Poisoning-based Link Inference Attacks Against Federated Graph Neural Networks

Guizhen Yang
Deakin University

Yanjun Zhang
University of Technology Sydney

Leo Yu Zhang
Griffith University

Mengmeng Ge
Monash University

Shang Gao
Deakin University

Abstract

Federated graph neural networks (FedGNNs) have emerged as a promising solution for handling graph data distributed across multiple owners. They enable collaborative training while preserving data decentralisation and complying with privacy and regulatory constraints. However, the inherent structural dependencies in graph data and the message-passing mechanisms of GNNs introduce both cross-client and intra-client edges in FedGNNs. Cross-client edges, in combination with federated learning (FL) protocol designs, open additional channels for information propagation and heighten the risk of privacy leakage. In FedGNNs, once edge information is compromised, adversaries can infer local neighbourhood structures and reconstruct inter-client relationships, even without direct access to raw data. Existing research on privacy inference in FL has largely overlooked edge privacy threats specific to FedGNNs. To address this gap, we propose a poisoning link inference approach with two strategies: Label Flipping Link Inference Attack (LFLIA) and Gradient Ascent Link Inference Attack (GALIA). LFLIA flips the label of a candidate node so that its perturbation propagates along structural topology during training. GALIA perturbs the candidate node's gradient to amplify its loss. The perturbations on the candidate node can propagate to its linked neighbours by message-passing mechanism, which induces representation shifts on these linked nodes. By monitoring FedGNN outputs of a target node set before and after poisoning, an adversary can distinguish linked nodes through observable output shifts, whereas unlinked nodes exhibit little to no change. Experimental results on multiple benchmark datasets show that our poisoning-based LIA can effectively infer link existence and structure with high accuracy across diverse federated settings.

Keywords

Poisoning Inference Attack, Link Inference Attack, Privacy Leakage, Federated Graph Neural Networks

1 Introduction

Graph neural networks (GNNs) have demonstrated remarkable effectiveness in learning graph data [1–3], such as recommendation systems [4], disease discovery [5–7]), and social networks [8, 9]. However, in real-world scenarios, graph data is often distributed across multiple data holders (e.g., hospitals or institutes) that cannot share raw data due to privacy regulations or commercial constrains (e.g., HIPAA [10, 11], GDPR [12]). To address this, federated graph neural networks (FedGNNs) have been proposed to enable collaborative model training across decentralised graph data without raw data sharing [13–16].

Despite mitigating the risk of data centralisation, FedGNNs amplify privacy issues due to the structural dependencies in graph data [17] and the message passing mechanism [18] over both cross-client and intra-client edges [19–21]. Cross-client edges connect nodes located across multiple clients, while intra-client edges connect nodes within the same local subgraph. With this message passing mechanism, node representations can encode not only node features but also the structural dependencies of their neighbours from both the same local subgraph and across clients.

Since node representations may unintentionally encode sensitive user attributes and relational structures, which make them vulnerable to adversaries to exploit these representations to steal private data information [22–24] (e.g., link inference attacks [25], attribute inference attacks, and property inference attacks [26]). In particular, link inference attacks (LIAs) [27, 28] aim to infer whether an edge or link exists between two nodes, which posing serious threats to structural privacy as edges in graphs are often sensitive and private (e.g., social relationships, disease correlations, or behavioural associations).

Recently, poisoning attacks (e.g., data poisoning attacks [29, 30], model poisoning attacks [31, 32]) have emerged as effective privacy threats for privacy inference in federated learning (FL) by amplifying inference signals through crafted perturbations [33, 34]. By acting as malicious clients, adversaries can inject crafted perturbations into local data or model updates to subtly manipulate global model behaviour, and then extract sensitive information by observing behaviour shift before and after attacks. However, existing poisoning-based LIAs and centralised LIAs fundamentally rely on strong assumptions that are not realistic, including access to global node features [27, 28, 35], the ability to inject auxiliary nodes into the centralised graph [28, 35], and knowledge of node embeddings [36].

