RLR [54]; and (7) FedCPA [55] from traditional FL, as well as (8) FedTGE [17] from FGL. For task-level comparison with graph-specific defenses, we additionally evaluate CertGNN [19] and MultiKD [18] in the federated graph-classification setting in Table 2. Detailed descriptions of these methods are provided in Appendix C.

Network structure. Following the common practice in graph-based backdoor evaluation and FGL [6, 17], we adopt a 3-layer GCN as the feature extractor and classifier, with the hidden layer size set to 128 for all datasets.

Backdoor attack. We demonstrate the effectiveness of the proposed method under the popular graph backdoor attack paradigm [17, 37, 56, 57]. Considering the stealth of the backdoor attacks, we set the trigger size to 4 nodes for all experiments, and its type and location are Renyi and random, respectively. We set the malicious client ratio Υ as $\{0.1, 0.3, 0.5\}$, and conduct experiments under both IID and Non-IID-Louvain settings.

Additional attack and task settings. We further evaluate adaptive risk-forging attacks, where malicious clients perform standard backdoor training but report either zero or median drift risk before communication. To our knowledge, existing FGL backdoor attacks do not instantiate a fully adaptive objective that jointly suppresses drift, update abnormality, and backdoor loss; we discuss this stronger case as a limitation in Appendix D. To test task-level generality, we also evaluate federated graph classification on MUTAG, PROTEINS, and NCI1 under adaptive subgraph triggers based on Opt-GDBA, comparing with NoDefense, FLAME, CertGNN, MultiKD, and VeriDrift.

Implementation details. We summarize implementation details as follows:

- **Dataset Split**: For node classification, following [56, 57], we partition labeled nodes into training, validation, and testing sets with a 60%/20%/20% split. Unlabeled nodes are used for trigger injection and relabeled with the target class.
- **Training Setting**: We repeat each experiment five times for each method to ensure the robustness and reliability of the results. The Adam optimizer [58] with a learning rate of 0.01 is used to train the GNN models.
- **Evaluation Metrics**: Following [17, 59, 52], we report clean accuracy ($\mathcal{A}$), backdoor failure rate ($\mathcal{R}$), and their average score ($\mathcal{V}$), as defined in Eq. (16).

$$\mathcal{A} = \frac{1}{K}\sum_{k=1}^{K}\frac{\sum_{\xi\in D_k^{\text{test}}}\mathbf{1}[\arg\max z_\xi = y_\xi]}{|D_k^{\text{test}}|}, \quad \mathcal{R} = \frac{1}{K_m}\sum_{k\in\mathcal{C}_m}\left(1-\frac{\sum_{\tilde{\xi}\in\tilde{D}_k^{\text{test}}}\mathbf{1}\left[\arg\max \tilde{z}_{\tilde{\xi}} = \tilde{y}_{\tilde{\xi}}\right]}{\left|\tilde{D}_k^{\text{test}}\right|}\right), \quad \mathcal{V} = \frac{1}{2}(\mathcal{A}+\mathcal{R}). \tag{16}$$

We define a query instance ξ with logits output $z = \mathcal{M}^t(\xi)$, where $\mathcal{M}^t$ denotes the globally shared model. The mean accuracy across K clients is denoted by $\mathcal{A}$. Similarly, backdoor queries $\tilde{\xi}$ and test samples $\tilde{D}_k^{\text{test}}$ yield a mean backdoor failure rate $\mathcal{R}$ across $\tilde{K}$ malicious clients. The final evaluation metric $\mathcal{V}$ is defined as the average of $\mathcal{A}$ and $\mathcal{R}$.

5.2 Experimental results

Table 2: Robustness evaluation of VeriDrift. Left: robustness under adaptive risk-forging attacks, where malicious clients submit attack-effective updates but forge low risk claims. Right: defense performance on federated graph classification under adaptive subgraph triggers based on Opt-GDBA.

(a) Adaptive risk-forging attacks.

Method	Veri.	Forging	CiteSeer			PubMed			Amz-Photo		
			$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
NoDefense	–	–	74.25	0.51	37.38	86.04	5.90	45.97	88.82	27.29	58.06
VeriDrift	×	zero	77.85	6.09	41.97	84.96	4.77	44.87	87.96	13.93	50.95
VeriDrift	✓	zero	**73.65**	**71.33**	**72.49**	**80.51**	**64.46**	**72.49**	**78.81**	**76.69**	**77.75**
VeriDrift	×	median	74.63	4.24	39.44	84.90	11.43	48.17	81.96	19.57	50.77
VeriDrift	✓	median	**71.04**	**46.86**	**58.95**	**78.84**	**72.70**	**75.77**	**74.27**	**52.46**	**63.37**

(b) Federated graph classification.

Method	MUTAG			PROTEINS			NCI1		
	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
NoDefense	71.14	29.36	50.25	75.35	46.28	60.82	70.16	31.28	50.72
FLAME	71.90	87.37	79.64	70.53	82.94	76.74	71.21	65.87	68.54
CertGNN	69.15	56.21	62.68	71.41	89.26	80.34	70.77	59.61	65.19
MultiKD	70.26	67.96	69.11	68.37	66.96	67.67	73.51	58.32	65.92
VeriDrift	**76.38**	**96.22**	**86.30**	**70.45**	**94.37**	**82.41**	**72.04**	**98.73**	**85.39**

Main performance. Table 1 compares VeriDrift with FL and FGL defenses. Under IID, VeriDrift attains the highest $\mathcal{V}$ on all six datasets, with average $\mathcal{A} = 86.30$, $\mathcal{R} = 89.07$, and $\mathcal{V} = 87.69$. Under Non-IID-Louvain, it again ranks first in $\mathcal{V}$ on all six datasets, with average $\mathcal{A} = 82.16$, $\mathcal{R} = 87.63$, and $\mathcal{V} = 84.90$. VeriDrift raises $\mathcal{R}$ while keeping $\mathcal{A}$ competitive, showing that cross-layer drift suppresses attack propagation without sacrificing clean utility. Figure 3 confirms this utility-robustness balance on PubMed under varying malicious ratios, partitions, and trigger types.

Additional robustness. Table 2a evaluates adaptive risk-forging attacks. Without verification, forged zero and median risks retain $\mathcal{A}$ but drop average $\mathcal{R}$ to 8.26 and 11.75, reducing $\mathcal{V}$ to 45.93 and 46.13. With consistency verification, VeriDrift raises $\mathcal{R}$ to 70.83 and 57.34 and $\mathcal{V}$ to 74.24 and 66.03, with usable $\mathcal{A}$. This shows that update-space evidence detects forged low risks and restores the $\mathcal{A}$–$\mathcal{R}$ balance. Table 2b shows that VeriDrift generalizes to graph classification, achieving the best $\mathcal{V}$ on MUTAG, PROTEINS, and NCI1 through high $\mathcal{R}$ while maintaining competitive $\mathcal{A}$.

5.3 Diagnostic analysis

Table 3: Diagnostic ablation studies of VeriDrift.

(a) Core module ablation.

Drift	Veri.	CiteSeer			PubMed			Amz-Photo		
		$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
×	×	74.25	0.51	37.38	86.04	5.90	45.97	88.82	27.29	58.06
✓	×	71.97	26.98	49.48	82.03	47.52	64.78	82.79	46.01	64.40
×	✓	72.51	73.24	72.88	77.32	60.85	69.09	84.76	75.83	80.30
✓	✓	**76.11**	**96.08**	**86.10**	**80.80**	**93.30**	**87.05**	**83.77**	**86.20**	**84.99**

(b) Impact of backbone models.

Backbone	CiteSeer			PubMed			Amz-Photo		
	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
GAT	68.94	69.60	69.27	71.33	41.67	56.50	76.32	92.88	84.60
GIN	70.74	73.42	72.08	67.33	95.56	81.45	65.33	83.59	74.46
GraphSAGE	77.91	48.87	63.39	84.60	93.33	88.97	76.45	87.54	82.00
GCN	**76.11**	**96.08**	**86.10**	**80.80**	**93.30**	**87.05**	**83.77**	**86.20**	**84.99**

Core module ablation. We ablate the two key components of VeriDrift: cross-layer drift risk estimation and consistency verification. As shown in Table 3, consistency verification is a strong server-side component, whereas drift-only risk is insufficient under strong non-IID attacks. This is expected because attack-effective backdoors must eventually enter aggregation through model updates. Still, adding cross-layer drift to verification consistently improves V over verification alone on all three datasets, indicating that drift provides complementary representation-side evidence rather than replacing update-space verification. Appendix G further visualizes such drift patterns across datasets and triggers, and Appendix B.4 reports additional hyper-parameter sensitivity results.

Impact of backbone models. We further replace the default GCN with GAT, GIN, and GraphSAGE to examine whether VeriDrift depends on a specific GNN architecture. Table 3 shows that VeriDrift maintains defense performance across different backbones, suggesting that backdoor-induced cross-layer drift is a general propagation phenomenon rather than an artifact of a particular GNN model.

6 Conclusion

In this paper, we studied backdoor defense in federated graph learning under graph-structured heterogeneity. We proposed VeriDrift, a backdoor defense framework based on cross-layer drift and consistency verification that characterizes backdoor-induced propagation distortion. By combining drift-based risk estimation, consistency verification with model-update evidence, and risk-aware aggregation, VeriDrift suppresses suspicious contributions while preserving benign structural heterogeneity. Extensive experiments under diverse attacks, non-IID settings, and malicious-client ratios demonstrate that VeriDrift effectively reduces attack success rate while maintaining clean accuracy. We hope this work offers a useful perspective for future research on representation-aware and evidence-guided backdoor defense in federated graph learning.

Acknowledgments and Disclosure of Funding

This work was supported by the National Natural Science Foundation of China under Grants 62361166629 and 623B2080, the Independent Research Project of the National Key Laboratory of Big Data and Decision, the National Research Foundation (NRF), Singapore, under Grant ID NRF-PA2025-NTU, and Nanyang Technological University (NTU). The supercomputing system at the Supercomputing Center of Wuhan University supported the numerical calculations in this paper.

References

[1] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2):12:1–12:19, 2019.

[2] Priyanka Mary Mammen. Federated learning: Opportunities and challenges. *arXiv preprint arXiv:2101.05428*, 2021.

[3] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1–2):1–210, 2021.

[4] Zitong Shi, Guancheng Wan, Wenke Huang, Guibin Zhang, Jiawei Shao, Mang Ye, and Carl Yang. Privacy-enhancing paradigms within federated multi-agent systems. *arXiv preprint arXiv:2503.08175*, 2025.

[5] Wenke Huang, Yangxu Liao, Wenjia Lei, Guancheng Wan, Xuankun Rong, Chi Wen, He Li, Mang Ye, Qingqing Wu, and Bo Du. Federated clinical concept and disease semantic learning for congenital heart disease diagnosis. *npj Digital Medicine*, 2026.

[6] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

[7] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[8] Xingbo Fu, Binchi Zhang, Yushun Dong, Chen Chen, and Jundong Li. Federated graph machine learning: A survey of concepts, techniques, and applications. *ACM SIGKDD Explorations Newsletter*, 24(2):32–47, 2022.