Unlike standard FL tasks, the unique graph characteristics introduce several challenges for link inference in FedGNNs. Specifically, each client only stores a partial subgraph, with cross-client edges remaining invisible to any single client, resulting in a fragmented topology that prevents direct access to complete neighbourhood information. Furthermore, the influence of these hidden connections is implicitly embedded in the federated aggregation process and entangled with local message passing, making it difficult to isolate the structural contribution of a specific link from aggregated updates.

Proceedings on Privacy Enhancing Technologies 2026(2), 157–179

https://doi.org/10.56553/popets-2026-0042

To bridge this gap and overcome these challenges, we propose a poisoning-based LIA approach for FedGNNs that leverages the propagation of crafted perturbations through both intra-client and cross-client dependencies. Although these dependencies are not directly observable, they still shape learned node representations. By amplifying the link-specific effects, our approach renders structural signals in aggregated updates distinguishable even under incomplete topology. The key intuition is that perturbations injected into a candidate node can propagate through both local intra-client and hidden cross-client dependencies during message passing, leaving measurable traces in the representations or model outputs of its neighbours.

We design two poisoning strategies: label-flipping link inference attack (LFLIA) and gradient-ascending link inference attack (GALIA). LFLIA flips the label of a candidate node so that its perturbation propagates along structural topology during training. GALIA perturbs the candidate node's gradient to amplify its loss and propagate the perturbations to its neighbours. Both strategies cause perturbations to propagate through the FedGNN message-passing and aggregation process, enabling the adversary to observe behavioural shifts indicative of link existence.

Fig. 1 illustrates a healthcare system under the FedGNN architecture. An adversary has compromised hospital k and control its local training processing. The adversary wants to know patients' private information, such as if patient u (in red) is also the patient of other hospitals or if a patience v (in green) has relationship with u. The adversary injects perturbations into u. This perturbations propagates to u's neighbours via both intra-client and cross-client edges. If a link or an edge exists between u and v (e.g., a therapeutic relationship, a family link, or the same patient appearing in multiple hospitals), the perturbation on u induces observable changes in v's output. Otherwise, no shift occurs. By monitoring these behavioural shifts, an adversary can infer the existence of a link between u and v, thereby stealing sensitive structural relationships in FedGNNs. Our contributions can be summarised as follows:

- We propose a poisoning-based LIA approach for FedGNNs with two poisoning strategies: label flipping link inference attack (LFLIA) and gradient ascending link inference attack (GALIA). LFLIA flips the label of the target node to mislead supervision propagation through message passing, which exploits structural dependencies to reveal link-specific influence. GALIA perturbs the target node's gradient via loss amplification, which amplifies its impact on neighbouring representations to induce and trace representation shifts via message passing.
- We explore the impact of inferred structural relationships to subgraph reconstruction. Based on the inferred relations from our poisoning-based LIA, we investigate to construct subgraphs via breadth-first search and evaluate reconstruction fidelity using embedding similarity.
- We conduct comprehensive evaluation across GNN architectures, robust FL aggregators, and both homogeneous and heterogeneous graphs on five benchmark graph datasets. We also assess attack performance under existing defense strategies. Results show that both poisoning strategies achieve higher precision than existing LIAs (i.e., SLA [25], LTA [27]) in most settings (e.g., GALIA achieves 91% precision and 92% recall on Citeseer under FedGCN). We also analyse the impact of various factors on attack performance, including number of adversaries, node overlapping rate, and attack timing.