[9] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[10] Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 35(1):5–22, 2024.

[11] Krishna Pillutla, Sham M. Kakade, and Zaid Harchaoui. Robust aggregation for federated learning. *IEEE Transactions on Signal Processing*, 70:1142–1154, 2022.

[12] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. FLTrust: Byzantine-robust federated learning via trust bootstrapping. In *Proceedings of the Network and Distributed System Security Symposium*, 2021.

[13] Thien Duc Nguyen, Phillip Rieger, Huili Chen, Hossein Yalame, Helen Möllering, Hossein Fereidooni, Samuel Marchal, Markus Miettinen, Azalia Mirhoseini, Shaza Zeitouni, Farinaz Koushanfar, Ahmad-Reza Sadeghi, and Thomas Schneider. FLAME: Taming backdoors in federated learning. In *31st USENIX Security Symposium*, pages 1415–1432, 2022.

[14] Zihan Tan, Guancheng Wan, Wenke Huang, and Mang Ye. FedSSP: Federated graph learning with spectral knowledge and personalized preference. *Advances in Neural Information Processing Systems*, 37:34561–34581, 2024.

[15] Zihan Tan, Guancheng Wan, Wenke Huang, He Li, Guibin Zhang, Carl Yang, and Mang Ye. FedSPA: Generalizable federated graph learning under homophily heterogeneity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15464–15475, 2025.

[16] Zihan Tan, Suyuan Huang, Guancheng Wan, Wenke Huang, He Li, and Mang Ye. S2FGL: Spatial spectral federated graph learning. *arXiv preprint arXiv:2507.02409*, 2025.

[17] Guancheng Wan, Zitong Shi, Wenke Huang, Guibin Zhang, Dacheng Tao, and Mang Ye. Energy-based backdoor defense against federated graph learning. In *International Conference on Learning Representations*, 2025.

[18] Jiale Zhang, Yanan Wang, Bosen Rao, Chengcheng Zhu, Xiaobing Sun, and Yu Li. MultiKD: Backdoor defense in federated graph learning via attention-guided multi-teacher distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 28239–28246, 2026.

[19] Yuxin Yang, Qiang Li, Jinyuan Jia, Yuan Hong, and Binghui Wang. Distributed backdoor attacks on federated graph learning and certified defenses. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security*, pages 2829–2843, 2024.

[20] Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Carl Yang, Han Xie, Lichao Sun, Lifang He, Liangwei Yang, Philip S. Yu, Yu Rong, Peilin Zhao, Junzhou Huang, Murali Annavaram, and Salman Avestimehr. FedGraphNN: A federated learning system and benchmark for graph neural networks. *arXiv preprint arXiv:2104.07145*, 2021.

[21] Zhen Wang, Weirui Kuang, Yuexiang Xie, Liuyi Yao, Yaliang Li, Bolin Ding, and Jingren Zhou. FederatedScope-GNN: Towards a unified, comprehensive and efficient package for federated graph learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4110–4120, 2022.

[22] Xunkai Li, Yinlin Zhu, Boyang Pang, Guochen Yan, Yeyu Yan, Zening Li, Zhengyu Wu, Wentao Zhang, Rong-Hua Li, and Guoren Wang. OpenFGL: A comprehensive benchmark for federated graph learning. *Proceedings of the VLDB Endowment*, 18(5):1305–1320, 2025.

[23] Xunkai Li, Zhengyu Wu, Wentao Zhang, Yinlin Zhu, Rong-Hua Li, and Guoren Wang. FedGTA: Topology-aware averaging for federated graph learning. *Proceedings of the VLDB Endowment*, 17(1):41–50, 2023.

[24] Guancheng Wan, Wenke Huang, and Mang Ye. Federated graph learning under domain shift with generalizable prototypes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 15429–15437, 2024.

[25] Guancheng Wan, Fengyuan Ran, Ruikang Zhang, Wenke Huang, Xuankun Rong, Guibin Zhang, Yuxin Wu, Bo Du, and Mang Ye. MOTION: Multi-sculpt evolutionary coarsening for federated continual graph learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=qmbG6u7DK0`.

[26] Jinheon Baek, Wonyong Jeong, Jiongdao Jin, Jaehong Yoon, and Sung Ju Hwang. Personalized subgraph federated learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 1396–1415, 2023.

[27] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. FedProto: Federated prototype learning across heterogeneous clients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8432–8440, 2022.

[28] Wenke Huang, Guancheng Wan, Mang Ye, and Bo Du. Federated graph semantic and structural learning. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 3830–3838, 2023.

[29] Jiaru Qian, Guancheng Wan, Wenke Huang, Guibin Zhang, Yuxin Wu, Bo Du, and Mang Ye. GHOST: Generalizable one-shot federated graph learning with proxy-based topology knowledge retention. In *Forty-second International Conference on Machine Learning*, 2025. URL `https://openreview.net/forum?id=nAk0ENu8LS`.

[30] Zitong Shi, Guancheng Wan, Wenke Huang, Guibin Zhang, He Li, Carl Yang, and Mang Ye. EAGLES: Towards effective, efficient, and economical federated graph learning via unified sparsification. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 55046–55064, 2025.

[31] Guancheng Wan, Xiaoran Shang, Yuxin Wu, Guibin Zhang, Jinhe Bi, Liangtao Zheng, Xin Lin, Yue Liu, Yanbiao Ma, Wenke Huang, and Bo Du. HYPERION: Fine-grained hypersphere alignment for robust federated graph learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=TZB6YT8Owr`.

[32] Zihan Tan, Guancheng Wan, Wenke Huang, Bin Yang, and Mang Ye. FedPRE: Robust federated graph learning against topological corruption. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pages 1390–1400, 2026.

[33] Jiaqi Liu, Zihan Tan, Guancheng Wan, Wenke Huang, He Li, and Mang Ye. FedSDR: Federated graph learning with structural noise detection and reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

[34] Zitong Shi, Guancheng Wan, Wenke Huang, Yuxin Wu, Quan Zhang, and Mang Ye. Towards robust text-attributed federated graph learning: multimodal threats and defense. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 25383–25391, 2026.

[35] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2938–2948, 2020.

[36] Zhaohan Xi, Ren Pang, Shouling Ji, and Ting Wang. Graph backdoor. In *30th USENIX Security Symposium*, pages 1523–1540, 2021.

[37] Enyan Dai, Minhua Lin, Xiang Zhang, and Suhang Wang. Unnoticeable backdoor attacks on graph neural networks. In *Proceedings of the ACM Web Conference 2023*, pages 2263–2273, 2023.

[38] Fan Liu, Siqi Lai, Yansong Ning, and Hao Liu. Bkd-FedGNN: A benchmark for classification backdoor attacks on federated graph neural network. *arXiv preprint arXiv:2306.10351*, 2023.

[39] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[40] El Mahdi El Mhamdi, Rachid Guerraoui, and Sébastien Rouault. The hidden vulnerability of distributed learning in byzantium. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3521–3530, 2018.

[41] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5650–5659, 2018.

[42] Clement Fung, Chris J. M. Yoon, and Ivan Beschastnikh. The limitations of federated learning in sybil settings. In *Proceedings of the 23rd International Symposium on Research in Attacks, Intrusions and Defenses*, pages 301–316, 2020.

[43] Guancheng Wan, Yuxin Wu, Zitong Shi, Wenke Huang, Bo Du, and Mang Ye. Dual-view energy-topological backdoor defense for federated graph learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

[44] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[45] Lingxiao Zhao and Leman Akoglu. PairNorm: Tackling oversmoothing in GNNs. In *International Conference on Learning Representations*, 2020.

[46] Guohao Li, Matthias Müller, Ali Thabet, and Bernard Ghanem. DeepGCNs: Can GCNs go as deep as CNNs? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9267–9276, 2019.

[47] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272, 2017.

[48] Ke Zhang, Carl Yang, Xiaoxiao Li, Lichao Sun, and Siu-Ming Yiu. Subgraph federated learning with missing neighbor generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 6671–6682, 2021.

[49] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 40–48, 2016.

[50] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

[51] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52, 2015.

[52] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282, 2017.

[53] Virat Shejwalkar and Amir Houmansadr. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. In *Proceedings of the Network and Distributed System Security Symposium*, 2021.

[54] Mustafa Safa Ozdayi, Murat Kantarcioglu, and Yulia R. Gel. Defending against backdoors in federated learning with robust learning rate. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9268–9276, 2021.

[55] Sungwon Han, Sungwon Park, Fangzhao Wu, Sundong Kim, Bin Zhu, Xing Xie, and Meeyoung Cha. Towards attack-tolerant federated learning via critical parameter analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4999–5008, 2023.

[56] Jing Xu, Minhui Xue, and Stjepan Picek. Explainability-based backdoor attacks against graph neural networks. In *Proceedings of the 3rd ACM Workshop on Wireless Security and Machine Learning*, pages 31–36, 2021.

[57] Haibin Zheng, Haiyang Xiong, Jinyin Chen, Haonan Ma, and Guohan Huang. Motif-backdoor: Rethinking the backdoor attack on graph neural networks via motifs. *IEEE Transactions on Computational Social Systems*, 11(2):2479–2493, 2024.

[58] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[59] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems*, volume 2, pages 429–450, 2020.

[60] Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. TUDataset: A collection of benchmark datasets for learning with graphs. In *ICML 2020 Workshop on Graph Representation Learning and Beyond*, 2020.

[61] Asim Kumar Debnath, Rosa L. Lopez de Compadre, Gargi Debnath, Alan J. Shusterman, and Corwin Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of Medicinal Chemistry*, 34(2):786–797, 1991.

[62] Karsten M. Borgwardt, Cheng Soon Ong, Stefan Schönauer, S. V. N. Vishwanathan, Alexander J. Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21(suppl_1):i47–i56, 2005.

[63] Nikil Wale, Ian A. Watson, and George Karypis. Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowledge and Information Systems*, 14(3):347–375, 2008.

[64] Zaixi Zhang, Jinyuan Jia, Binghui Wang, and Neil Zhenqiang Gong. Backdoor attacks to graph neural networks. In *Proceedings of the 26th ACM Symposium on Access Control Models and Technologies*, pages 15–26, 2021.

[65] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, 1998.

[66] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[67] Murray Rosenblatt. Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832–837, 1956.

[68] Emanuel Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

A Dataset details

The statistics of the datasets used in our experiments are provided in Table 4. Our experiments involve two types of graph learning tasks: node classification on a single large graph partitioned across clients, and graph classification on collections of graph instances.

- **Node classification datasets.** For node-level federated graph learning, we use six benchmark graphs from three application scenarios.
  - **Citation Network** (Cora, CiteSeer, PubMed): Citation network datasets such as Cora, CiteSeer, and PubMed are composed of research papers connected by citation links, where nodes denote papers and edges represent citation relationships. These datasets are widely used for paper classification and knowledge graph construction, and they also provide useful evidence for analyzing research trends and emerging topics in academic domains [49].
  - **Co-authorship** (CS, Physics): Co-authorship datasets, including Computer Science (CS) and Physics, are built from the Microsoft Academic Graph. In these graphs, nodes correspond to authors, while edges indicate co-author relationships. They are commonly used to predict authors' research fields and support the analysis of collaboration patterns, research-area distributions, and academic influence [50].
  - **Amazon co-purchase** (Photo): Amazon co-purchase datasets such as Photo are constructed from product co-purchase relations. In these datasets, nodes represent products and edges denote co-purchase links. The main task is to predict product categories, which is relevant to applications such as recommendation systems and market analysis [51].
- **Graph classification datasets.** For task-level generality evaluation, we further use three graph classification datasets from TUDataset: MUTAG, PROTEINS, and NCI1 [60]. In this setting, each data sample is an individual graph instance, and the goal is to predict the graph-level label rather than node labels.
  - **MUTAG**: MUTAG is a molecular graph dataset consisting of nitroaromatic compounds, where nodes represent atoms and edges represent chemical bonds. The binary classification task is to predict whether a compound has mutagenic effects [61].
  - **PROTEINS**: PROTEINS is a bioinformatics graph dataset in which each graph represents a protein structure. Nodes correspond to amino acids, and edges represent spatial or structural relationships between amino acids. The task is to classify proteins into enzyme and non-enzyme categories [62].
  - **NCI1**: NCI1 is a molecular graph dataset derived from chemical compound screening, where each graph represents a chemical compound with atoms as nodes and bonds as edges. The binary graph classification task is to distinguish compounds according to their activity in anticancer screening assays [63].

Table 4: **Statistics** of datasets used in experiments.

Node Dataset	#Nodes	#Edges	#Classes	#Features	Graph Dataset	#Graphs	Avg. #Nodes	Avg. #Edges	#Classes
Cora	2,708	5,278	7	1,433	MUTAG	188	17.93	19.79	2
CiteSeer	3,327	4,732	6	3,703	PROTEINS	1,113	39.06	72.82	2
PubMed	19,717	44,324	3	500	NCI1	4,110	29.87	32.30	2
Coauthor-CS	18,333	327,576	15	6,805	–	–	–	–	–
Coauthor-Phy	34,493	495,924	5	8,415	–	–	–	–	–
Amz-Photo	7,650	287,326	8	745	–	–	–	–	–

B Additional experimental details

B.1 Compute resources

All experiments were conducted on a Linux workstation. The machine was equipped with NVIDIA RTX 3090 GPU with 24GB memory and 15 vCPUs of Intel(R) Xeon(R) Platinum 8358P CPU @ 2.60GHz. Each experimental run used one GPU, and no distributed or multi-GPU training was used. The reported results were obtained from independent runs under the same hardware configuration.

B.2 Experimental protocol

Following common FGL experimental protocols, the node-classification experiments use $K = 10$ clients, a client sampling rate of 1.0, and 100 communication rounds. The main results use Renyi triggers, while GTA, WS, and BA triggers are evaluated in Appendix B.3. As shown in Figure 4, VeriDrift remains stable across a reasonable range of β, with both clean accuracy and backdoor failure rate staying competitive around $\beta \in [0.5, 0.9]$. Increasing λ_c generally improves backdoor suppression, while an overly large value may reduce clean accuracy. Thus, we use moderate default values to balance clean utility and defense effectiveness.

B.3 Additional results under different attacks, partitions, and malicious ratios

In this section, we conduct additional experiments to further validate VeriDrift under diverse attack settings, including different Non-IID partitions (i.e., Non-IID-Louvain, Non-IID-Label-Skew, and Non-IID-Feature-Skew), varying malicious ratios, and multiple trigger types. We briefly introduce the main characteristics of these triggers as follows.

- **Renyi Trigger** [64]: Inspired by the Erdős–Rényi random graph model, the Renyi trigger inserts random nodes and edges into the graph, where each edge is independently generated with a certain probability. Such stochastic structures lack clear patterns, which improves their obfuscation capability and makes them difficult to detect.
- **GTA Trigger** [36]: The GTA trigger injects carefully designed subgraphs, such as star- or ring-shaped structures, into the original graph. It also modifies the features of inserted nodes to strengthen their influence on target predictions. Although this design can achieve high attack success rates, its regular structure may make it more detectable by defense methods.
- **WS Trigger** [65]: Based on the Watts–Strogatz small-world model, the WS trigger creates subgraphs with high clustering coefficients and short average path lengths. These properties resemble real-world networks and thus enhance stealthiness. Meanwhile, the strong local clustering can affect graph-level structural properties and improve attack effectiveness, though its performance depends on parameters such as the rewiring probability.
- **BA Trigger** [66]: Derived from the Barabási–Albert scale-free model, the BA trigger constructs subgraphs with a power-law degree distribution, where a small number of hub nodes have high connectivity. These hub nodes can help propagate the backdoor effect more effectively. While this structure is close to real-world networks, inserting highly connected nodes may require more graph modifications and potentially reduce efficiency.

Table 5: Additional comparison results under different malicious-client ratios with Renyi triggers.

Methods	Topo.	Ref.	Veri.	Cora			PubMed			CiteSeer			Coauthor-CS			Coauthor-Phy			Amz-Photo		
				$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
IID with a malicious proportion of $\Upsilon = 0.5$ and a trigger type of Renyi																					
Vanilla [AISTATS17]	×	✓	×	82.16	2.52	42.34	81.82	2.00	41.91	78.15	0.41	39.28	90.98	0.37	45.68	95.49	0.01	47.75	94.00	0.52	47.26
FLTrust [NDSS21]	×	×	×	81.94	27.11	54.53	79.75	6.50	43.13	75.62	41.84	58.73	88.67	16.33	52.50	92.01	22.61	57.31	92.91	26.99	59.95
RLR [AAAI21]	×	✓	×	80.24	11.72	45.98	77.48	11.57	44.53	75.29	12.81	44.05	90.20	10.88	50.54	95.09	11.78	53.44	93.67	10.93	52.30
RFA [TSP22]	×	✓	×	84.61	12.41	48.51	76.30	13.64	44.97	77.87	12.85	45.36	91.11	13.93	52.52	90.08	14.83	52.46	93.39	9.37	51.38
DnC [NDSS21]	×	✓	×	82.52	11.23	46.88	81.24	16.00	48.62	77.36	15.79	46.58	90.82	11.20	51.01	94.32	13.41	53.87	93.78	11.17	52.48
FLAME [USENIX22]	×	✓	×	83.10	10.15	46.63	80.65	15.74	48.20	78.08	17.48	47.78	90.76	11.67	51.22	95.51	12.22	53.87	93.42	14.60	54.01
FedCPA [ICCV23]	×	✓	×	84.52	12.72	48.62	80.99	12.00	46.50	77.80	10.20	44.00	89.69	12.58	51.14	95.05	11.14	53.10	94.02	15.91	54.97
FedTGE [ICLR25]	✓	✓	×	82.10	55.32	68.71	81.20	53.22	67.21	76.62	51.44	64.03	88.60	67.99	78.30	91.43	52.75	72.09	92.86	50.64	71.75
VeriDrift [Ours]	✓	✓	✓	88.08	55.43	**71.76**	83.33	66.00	**74.67**	80.77	65.80	**73.29**	88.16	78.22	**83.19**	94.29	85.33	**89.81**	93.18	83.32	**88.25**

Methods	Topo.	Ref.	Veri.	Cora			PubMed			CiteSeer			Coauthor-CS			Coauthor-Phy			Amz-Photo		
				$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
Non-IID-Louvain with a malicious proportion of $\Upsilon = 0.5$ and a trigger type of Renyi																					
Vanilla [AISTATS17]	×	✓	×	78.12	3.63	40.88	85.03	7.25	46.14	76.60	1.17	38.88	83.82	4.10	43.96	89.36	4.32	46.84	89.07	32.79	60.93
FLTrust [NDSS21]	×	×	×	80.69	9.31	45.00	83.75	14.96	49.35	73.95	25.59	49.77	82.22	24.27	53.25	90.14	30.28	60.21	89.58	41.65	65.61
RLR [AAAI21]	×	✓	×	65.78	18.44	42.11	71.53	40.39	55.96	61.18	0.47	30.83	83.20	27.95	55.58	92.27	28.47	60.37	77.29	65.10	71.20
RFA [TSP22]	×	✓	×	80.93	2.86	41.90	80.42	4.50	42.46	77.16	11.81	44.48	83.74	14.29	49.02	92.31	31.61	61.96	88.37	38.67	63.52
DnC [NDSS21]	×	✓	×	77.51	13.35	45.43	85.06	19.21	52.14	74.28	17.51	45.90	82.16	6.71	44.44	86.79	42.34	64.56	85.03	53.11	69.07
FLAME [USENIX22]	×	✓	×	73.99	23.64	48.81	82.34	35.80	59.07	74.41	19.51	46.96	83.73	12.45	48.09	89.85	67.52	78.69	81.70	60.05	70.88
FedCPA [ICCV23]	×	✓	×	80.84	5.08	42.96	84.16	6.64	45.40	76.91	14.93	45.92	83.61	20.23	51.92	91.23	38.75	64.99	89.97	35.76	62.87
FedTGE [ICLR25]	✓	✓	×	73.16	27.90	50.53	83.48	31.71	57.60	73.67	41.48	57.58	81.96	40.85	61.40	90.54	64.53	77.53	86.34	59.45	72.89
VeriDrift [Ours]	✓	✓	✓	64.01	88.29	**76.15**	75.17	51.00	**63.09**	80.35	61.62	**70.98**	80.82	80.97	**80.90**	90.99	73.48	**82.24**	86.44	92.35	**89.39**

Methods	Topo.	Ref.	Veri.	Cora			PubMed			CiteSeer			Coauthor-CS			Coauthor-Phy			Amz-Photo		
				$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
IID with a malicious proportion of $\Upsilon = 0.1$ and a trigger type of Renyi																					
Vanilla [AISTATS17]	×	✓	×	83.95	2.48	43.22	81.78	2.94	42.36	77.31	0.91	39.11	89.30	7.26	48.28	93.76	9.08	51.42	93.32	5.84	49.58
FLTrust [NDSS21]	×	×	×	84.93	32.57	58.75	82.62	27.00	54.81	76.71	38.64	57.68	88.49	83.32	85.91	94.52	61.07	77.80	93.24	73.40	83.32
RLR [AAAI21]	×	✓	×	84.25	30.39	57.32	79.60	82.33	80.97	76.34	86.38	81.36	89.67	19.23	54.45	94.69	45.73	70.21	93.34	31.91	62.63
RFA [TSP22]	×	✓	×	85.74	11.58	48.66	78.99	8.00	43.50	78.92	12.25	45.59	90.21	36.03	63.12	94.04	54.62	74.33	93.60	79.91	86.76
DnC [NDSS21]	×	✓	×	82.98	85.84	84.41	81.27	72.67	76.97	77.55	90.40	83.98	87.87	85.57	86.72	91.97	82.38	87.18	93.18	95.82	94.50
FLAME [USENIX22]	×	✓	×	83.62	84.73	84.18	81.05	81.73	81.39	76.18	93.61	84.90	88.07	87.49	87.78	93.16	86.66	89.91	92.70	93.75	93.23
FedCPA [ICCV23]	×	✓	×	83.56	78.86	81.21	78.05	76.00	77.03	78.48	91.59	**85.04**	88.40	85.57	86.99	92.89	86.71	89.80	93.16	95.13	94.15
FedTGE [ICLR25]	✓	✓	×	84.56	63.57	74.07	80.83	49.33	65.08	78.18	82.89	80.54	88.66	87.36	88.01	93.35	87.93	90.64	93.49	93.84	93.67
VeriDrift [Ours]	✓	✓	✓	84.95	88.89	**86.92**	83.86	80.98	**82.42**	75.93	92.59	84.26	87.29	91.15	**89.22**	94.72	91.20	**92.96**	92.42	97.87	**95.15**