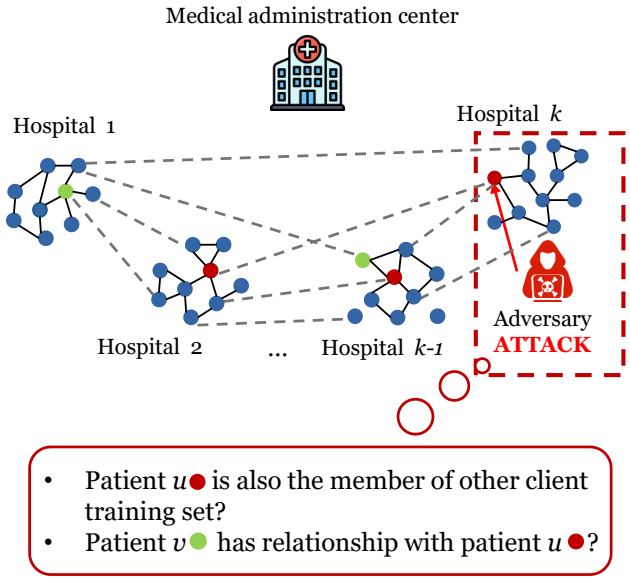


Figure 1: A healthcare system under the FedGNN architecture. In the global graph, nodes represent patients, with solid lines denoting intra-hospital patient relationships and dotted lines representing inter-hospital patient relationships. Each hospital trains a local GNN model that contributes to the federated learning process coordinated by the medical administration centre.

The rest of the paper is organised as follows. Section 2 introduces the preliminaries in FL, GNNs, and FedGNNs. Section 3 presents the threat model in terms of an adversary's knowledge, capabilities, and attack goals. Section 4 details our proposed poisoning-based LIAs. Section 5 describes the experiment setup, experimental results, and analysis. Section 6 presents the ablation study. Section 7 discusses the ethical considerations of our work. Section 8 discusses limitations and future work. Section 9 reviews existing LIAs against GNNs. Finally, Section 10 concludes the paper.

2 Preliminaries

2.1 Federated Learning

Federated Learning (FL) enables K clients to collaboratively train a global model $\mathbf{W}$ while keeping their datasets localised [31, 37, 38]. Unlike centralised learning, where all data are collected by a central server, FL trains a model in a decentralised manner. Each client k trains a local model using its local dataset and uploads its local model parameters $\{\mathbf{W}_k \mid k \in K\}$ to a parametric server. The global

optimisation objective of FL can be formulated as:

$$\min_{\mathbf{W}} \mathcal{L}(\mathbf{W}) = \sum_{k=1}^{K} \frac{b_k}{B} \mathcal{L}_k(\mathbf{W}_k), \quad \mathcal{L}_k(\mathbf{W}_k) = \frac{1}{b_k} \sum_{j \in P_k} \ell(\mathbf{W}_k, x_j), \tag{1}$$

where $B = \sum_{k=1}^{K} b_k$ is the total data size, $\mathcal{L}_k(\mathbf{W}_k)$ represents the local loss of client k, $b_k = |P_k|$ is the local data size of client k, and P_k represents the index set of data samples held by client k. For each sample $x_j \in P_k$, the local loss is $\ell(\mathbf{W}_k, x_j)$.

At communication round t, FL proceeds in three stages:

(1) **Global model download.** The server sends the current global model parameters $\mathbf{W}^t$ to all participating clients.
(2) **Local training.** Each client updates the model by training with its local data via gradient descent,
$$\mathbf{W}_k^{t+1} \leftarrow \mathbf{W}^t - \eta \nabla \mathcal{L}_k(\mathbf{W}^t), \tag{2}$$
where η is the learning rate.
(3) **Model aggregation.** Clients upload their updated local model parameters $\{\mathbf{W}_k^{t+1}\}_{k=1}^{K}$ to the server. Then, the server aggregates the uploaded local model parameters (e.g., via FedAvg [39]) to produce new global model parameter $\mathbf{W}^{t+1}$.

2.2 Graph Neural Networks

Graph neural networks (GNNs) have emerged as powerful tools for processing non-Euclidean data by leveraging both graph structural information and node features to learn node representations [2, 40, 41]. We briefly introduce the basic notions and general architectures of GNNs.

A graph is defined as $G = (V, E, \mathbf{X})$, where V is the set of nodes, E is the set of edges, and $\mathbf{X} \in \mathbb{R}^{|V| \times d}$ is the node feature matrix with d-dimension input features. Each row $\mathbf{x}_v \in \mathbf{X}$ corresponds to the initial feature vector of node $v \in V$, which encodes features (e.g., user profiles, gender, age). $|V|$ and $|E|$ denote the number of nodes and edges, respectively. L-layer GNNs learn a node representation $\mathbf{h}_v$ (i.e., node embedding) for each node $v \in V$ by iteratively aggregating information from its neighbours through message passing. The initial representation is set as $\mathbf{h}_v^{(0)} = \mathbf{x}_v$, and updated layer by layer as:

$$\mathbf{h}_v^{(l+1)} = \text{AGG}\Big(\mathbf{h}_v^{(l)}, \{\mathbf{h}_u^{(l)} : u \in \mathcal{N}(v)\}\Big), l \in L \tag{3}$$

where $\mathbf{h}_v^{(l+1)}$ is the hidden representations of node v at layer l. $\mathcal{N}(v)$ denotes the neighbours of v, and AGG represents a neighbourhood aggregation function (e.g., mean, max, or sum). Then, the node representation $\mathbf{h}_v$ is,

$$h_v = h_v^{(L)}, \tag{4}$$

which $\mathbf{h}_v$ captures both node v's own features and its structural information of its neighbours.

2.3 Federated Graph Neural Networks

Federated Graph Neural Networks (FedGNNs) enables distributed clients to collaboratively train a global GNN model without sharing their raw graph data [42–44]. A typical FedGNN primarily consists of two levels: client-side local training and server-side federated optimisation as illustrated in Fig. 2 with K clients and one server. Let the full dataset be $D = (G, Y) = ((V, E, \mathbf{X}), Y)$. Client k has its own dataset $D_k = (G_k, Y_k)$ with $G_k = (V_k, E_k, \mathbf{X}_k)$, where $V_k \subseteq V$,$E_k \subseteq E$, and $\mathbf{X}_k$ is the feature matrix restricted to V_k. Y_k is the corresponding label set of V_k.

2.3.1 Client-side Local Training. The client-side training follows the standard message passing paradigm [45], which includes two stages: message passing and readout. Generally, the message passing neural networks (MPNN) are employed to cover a range of GNN variants, such as GCN [40], GAT [41], and GraphSAGE [2]. In client-level local training, each client k processes a local GNN model f_k and performs local training.

Message passing: For node $i \in V_k$ in client k, the l-layer of the GNN conducts aggregation over its neighbours $N(i)$ as follows,

$$\mathbf{m}_{k,i}^{(l)} = \text{AGG}\left(\left\{\mathbf{h}_{k,j}^{(l-1)} \mid j \in \mathcal{N}(i)\right\}\right), \tag{5}$$

$$\mathbf{h}_{k,i}^{(l)} = \sigma\left(\mathbf{W}_k^{(l)} \cdot \text{COMBINE}\left(\mathbf{h}_{k,i}^{(l-1)}, \mathbf{m}_{k,i}^{(l)}\right)\right), \tag{6}$$

where $\mathbf{m}_{k,i}^{(l)}$ denotes the aggregated message from the neighbours of node i at layer l. $\mathbf{h}_{k,i}^{(l)}$ is the hidden representation of node v_i at layer l, and $\mathbf{h}_{k,i}^{(0)} = \mathbf{x}_i$ is the initial node feature. AGG is the aggregation function to aggregate the messages from hidden representations of neighbours at previous layer $l-1$ by using operations like sum, mean, or max. COMBINE is a concatenation function to combine node i's own representation with the aggregated messages from its neighbours j at previous layer $l-1$ to update node i's representation at layer l. σ is an activate function (e.g., ReLU). $\mathbf{W}_k^{(l)}$ is the trainable parameters at layer l of client k.

Readout: After L layers message passing, the readout phase generates task-specific outputs. For node classification, the prediction for node v_i is defined as,

$$\hat{y}_{k,i} = R_{\theta_k}(\mathbf{h}_{k,i}^{(L)}), \tag{7}$$

where R_{θ_k} is the readout function or classification function (e.g., an MLP with softmax), θ_k is the parameters of readout function, $\hat{y}_{k,i}$ is the predicted label of node i.

2.3.2 Sever-side Federated Optimisation. Following local training, clients collaboratively update a global model using a standard FL protocol (e.g., FedAvg [39]). Let $\mathbf{W}_k = \{\mathbf{W}_k^{(l)}\}_{l=1}^{L} \cup \{\theta_k\}$ denote the model parameters of client k's local GNN. The local objective can be defined as:

$$\mathcal{L}_k(\mathbf{W}_k) = \frac{1}{|D_k|} \sum_{(G_k, Y_k) \in D_k} \ell(f_k(G_k; \mathbf{W}_k), Y_k), \tag{8}$$

where $\ell(\cdot)$ is the loss for each sample in D_k (e.g., cross-entropy). $\mathcal{L}_k(\mathbf{W}_k)$ the local loss of client k that averages loss over client k's dataset D_k.