Methods	Topo.	Ref.	Veri.	Cora			PubMed			CiteSeer			Coauthor-CS			Coauthor-Phy			Amz-Photo		
				$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
Non-IID-Louvain with a malicious proportion of $\Upsilon = 0.1$ and a trigger type of Renyi																					
Vanilla [AISTATS17]	×	✓	×	79.14	0.67	39.91	87.01	2.86	44.94	74.30	1.56	37.93	82.35	11.36	46.85	87.76	32.41	60.09	84.85	21.82	53.34
FLTrust [NDSS21]	×	×	×	82.48	20.20	51.34	84.08	27.43	55.75	75.51	22.10	48.80	81.28	64.46	72.87	88.91	58.76	73.83	89.19	63.25	76.22
RLR [AAAI21]	×	✓	×	80.18	80.75	80.47	85.79	65.95	75.87	76.71	92.86	84.78	83.60	25.96	54.78	90.13	82.08	86.10	87.11	93.59	90.35
RFA [TSP22]	×	✓	×	85.65	6.32	45.99	81.32	12.86	47.09	77.57	7.98	42.77	84.57	24.66	54.62	89.46	48.05	68.75	88.50	52.51	70.50
DnC [NDSS21]	×	✓	×	81.35	81.66	81.50	85.15	68.00	76.58	76.92	94.77	85.84	78.70	81.80	80.25	85.18	86.13	85.66	87.96	96.25	92.11
FLAME [USENIX22]	×	✓	×	78.63	85.14	81.88	82.68	71.95	77.31	76.15	95.21	85.68	80.60	81.82	81.21	88.59	83.34	85.97	85.80	97.17	91.48
FedCPA [ICCV23]	×	✓	×	81.87	74.43	78.15	79.98	74.48	77.23	76.20	96.36	**86.28**	82.24	80.34	81.29	88.04	85.51	86.78	87.52	98.52	93.02
FedTGE [ICLR25]	✓	✓	×	77.16	74.45	75.80	81.66	74.67	78.16	75.86	80.23	78.05	81.08	83.12	82.10	90.15	85.72	87.94	87.10	93.94	90.52
VeriDrift [Ours]	✓	✓	✓	71.64	94.33	**82.98**	90.20	71.43	**80.81**	72.88	98.44	85.66	71.26	97.01	**84.13**	89.94	97.39	**93.66**	87.89	99.66	**93.78**

Table 6: Additional comparison results under different heterogeneity settings with Renyi triggers.

Methods	Topo.	Ref.	Veri.	Cora			PubMed			CiteSeer			Coauthor-CS			Coauthor-Phy			Amz-Photo		
				$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
Non-IID-Label-Skew with $\alpha = 0.5$ and a malicious proportion of $\Upsilon = 0.3$ under a trigger type of Renyi																					
Vanilla [AISTATS17]	×	✓	×	80.53	1.36	40.95	82.48	2.22	42.35	78.31	0.95	39.63	88.47	1.02	44.75	89.28	7.70	48.49	91.50	0.16	45.83
FLTrust [NDSS21]	×	×	×	81.66	22.57	52.12	86.25	12.41	49.33	76.39	42.12	59.26	84.97	45.48	65.23	86.74	38.75	62.75	88.60	30.90	59.75
RLR [AAAI21]	×	✓	×	79.74	66.44	73.09	78.50	76.65	77.58	70.78	71.00	70.89	87.47	14.23	50.85	87.96	57.82	72.89	88.77	70.19	79.48
RFA [TSP22]	×	✓	×	83.86	2.98	43.42	79.44	2.22	40.83	77.44	6.75	42.10	88.47	6.12	47.30	88.91	35.64	62.28	89.53	22.71	56.12
DnC [NDSS21]	×	✓	×	77.08	49.92	63.50	88.03	26.96	57.50	74.32	44.20	59.26	87.23	31.93	59.58	87.92	39.63	63.78	88.82	40.64	64.73
FLAME [USENIX22]	×	✓	×	79.74	70.31	75.03	85.68	79.35	82.52	73.09	66.66	69.88	84.75	84.98	84.87	83.60	68.76	76.18	86.75	94.09	90.42
FedCPA [ICCV23]	×	✓	×	82.75	19.97	51.36	87.49	13.94	50.72	75.26	48.64	61.95	85.91	46.85	66.38	87.63	45.11	66.37	89.74	47.30	68.52
FedTGE [ICLR25]	✓	✓	×	76.95	73.97	75.46	82.70	75.08	78.89	69.97	72.41	71.19	86.89	59.67	73.28	88.34	66.14	77.24	89.91	45.45	67.68
VeriDrift [Ours]	✓	✓	✓	73.77	98.41	**86.09**	91.38	90.57	**90.98**	70.58	72.81	**71.70**	84.33	93.39	**88.86**	79.84	97.34	**88.59**	87.85	96.95	**92.40**

Methods	Topo.	Ref.	Veri.	Cora			PubMed			CiteSeer			Coauthor-CS			Coauthor-Phy			Amz-Photo		
				$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
Non-IID-Feature-Skew with $\alpha = 0.5$ and a malicious proportion of $\Upsilon = 0.3$ under a trigger type of Renyi																					
Vanilla [AISTATS17]	×	✓	×	84.27	0.91	42.59	85.37	9.29	47.33	76.89	0.96	38.93	89.48	1.27	45.38	93.68	12.43	53.05	93.74	16.68	55.21
FLTrust [NDSS21]	×	×	×	83.88	26.11	54.99	85.25	32.62	58.94	76.32	38.99	57.66	86.82	59.19	73.00	92.57	47.72	70.14	93.03	47.36	70.19
RLR [AAAI21]	×	✓	×	81.14	27.01	54.08	79.67	69.59	74.63	72.73	88.42	80.57	88.63	10.93	49.78	92.14	34.28	63.21	93.05	28.61	60.83
RFA [TSP22]	×	✓	×	85.08	4.16	44.62	81.45	13.29	47.37	77.22	4.95	41.09	89.70	6.20	47.95	93.36	32.85	63.11	92.45	32.29	62.37
DnC [NDSS21]	×	✓	×	81.20	42.48	61.84	82.95	38.74	60.84	74.52	46.48	60.50	87.49	32.98	60.23	91.89	38.62	65.25	92.52	36.97	64.75
FLAME [USENIX22]	×	✓	×	82.63	70.49	76.56	85.97	67.99	76.98	76.14	98.03	**87.09**	85.43	82.35	83.89	92.35	83.14	87.75	93.02	85.90	89.46
FedCPA [ICCV23]	×	✓	×	83.21	23.42	53.31	83.38	19.15	51.27	75.49	39.40	57.44	87.65	48.62	68.13	92.48	56.13	74.30	92.67	67.00	79.84
FedTGE [ICLR25]	✓	✓	×	82.45	72.49	77.47	82.64	72.97	77.81	74.43	76.46	75.44	87.59	53.96	70.78	92.32	64.36	78.34	92.83	69.24	81.03
VeriDrift [Ours]	✓	✓	✓	79.96	84.06	**82.01**	79.94	78.90	**79.42**	74.62	97.64	86.13	86.12	88.39	**87.25**	93.10	90.29	**91.70**	93.12	98.93	**96.03**

Table 7: Additional comparison results under different trigger types in the Non-IID-Louvain setting.

Methods	Topo.	Ref.	Veri.	Cora			PubMed			CiteSeer			Coauthor-CS			Coauthor-Phy			Amz-Photo		
				$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
Non-IID-Louvain with a malicious proportion of $\Upsilon = 0.3$ under a trigger type of GTA																					
Vanilla [AISTATS17]	×	✓	×	78.97	2.53	40.75	85.32	2.29	43.80	75.73	1.14	38.44	84.03	7.56	45.80	93.65	13.26	53.46	86.62	35.12	60.87
FLTrust [NDSS21]	×	×	×	81.24	35.22	58.23	84.73	19.84	52.29	74.10	28.12	51.11	81.24	64.17	72.70	93.48	62.68	78.08	88.04	74.43	81.23
RLR [AAAI21]	×	✓	×	69.15	25.82	47.48	76.90	52.73	64.81	68.84	18.39	43.61	83.21	48.81	66.01	90.50	64.07	77.28	82.87	89.64	86.25
RFA [TSP22]	×	✓	×	84.29	8.57	46.43	82.34	3.56	42.95	77.72	10.22	43.97	85.44	39.68	62.56	94.51	51.96	73.24	87.59	54.70	71.14
DnC [NDSS21]	×	✓	×	78.57	42.09	60.33	80.18	43.16	61.67	72.94	34.12	53.53	81.45	54.56	68.00	85.71	65.39	75.55	80.08	74.43	77.25
FLAME [USENIX22]	×	✓	×	79.40	83.22	81.31	84.45	62.89	73.67	73.83	68.48	71.16	80.68	88.81	84.75	92.64	81.36	87.00	85.51	85.16	85.33
FedCPA [ICCV23]	×	✓	×	82.94	21.16	52.05	81.75	10.67	46.21	76.54	26.15	51.34	83.26	59.78	71.52	93.87	59.15	76.51	87.69	69.39	78.54
FedTGE [ICLR25]	✓	✓	×	80.60	41.98	61.29	81.82	72.46	77.14	76.08	69.95	73.02	81.18	91.43	86.31	93.12	72.71	82.92	87.05	87.20	87.12
VeriDrift [Ours]	✓	✓	✓	78.18	95.79	**86.98**	88.08	70.00	**79.04**	66.19	96.48	**81.34**	81.60	95.56	**88.58**	90.09	92.30	**91.20**	86.21	88.95	**87.58**

Methods	Topo.	Ref.	Veri.	Cora			PubMed			CiteSeer			Coauthor-CS			Coauthor-Phy			Amz-Photo		
				$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
Non-IID-Louvain with a malicious proportion of $\Upsilon = 0.3$ under a trigger type of BA																					
Vanilla [AISTATS17]	×	✓	×	78.41	5.64	42.03	87.21	7.40	47.31	74.52	1.93	38.23	83.30	6.54	44.92	88.24	9.77	49.01	84.49	38.70	61.60
FLTrust [NDSS21]	×	×	×	79.87	53.98	66.93	79.96	62.38	71.17	71.70	66.07	68.89	81.61	89.23	85.42	92.65	85.68	89.17	88.13	82.30	85.22
RLR [AAAI21]	×	✓	×	74.40	86.09	80.25	75.82	74.98	75.40	72.97	85.65	79.31	82.36	72.69	77.53	90.86	85.46	88.16	81.82	89.42	85.62
RFA [TSP22]	×	✓	×	79.01	19.73	49.37	80.57	16.76	48.67	74.01	33.23	53.62	83.15	54.46	68.81	93.47	63.42	78.45	86.10	69.71	77.91
DnC [NDSS21]	×	✓	×	76.91	55.98	66.45	84.63	40.30	62.47	72.65	60.17	66.41	78.54	61.05	69.80	91.84	70.53	81.19	81.22	79.71	80.47
FLAME [USENIX22]	×	✓	×	77.80	74.19	76.00	81.16	77.22	79.19	72.81	89.16	80.99	80.57	81.81	81.19	91.97	84.65	88.31	84.96	89.77	87.37
FedCPA [ICCV23]	×	✓	×	80.01	29.63	54.82	79.72	33.09	56.41	75.28	50.16	62.72	81.31	84.21	82.76	92.81	78.36	85.59	87.02	86.77	86.90
FedTGE [ICLR25]	✓	✓	×	79.03	61.31	70.17	79.16	80.01	79.59	72.88	57.29	65.09	81.42	89.92	85.67	92.36	86.05	89.21	85.46	90.58	88.02
VeriDrift [Ours]	✓	✓	✓	72.57	94.17	**83.37**	83.08	93.33	**88.21**	72.32	94.33	**83.33**	80.34	95.21	**87.78**	91.28	97.19	**94.24**	80.76	95.38	**88.07**

Methods	Topo.	Ref.	Veri.	Cora			PubMed			CiteSeer			Coauthor-CS			Coauthor-Phy			Amz-Photo		
				$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$	$\mathcal{A}$	$\mathcal{R}$	$\mathcal{V}$
Non-IID-Louvain with a malicious proportion of $\Upsilon = 0.3$ under a trigger type of WS																					
Vanilla [AISTATS17]	×	✓	×	78.60	9.05	43.83	86.22	8.73	47.48	73.31	6.55	39.93	83.09	11.86	47.48	93.26	16.08	54.67	83.66	36.72	60.19
FLTrust [NDSS21]	×	×	×	79.21	50.90	65.06	82.12	56.58	69.35	74.45	65.62	70.03	81.22	88.86	85.04	92.78	84.54	88.66	88.42	86.10	87.26
RLR [AAAI21]	×	✓	×	73.11	70.01	71.56	73.72	72.86	73.29	71.26	87.07	79.17	83.48	80.54	82.01	90.75	90.82	90.78	83.05	92.54	87.80
RFA [TSP22]	×	✓	×	79.16	17.79	48.47	81.18	15.87	48.52	73.68	29.69	51.69	83.66	52.45	68.05	93.54	62.71	78.12	85.84	61.73	73.78
DnC [NDSS21]	×	✓	×	77.83	54.57	66.20	84.75	51.19	67.97	74.41	66.42	70.41	79.31	58.44	68.88	91.72	71.36	81.54	81.38	68.58	74.98
FLAME [USENIX22]	×	✓	×	78.49	63.58	71.03	81.29	87.89	84.59	71.83	78.09	74.96	81.18	85.16	83.17	90.16	91.85	91.00	87.29	89.97	88.63
FedCPA [ICCV23]	×	✓	×	79.97	33.84	56.91	80.53	25.43	52.98	73.33	50.48	61.91	82.74	78.30	80.52	92.91	81.63	87.27	85.76	89.23	87.50
FedTGE [ICLR25]	✓	✓	×	76.14	68.72	72.43	81.90	77.94	79.92	74.56	86.30	80.43	81.77	89.66	85.72	92.43	87.16	89.80	86.19	88.11	87.15
VeriDrift [Ours]	✓	✓	✓	76.64	79.29	**77.97**	84.42	94.44	**89.43**	81.39	97.72	**89.56**	82.70	95.40	**89.05**	87.82	97.55	**92.69**	85.18	98.77	**91.97**

B.4 Hyper-parameter sensitivity

We further study the sensitivity of VeriDrift to the high-risk quantile β and the verification strength λ_c. As shown in Figure 4, VeriDrift remains stable within a reasonable range of both hyper-parameters, indicating that the method does not rely on fragile parameter tuning.

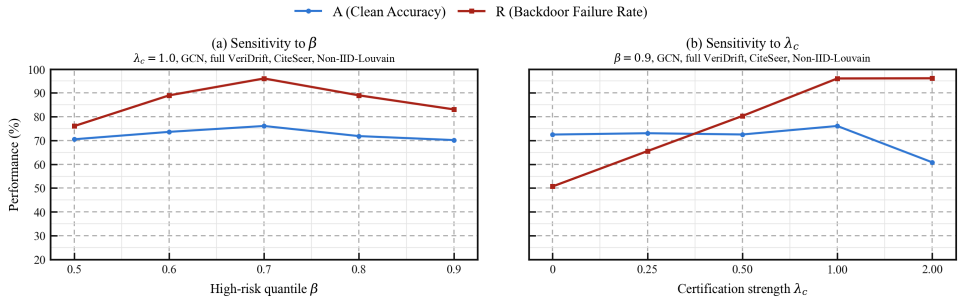

Figure 4: Hyper-parameter sensitivity of VeriDrift with respect to the high-risk quantile β and the verification strength λ_c.

C Details of the comparison methods

We compare VeriDrift with representative baselines from standard federated learning, robust federated aggregation, backdoor defense, and federated graph learning. The compared methods include FedAvg, FLTrust, RLR, RFA, DnC, FLAME, FedCPA, and FedTGE. We briefly introduce them as follows.

- **FedAvg** [52]: The standard federated averaging algorithm, where updates from all clients are averaged at the server without applying any defense mechanisms. This method is widely used in federated learning but is vulnerable to adversarial attacks.
- **FLTrust** [12]: FLTrust introduces a small clean root dataset on the server side to bootstrap trust. In each round, the server computes a trusted server update from the root dataset, assigns trust scores to client updates according to their directional similarity to the server update, normalizes their magnitudes, and then performs trust-weighted aggregation.
- **RLR** [54]: Robust Learning Rate (RLR) is a lightweight defense against federated backdoor attacks. Instead of assigning reliability scores to clients, it adjusts the server learning rate in a dimension-wise and round-wise manner according to the sign information of client updates. When

the sign agreement on a parameter dimension is insufficient, the corresponding learning rate is reversed to suppress adversarial directions.

- **RFA** [11]: Robust Federated Aggregation (RFA) replaces the standard weighted arithmetic mean with an approximate geometric median. By using a robust aggregation oracle based on the geometric median, RFA reduces the influence of corrupted or outlier client updates during federated training.
- **DnC** [53]: Divide-and-Conquer (DnC) is a Byzantine-robust aggregation method designed to defend against model poisoning attacks. It uses spectral analysis to identify suspicious directions in the client update space and filters out updates that deviate strongly along these directions before aggregation.
- **FLAME** [13]: FLAME is a backdoor defense framework for federated learning that combines anomalous model update detection, model weight clipping, and calibrated noise injection. By clustering client models to remove potentially poisoned updates and clipping model weights before adding noise, FLAME aims to eliminate backdoors while preserving benign performance.
- **FedCPA** [55]: FedCPA improves attack tolerance through critical parameter analysis. It is based on the observation that benign local models tend to share similar sets of top-k and bottom-k critical parameters, whereas poisoned models show different critical-parameter patterns. FedCPA computes a normality score for each client update based on this similarity and performs weighted aggregation to reduce the influence of likely malicious updates.
- **FedTGE** [17]: FedTGE is a federated graph backdoor defense method based on Topological Graph Energy. At the client level, it injects distribution knowledge into the local model by assigning low energy to benign samples and high energy to constructed malicious substitutes, and then selects benign clients through clustering. At the server level, it constructs a global energy graph from uploaded energy elements and uses energy propagation to further adjust aggregation weights.
- **CertGNN [19]:** CertGNN provides certified defenses for distributed graph backdoor attacks, mainly targeting graph classification and inference-time robustness. We include it in the federated graph-classification comparison where its task setting is directly applicable.
- **MultiKD [18]:** MultiKD is an FGL backdoor defense based on attention-guided multi-teacher distillation. Since it is designed around purification/distillation in graph classification, we compare it in the federated graph-classification setting rather than the main node-classification table.

D Consistency verification under risk forging

This appendix provides a formal analysis of the consistency-verification module under risk-forging attacks. The key question is whether a malicious client can obtain a small verified risk by reporting a low drift-based risk while still submitting a backdoor-effective update.

At round t, let $p_k^{(t)}$ denote the reported risk pressure and $a_k^{(t)}$ denote the update-space abnormality. The server fits a leave-one-out predictor $g_{-k}^{(t)}$ and computes the consistency residual

$$\rho_k^{(t)} = \left| a_k^{(t)} - g_{-k}^{(t)}(p_k^{(t)}) \right|. \tag{17}$$

Let $\mu_t(p)$ denote the benign score-update relation, i.e., the expected update abnormality of a benign client with risk pressure p at round t.

Assumption D.1 (calibration of leave-one-out prediction). For a low-risk pressure region $\mathcal{P}_{\text{low}}^{(t)}$, the leave-one-out predictor satisfies

$$\sup_{p \in \mathcal{P}_{\text{low}}^{(t)}} \left| g_{-k}^{(t)}(p) - \mu_t(p) \right| \le \varepsilon_t \tag{18}$$

with probability at least $1 - \delta_t$.

Assumption D.2 (attack-induced update separation). A risk-forging malicious client m reports $p_m^{(t)} \in \mathcal{P}_{\text{low}}^{(t)}$, but its submitted update remains separated from the benign score-update relation:

$$\left| a_m^{(t)} - \mu_t(p_m^{(t)}) \right| \ge \Delta_t. \tag{19}$$

Theorem 1 (Residual separation under low-risk forgery)**.** *Under Assumptions D.1 and D.2, if*

$$\Delta_t > \varepsilon_t + \theta_t, \tag{20}$$

then the malicious client satisfies

$$\rho_m^{(t)} > \theta_t \tag{21}$$

with probability at least $1 - \delta_t$.

Proof. By adding and subtracting $\mu_t(p_m^{(t)})$ and applying the reverse triangle inequality,

$$\begin{aligned}
\rho_m^{(t)} &= \left| a_m^{(t)} - g_{-m}^{(t)}(p_m^{(t)}) \right| \\
&= \left| a_m^{(t)} - \mu_t(p_m^{(t)}) + \mu_t(p_m^{(t)}) - g_{-m}^{(t)}(p_m^{(t)}) \right| \\
&\geq \left| a_m^{(t)} - \mu_t(p_m^{(t)}) \right| - \left| g_{-m}^{(t)}(p_m^{(t)}) - \mu_t(p_m^{(t)}) \right|.
\end{aligned} \tag{22}$$

On the calibration event in Eq. (18), the second term is at most ε_t. By Eq. (19), the first term is at least Δ_t. Therefore,

$$\rho_m^{(t)} \geq \Delta_t - \varepsilon_t > \theta_t. \tag{23}$$

Since the calibration event holds with probability at least $1 - \delta_t$, the claim follows. □

The theorem shows that low-risk forgery fails when the submitted update remains sufficiently separated from the benign low-risk update relation. The attacker can reduce the reported pressure $p_m^{(t)}$, but this also lowers the predicted abnormality $g_{-m}^{(t)}(p_m^{(t)})$. If the update abnormality $a_m^{(t)}$ still reflects backdoor-effective training, the mismatch appears as a large residual.