The global objective minimises global loss of $\mathcal{L}(\mathbf{W})$ by aggregates all local updates through a protocol (e.g., FedAvg [39]).

$$\min_{\mathbf{W}} \mathcal{L}(\mathbf{W}) = \sum_{k=1}^{K} \frac{|D_k|}{|D|} \cdot \mathcal{L}_k(\mathbf{W}_k), \tag{9}$$

where $\mathbf{W}$ is the global model parameter. $|D| = \sum_{k=1}^{K} |D_k|$ is the total dataset size across all clients. $|D_k|$ is the dataset size of client k.

In each training round t, the server broadcasts the current global parameter $\mathbf{W}^t$ to participating clients. Each client k performs local

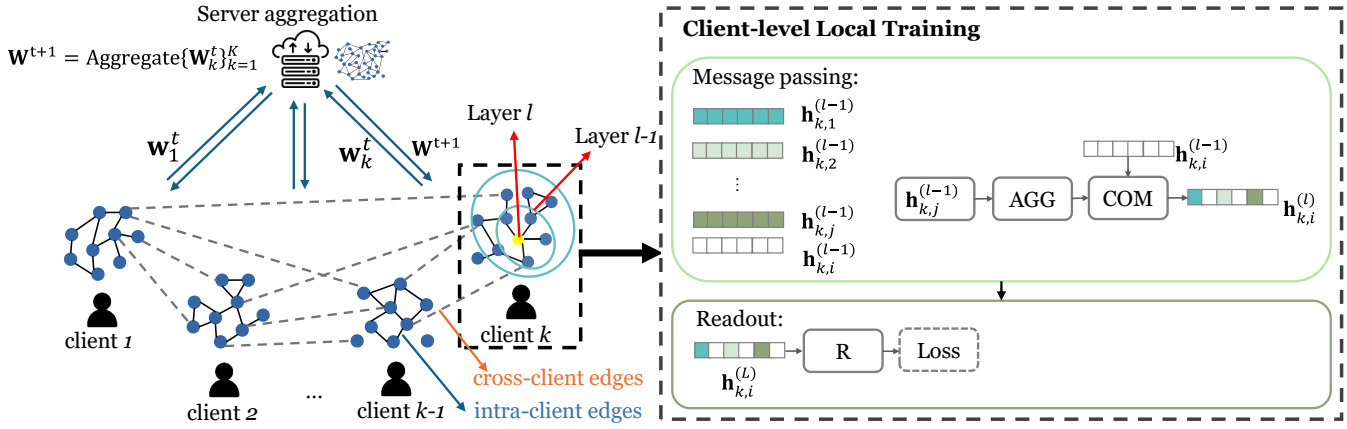


Figure 2: Illustration of FedGNN.

training on its local subgraph G_k to obtain updated parameters $\mathbf{W}_k^{t+1}$ according to Eq. 2, which is then uploaded to the server. The server aggregates all the uploaded updates from clients to update global parameters $\mathbf{W}^{t+1}$.

3 Threat Model

We characterise the threat model in terms of the adversary's knowledge, capability, and goals. In this work, we focus on client-side attacks that $\mathcal{A}$ can compromise one or more clients and participate in the FedGNN model training process. As a participating client in FedGNN, $\mathcal{A}$ has the same view of FL protocol as honest clients. Specifically, $\mathcal{A}$ has full access to its own local dataset and is able to perturb its local node features, labels, and the local graph structure.

3.1 Adversary's knowledge

The adversary's knowledge about FedGNN varies depending on the access setting, we consider two settings in terms of our two poisoning strategies,

- **Black-box:** $\mathcal{A}$ can only query the global model via APIs and observe its output predictions (e.g., posteriors, confidence score, etc.) for given inputs, but without access to the model parameters, gradients, and updates of either from the global model nor other clients. $\mathcal{A}$ can only control and manipulate its local dataset D_k.
- **White-box:** $\mathcal{A}$ can not only control its local dataset D_k, but also observe and manipulate its own local training process (e.g., local updates ∇_k), and view global parameters $\mathbf{W}$, but cannot access other clients' datasets or local updates.