We next connect the residual to the verified risk used by aggregation. Recall that

$$\tilde{\rho}_k^{(t)} = \frac{\rho_k^{(t)} - \mathrm{median}_j\{\rho_j^{(t)}\}}{\mathrm{MAD}_j\{\rho_j^{(t)}\} + \epsilon}, \qquad s_k^{(t)} = \exp\left(-[\tilde{\rho}_k^{(t)}]_+\right), \tag{24}$$

and

$$\hat{r}_k^{(t)} = p_k^{(t)}(1 + \lambda_c s_k^{(t)}) + \lambda_c \frac{1 - s_k^{(t)}}{1 + p_k^{(t)}}. \tag{25}$$

Corollary 1 (Verified-risk penalty from residual inconsistency)**.** *If* $\tilde{\rho}_m^{(t)} > c$ *for some* $c > 0$*, then*

$$\hat{r}_m^{(t)} \geq p_m^{(t)} + \lambda_c \frac{1 - e^{-c}}{1 + p_m^{(t)}}. \tag{26}$$

Consequently, the unnormalized aggregation weight $\tilde{\omega}_m^{(t)} = n_m \exp(-\gamma_t \hat{r}_m^{(t)})$ *satisfies*

$$\tilde{\omega}_m^{(t)} \leq n_m \exp\left(-\gamma_t \left[p_m^{(t)} + \lambda_c \frac{1 - e^{-c}}{1 + p_m^{(t)}} \right] \right). \tag{27}$$

Proof. If $\tilde{\rho}_m^{(t)} > c$, then $s_m^{(t)} = \exp(-[\tilde{\rho}_m^{(t)}]_+) < e^{-c}$. Substituting this into Eq. (25) gives

$$\begin{aligned}
\hat{r}_m^{(t)} &= p_m^{(t)}(1 + \lambda_c s_m^{(t)}) + \lambda_c \frac{1 - s_m^{(t)}}{1 + p_m^{(t)}} \\
&\geq p_m^{(t)} + \lambda_c \frac{1 - e^{-c}}{1 + p_m^{(t)}}.
\end{aligned} \tag{28}$$

Since $\exp(-\gamma_t x)$ is monotonically decreasing in x, substituting the lower bound on $\hat{r}_m^{(t)}$ yields Eq. (27). □

Corollary 1 explains why consistency verification remains effective under low-risk forgery. Even when $p_m^{(t)}$ is forged to be small, a large residual decreases the consistency score and introduces the additional term $\lambda_c(1 - s_m^{(t)})/(1 + p_m^{(t)})$ in the verified risk. This term directly reduces the aggregation weight through the exponential penalty.

Diagnostic evidence. We evaluate the above mechanism using the same risk-forging setting as Table 2a. For each client-round after the warm-up period, we compute the residual ρ, normalized residual $\tilde{\rho}$, consistency score s, verified risk $\hat{r}$, and aggregation penalty factor. Malicious labels are used only for this offline diagnostic.

Residual AUC measures how well ρ separates malicious from benign client-rounds. Malicious Catch and Benign FP are computed using the fixed event $\tilde{\rho} > 1$, which corresponds to one robust MAD above the round-level residual median. Penalty Ratio is defined as

$$\text{Penalty Ratio} = \frac{\mathbb{E}[\exp(-\gamma_t \hat{r}_k^{(t)}) \mid \text{malicious}]}{\mathbb{E}[\exp(-\gamma_t \hat{r}_k^{(t)}) \mid \text{benign}]}. \tag{29}$$

Values below one indicate stronger aggregation penalty on malicious client-rounds.

As shown in Table 8, the residual is discriminative under both zero and median risk forging. Residual AUC is above 0.78 in all settings, while Benign FP remains below 0.10. This indicates that forged low risks often create observable score-update inconsistency rather than merely reflecting benign non-IID variation. The Penalty Ratio is below one in all six settings, confirming that the verified-risk penalty usually translates into smaller aggregation factors for malicious client-rounds.

Table 8: Risk-forging consistency diagnostics. Malicious Catch and Benign FP are computed using the fixed diagnostic event $\tilde{\rho} > 1$. Penalty Ratio is an auxiliary aggregation diagnostic; values below one indicate a smaller aggregation factor, i.e., stronger verified-risk penalty, on malicious client-rounds.

Dataset	Forging	Residual AUC ↑	Mal. Catch ($\tilde{\rho} > 1$) ↑	Benign FP ($\tilde{\rho} > 1$) ↓	Penalty Ratio
CiteSeer	zero	0.783	0.659	0.091	0.026
CiteSeer	median	1.000	0.968	0.000	0.968
PubMed	zero	0.952	0.635	0.018	0.866
PubMed	median	0.980	0.466	0.005	0.670
Amz-Photo	zero	0.887	0.697	0.045	0.130
Amz-Photo	median	0.999	0.878	0.000	0.187

E Algorithmic Details and Complexity Analysis

E.1 Algorithmic Details

Algorithm 1 VeriDrift: Consistency-Verified Risk-Aware Training

Input: Rounds T, clients $\mathcal{K}$, local graphs $\{\mathcal{G}_k\}_{k\in\mathcal{K}}$, supervised node sets $\{\mathcal{S}_k\}_{k\in\mathcal{K}}$, quantile β, penalty strength α, warm-up τ, memory decay ζ, verification strength λ_c

Output: Final global model w^T

Notation: $\mathrm{RZ}(\cdot)$ denotes median/MAD robust standardization, and $\mathrm{PT}(\cdot)$ denotes positive-tail normalization

Initialize w^0 and risk memory $M_k^0 \leftarrow 0$ for all $k \in \mathcal{K}$

for $t = 0, 1, \cdots, T-1$ **do**

Select participants $\mathcal{K}_t \subseteq \mathcal{K}$ and broadcast w^t

Client Side:

for $k \in \mathcal{K}_t$ ***in parallel*** **do**

$w_k^t, \mathbf{H}_k^t \leftarrow \textsc{LocalTrain}(w^t, \mathcal{G}_k)$ // collect last-three-layer representations on $\mathcal{S}_k$

$\Delta w_k^t \leftarrow w_k^t - w^t, \quad n_k \leftarrow |\mathcal{S}_k|$

$\mathbf{D}_k^t, \mathbf{U}_k^t \leftarrow \textsc{CrossLayerDrift}(\mathbf{H}_k^t, \mathcal{S}_k)$ // late displacement and directional turning

$\mathbf{r}_k^{\mathrm{node},t} \leftarrow \frac{1}{2}\big(\mathrm{PT}(\mathbf{D}_k^t) + \mathrm{PT}(\mathbf{U}_k^t)\big)$

$q_k^t \leftarrow Q_\beta(\mathbf{r}_k^{\mathrm{node},t}), \quad r_k^t \leftarrow q_k^t$

Upload only $(\Delta w_k^t, r_k^t, n_k)$ to the server // raw graphs, labels, and node-level representations stay local

Server Side:

$\mathbf{A}^t \leftarrow \textsc{UpdateAbnormality}\big(\{\Delta w_k^t\}_{k\in\mathcal{K}_t}, \{M_k^t\}_{k\in\mathcal{K}_t}\big)$ // leave-one-out low-risk update center; see Eqs. (7)–(11)

$\mathbf{P}^t \leftarrow \mathrm{softplus}(\mathrm{RZ}(\{r_k^t\}_{k\in\mathcal{K}_t}))$ // submitted drift-risk pressure

$\boldsymbol{\rho}^t \leftarrow \textsc{LOOIsoResidual}(\mathbf{P}^t, \mathbf{A}^t)$ // leave-one-out isotonic residual; see Eq. (12)

$s_k^t \leftarrow \exp(-[\mathrm{RZ}(\rho_k^t)]_+), \quad \hat{r}_k^t \leftarrow p_k^t(1 + \lambda_c s_k^t) + \lambda_c \dfrac{1 - s_k^t}{1 + p_k^t}$ // consistency-verified risk

$\gamma_t \leftarrow \alpha(1 - \exp(-t/\tau))$

$\omega_k^t \leftarrow \dfrac{n_k \exp(-\gamma_t \hat{r}_k^t)}{\sum_{j\in\mathcal{K}_t} n_j \exp(-\gamma_t \hat{r}_j^t)}$ for all $k \in \mathcal{K}_t$ // verified-risk-aware aggregation weight

$w^{t+1} \leftarrow w^t + \sum_{k\in\mathcal{K}_t} \omega_k^t \Delta w_k^t$

$M_k^{t+1} \leftarrow \zeta M_k^t + (1-\zeta)\hat{r}_k^t$ for all $k \in \mathcal{K}_t$

$M_k^{t+1} \leftarrow M_k^t$ for all $k \in \mathcal{K} \setminus \mathcal{K}_t$

return w^T

E.2 Complexity Analysis

We analyze the per-round complexity of VeriDrift. Let K denote the number of clients, $G_k = (V_k, E_k)$ denote the local graph of client k, P denote the number of model parameters, and $M = \sum_{k=1}^K n_k$ denote the total number of locally supervised nodes used for risk computation. Since the GNN depth and hidden dimension are fixed in our experiments, we absorb them into constant factors.

First, the standard local training cost of federated GNN learning is

$$\mathcal{C}_{\mathrm{train}} = O\left(\sum_{k=1}^{K} \tau_k(|E_k| + |V_k|)\right), \tag{30}$$

where τ_k is the number of local epochs for client k. This term is the baseline cost shared by ordinary FedAvg-style FGL training.

Client-side drift-risk computation. After local training, VeriDrift performs one inference-only GNN forward pass to collect layer-wise representations, and then computes node-level drift scores

and the local high-risk quantile q_k as the client-level risk summary. The client-side overhead is

$$\mathcal{C}_{\text{drift}} = O\left(\sum_{k=1}^{K}(|E_k| + |V_k|) + M\right). \tag{31}$$

The first term is the extra no-gradient forward pass, and the second term covers drift statistics over supervised nodes and the local high-risk quantile computation. Since the client-level risk is summarized by the local high-risk quantile, this step does not require cross-client prototype comparison.

Server-side consistency verification. On the server side, VeriDrift processes submitted model updates, verifies risk-update consistency, and performs risk-aware aggregation. The server-side overhead is

$$\mathcal{C}_{\text{ver}} = O(KP + K^2 \log K). \tag{32}$$

Here, KP comes from flattening, comparing, and aggregating K model updates, while $K^2 \log K$ is a conservative cost for leave-one-out monotone consistency fitting over client-level scalar scores. The leave-one-out update center is computed by precomputing the weighted sum of all normalized updates and subtracting the target client's contribution, so all centers require $O(KP)$ vector operations in total rather than $O(K^2P)$. Combining the baseline training cost and the two defense components, the total per-round complexity is

$$\mathcal{C}_{\text{total}} = O\left(\sum_{k=1}^{K}(\tau_k + 1)(|E_k| + |V_k|) + M + KP + K^2 \log K\right). \tag{33}$$

Under the common setting where the GNN architecture is fixed, the number of participating clients per round is small, and local graphs are sparse, we have $|E_k| = O(|V_k|)$ and treat K-dependent client-level terms as constants. The total per-round complexity can then be simplified as

$$\mathcal{C}_{\text{total}} = O\left(\sum_{k=1}^{K}(\tau_k + 1)|V_k| + P\right). \tag{34}$$

For a fixed backbone and a fixed number of participating clients, the defense overhead beyond standard local training is therefore approximately linear in the total local graph size and the number of model parameters.

F Broader Impacts

This work aims to improve the reliability of privacy-preserving federated graph learning by reducing the effect of backdoor updates during collaborative training. Its potential positive impact is to make graph-based systems more robust in settings where raw graph data cannot be centralized, such as recommendation, scientific collaboration, and biomedical analysis. At the same time, studying backdoor behavior and adaptive risk forging has an inherent dual-use aspect: the same analysis may help adversaries better understand weaknesses in existing defenses. We mitigate this risk by framing the method and experiments around defense, using public benchmark datasets, and avoiding private user data or deployment-specific attack guidance. Since risk-aware aggregation may down-weight benign but unusual clients, practical use should include calibration, monitoring, and human oversight to avoid suppressing legitimate heterogeneity. The method should be viewed as a robustness component rather than a standalone guarantee of security, fairness, or privacy.

G Visualization Results

To further illustrate the behavior of VeriDrift under heterogeneous federated graph backdoor settings, we visualize the distributions of cross-layer drift indicators using Kernel Density Estimation (KDE) [67, 68]. The visualizations cover six benchmark datasets under the Non-IID Louvain setting and four representative trigger types, including Renyi [64], GTA [36], WS [65], and BA [66]. Each figure reports late displacement, directional turning, and node-level risk, showing that poisoned samples often induce observable cross-layer propagation anomalies, while the discriminative indicator may vary across datasets and trigger topologies.

Cora

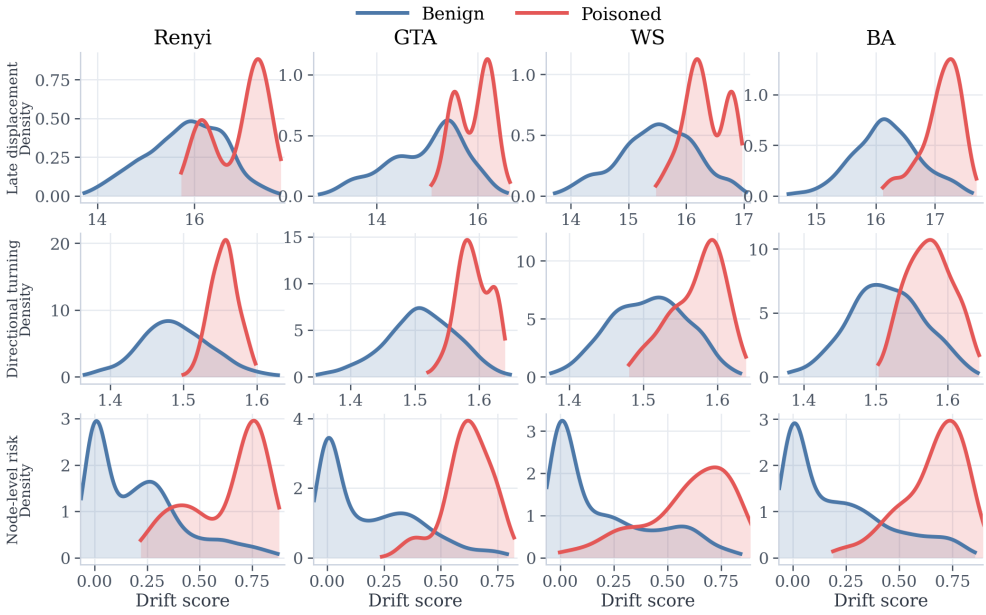

Figure 5: Cross-layer drift distributions on Cora under the Non-IID Louvain setting.

PubMed

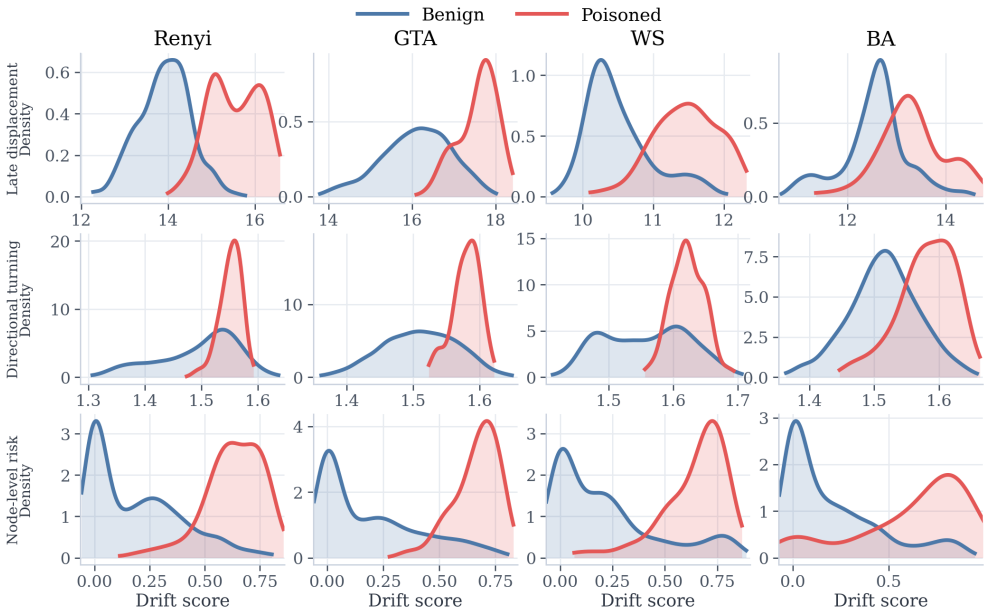

Figure 6: Cross-layer drift distributions on PubMed under the Non-IID Louvain setting.

CiteSeer

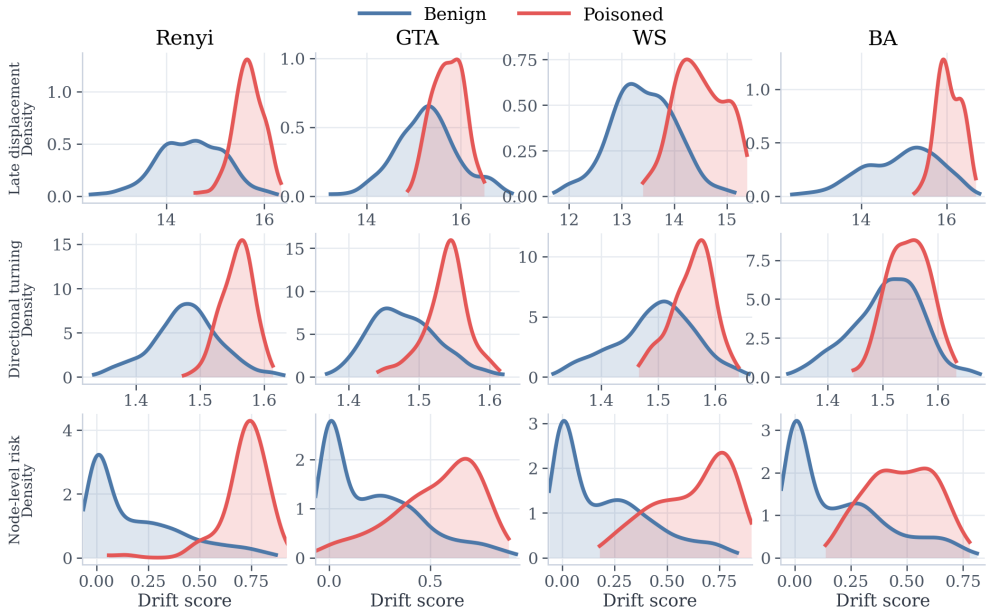

Figure 7: Cross-layer drift distributions on CiteSeer under the Non-IID Louvain setting.

Coauthor-CS

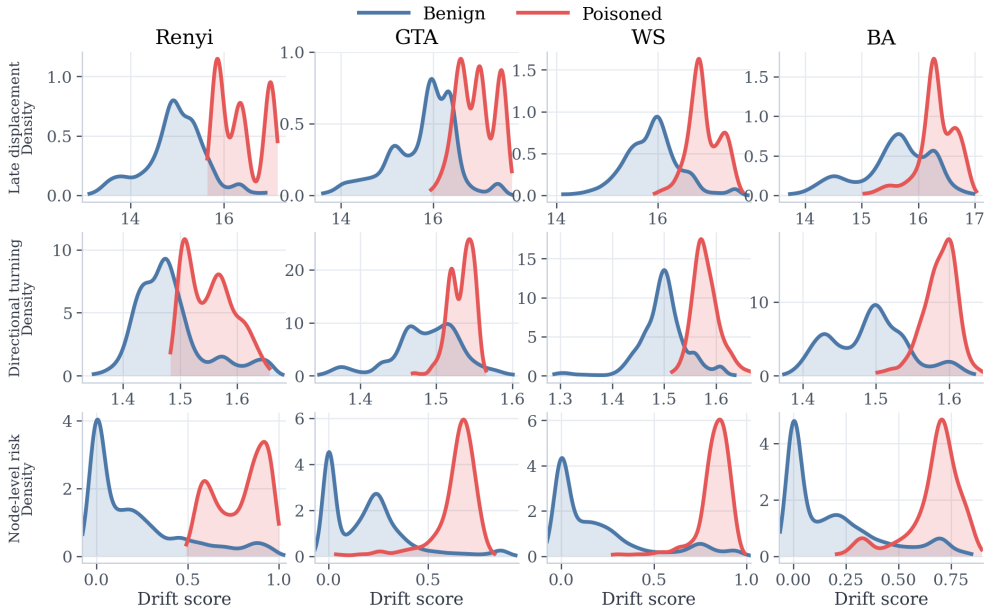

Figure 8: Cross-layer drift distributions on Coauthor-CS under the Non-IID Louvain setting.

Amz-Photo

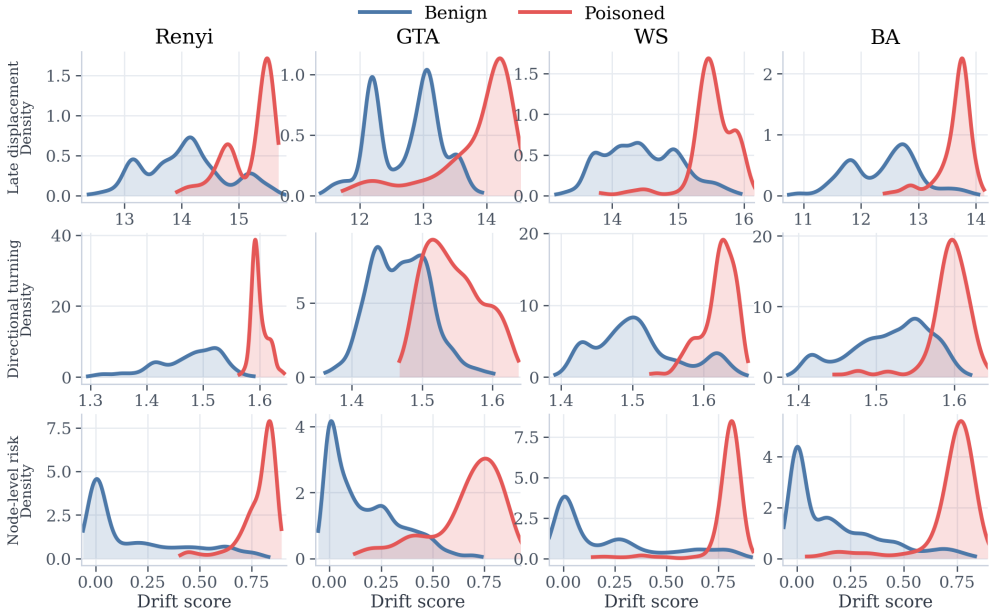

Figure 9: Cross-layer drift distributions on Amz-Photo under the Non-IID Louvain setting.

Coauthor-Phy

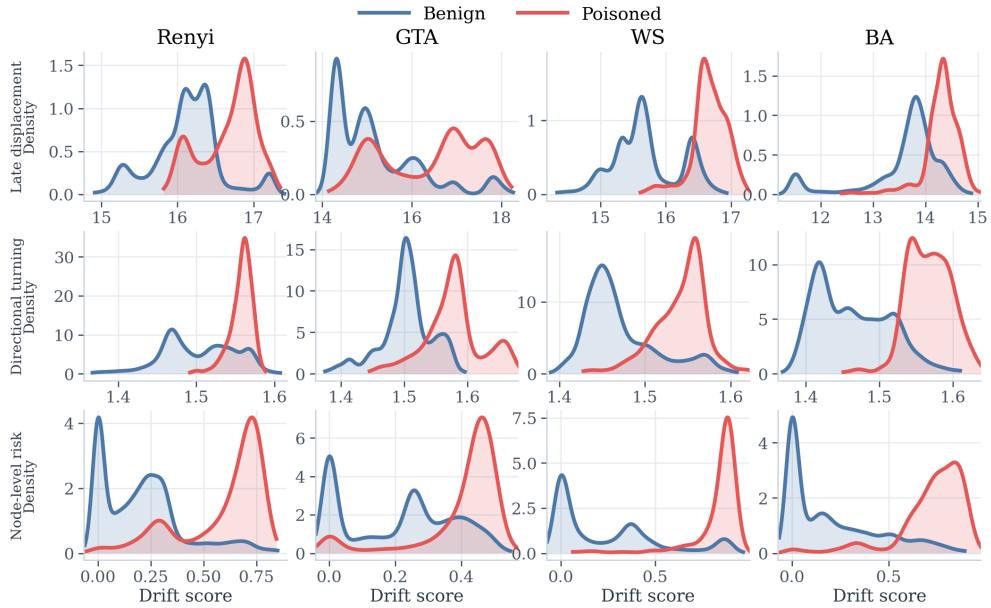

Figure 10: Cross-layer drift distributions on Coauthor-Phy under the Non-IID Louvain setting.

H Additional Discussion and Results

H.1 Partial Participation and GNN Depth

We first examine whether VeriDrift remains effective when only a subset of clients participates in each communication round. We conduct additional experiments on PubMed under Non-IID-Louvain, with a malicious ratio of 0.3 and Renyi triggers. As shown in Table 9, VeriDrift maintains useful defense performance despite the additional sampling variance introduced by partial participation. With 50% participation, it achieves $V = 82.40$, compared with 87.05 under full participation. The non-monotonic variation across participation rates mainly results from the discrete benign/malicious composition of sampled clients in each round. Increasing the federation to 20 clients under the same 50% participation rate gives $V = 77.92$.

Table 9: Results under partial client participation on PubMed with Non-IID-Louvain, a malicious ratio of 0.3, and Renyi triggers.

Clients	Participation	A	R	V
10	1.0	80.80	93.30	87.05
10	0.7	86.83	58.94	72.88
10	0.5	86.28	78.52	82.40
20	0.5	85.83	70.00	77.92

We also evaluate sensitivity to GNN depth and layer choice under 50% participation. We focus on late-layer transitions because they are closest to the classifier and more directly capture representation displacement or redirection before prediction, whereas earlier layers are more strongly affected by raw features and local topology. VeriDrift is not tied to a fixed GNN depth, since the same drift computation can be applied to the final transitions of a deeper model. As shown in Table 10, within the same 4-layer GCN, using the final transition improves V from 69.10 to 76.76 while maintaining comparable clean accuracy. The 4-layer model also retains useful defense performance, showing that VeriDrift extends beyond the default 3-layer architecture without modifying the defense.

Table 10: Sensitivity to GNN depth and layer choice under 50% client participation on PubMed.

GCN depth	Risk transition	A	R	V
3 layers	Final transition	86.28	78.52	82.40
4 layers	Earlier, $h_2 \to h_3$	86.24	51.96	69.10
4 layers	Final, $h_3 \to h_4$	85.79	67.72	76.76

H.2 Benign and Poisoning-Induced Cross-Layer Drift

Benign graph heterogeneity, heterophily, and difficult boundary nodes can also produce nonzero cross-layer drift. We therefore do not assume that drift is unique to backdoors. Instead, we examine whether poisoning introduces measurable distortion beyond such benign variation.

We define hard benign nodes as the lowest-margin 20% within each client using an independently trained clean FedAvg model, with the grouping fixed before attack injection. Table 11 shows that poisoned nodes exhibit substantially larger median drift than both ordinary and hard benign nodes. The lower AUROC against hard benign nodes on PubMed also confirms that overlap can occur, motivating the use of drift together with server-side verification rather than as a standalone malicious decision.

Table 11: Cross-layer drift of ordinary benign, hard benign, and poisoned nodes. Hard benign nodes denote the lowest-margin 20% within each client.

Dataset	Ordinary	Hard benign	Poisoned	AUROC vs.benign	AUROC vs.hard
CiteSeer	0.040	0.186	0.653	0.916 ± 0.028	0.853 ± 0.049
PubMed	0.022	0.357	0.674	0.788 ± 0.087	0.569 ± 0.148
Amz-Photo	0.098	0.128	0.577	0.853 ± 0.116	0.831 ± 0.086

We further compare the same nodes before and after poisoning while fixing the partition, initialization, train mask, client order, and training configuration. The median poisoned-minus-clean risk increases are 0.423, 0.421, and 0.444 on CiteSeer, PubMed, and Amz-Photo, respectively, with positive increases for 86.0%, 80.5%, and 87.2% of nodes. For nodes already near the clean decision boundary, the corresponding median increases remain positive at 0.391, 0.244, and 0.363. The same conclusion holds when the clean median/MAD normalization statistics are frozen before poisoning. These results indicate that benign difficulty may create background drift, while poisoning typically adds measurable distortion beyond that variation.

H.3 Robustness against a Stronger Adaptive Attacker

The zero- and median-risk attacks mainly evaluate direct manipulation of the client-reported risk score. Such a scalar can be forged without changing the actual model update, whereas evading server-observable update verification requires modifying the attack-effective update itself. We therefore consider a stronger training-time attacker that jointly preserves the clean and targeted-backdoor objectives while penalizing VeriDrift's drift indicators and the directional/norm abnormality of the submitted model update. We sweep the evasion coefficient over $\lambda \in \{0.5, 1, 2, 5\}$.

Table 12 shows a clear effectiveness–stealth trade-off. For $\lambda \leq 1$, where the undefended backdoor remains strong, malicious updates are still approximately 7–9$\times$ more abnormal than benign updates, allowing VeriDrift to substantially reduce ASR. As the attacker more strongly constrains the update toward benign geometry, its abnormality decreases, but the undefended attack itself also becomes considerably weaker and, in the strongest setting, clean utility can deteriorate. Across the evaluated settings, we do not observe a configuration that simultaneously maintains high ASR, normal clean utility, and benign-like update geometry.

Table 12: Results against the stronger adaptive attacker. ASR denotes attack success rate, ACC denotes clean accuracy, and AbnRatio denotes the malicious-to-benign update abnormality ratio.

Dataset	λ	ASR NoDefense	ASR VeriDrift	ACC NoDefense	ACC VeriDrift	AbnRatio
CiteSeer	0.5	100.0	49.2	72.5	60.5	9.115
CiteSeer	1	76.0	33.3	72.9	62.7	7.032
CiteSeer	2	56.8	43.1	73.4	73.9	1.741
CiteSeer	5	14.9	10.2	41.2	48.1	0.604
PubMed	0.5	100.0	0.0	73.2	75.3	7.472
PubMed	1	93.3	28.0	73.2	85.9	8.077
PubMed	2	50.0	43.8	78.2	80.4	2.081
PubMed	5	2.8	2.4	73.5	52.8	2.236

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims in the abstract and introduction are aligned with the proposed VeriDrift framework, including cross-layer drift risk estimation, server-side consistency verification, and risk-aware aggregation. The scope is supported by experiments across multiple FGL backdoor settings.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our work in Section 4.4

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Section D states the assumptions for the conditional consistency verification analysis and provides a proof showing when low-risk forgery leads to score-update inconsistency.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide a comprehensive and detailed experimental setup in Section 5.1

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Our code and dataset are openly released with clear and detailed instructions, ensuring that others can readily reproduce the experimental findings reported in this paper.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: The paper specifies dataset splits, optimizer, learning rate, repeated runs, evaluation metrics, backdoor settings, malicious ratios, and model architecture in Section 5.1

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We do not report error bars or formal statistical significance tests. The main experimental comparisons are reported under consistent settings across methods, with repeated runs used to improve robustness of the reported results.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports the hardware used for all experiments in Section B.1

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms to the NeurIPS Code of Ethics; it uses public benchmark datasets and studies backdoor attacks in a defensive evaluation setting.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses broader impacts in Appendix F, including the positive role of VeriDrift in improving robust federated graph learning, the potential dual-use risks of studying backdoor behavior, and the mitigation steps taken through public benchmarks, no private user data, and a defense-oriented evaluation.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All external assets used in this paper, including code, data, and models, are appropriately cited and used in accordance with their respective licenses and terms.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The paper releases anonymized code as a new asset.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The work does not involve crowdsourcing, human-subject experiments, or participant compensation.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The work does not involve human-subject research and therefore does not require IRB approval.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method, experiments, and scientific claims do not use LLMs as an important, original, or non-standard research component.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.