3.2 Adversary's capability

We define the adversary's capability based on what $\mathcal{A}$ can control and view under its knowledge.

- **Generate local auxiliary nodes.** $\mathcal{A}$ can generate auxiliary nodes N_a within its own client by adding new records to its local dataset. Auxiliary-node injection is realistic in many real-world FL deployments, because clients typically control over their own data repositories. The auxiliary nodes may be linked to local nodes (e.g., by forming plausible local relationships) or left isolated, and their features can be crafted to support the attack objectives. Importantly, the location of auxiliary-node injection is strictly limited to the adversary's own client, $\mathcal{A}$ cannot modify other clients' subgraphs or change cross-client topology.
- **Conduct local perturbation.** Since $\mathcal{A}$ can control its local dataset D_k, it can perturb node features, graph topology, or labels. In particular, in a white-box setting, $\mathcal{A}$ also can craft or manipulate its local model parameters or crafted updates during training. This allows $\mathcal{A}$ to steer the global model updates in a malicious direction [46].

The perturbed information associated with the auxiliary nodes propagates through the GNN's message-passing mechanism and the federated aggregation process, ultimately influencing the global model parameters and their linked neighbours. For example, in a federated hospital network, a compromised hospital (i.e., $\mathcal{A}$) can legitimately register new patient accounts that appear as new nodes with plausible attributes and optional relationships (e.g., being assigned to the same ward or treatment pathway), similar to creating new user accounts on social networks (e.g., Facebook, Twitter, Weibo, etc) [35]. Such modifications affect only the local client's subgraph topology and are fully aligned with practical FL data-ownership assumptions.

3.3 Adversary's goal

Given a target node set T and a candidate node u, the $\mathcal{A}$ aims to identify whether there exists a link between u and any node $v \in T$.

4 Methodology

4.1 Overview

In FedGNNs, messages propagate through both intra-client and cross-client edges, where node representations are iteratively updated by recursively aggregating the information from their neighbours in intra- and inter-graph. With this message-passing mechanism, node representations capture not only node features but

also information of neighbours and the underlying graph structure. However, such node representations may be vulnerable to adversaries as the representations may inadvertently encode sensitive patients' attributes and relational structures, making them vulnerable to various privacy inference attacks to steal private data information [17, 18, 22–24]. Fig. 3 illustrates the overview of the

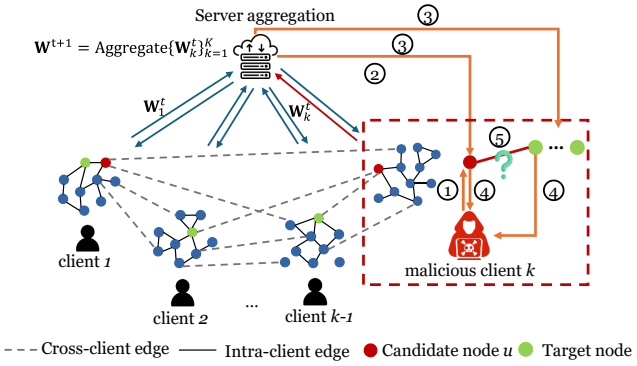


Figure 3: LIA scenario in FedGNNs. (1) Poison candidate node u in its local subgraph, (2) Train the poisoned data and upload local parameters to server for global aggregation, (3) Query the global model to obtain outputs on u and v, (4) Observe changes in u's and v's outputs (e.g., posterior probabilities or gradients), and (5) Infer link existence between u and v if noticeable shifts are observed in v's outputs.

attack scenario of LIA on FedGNNs. Clients with locally trained subgraphs collaboratively train a global graph in FL settings. Within these subgraphs, edges can be categorised into intra-client edges that connect nodes belonging to same subgraph, and cross-client edges that connect nodes across different subgraphs.

LIA attempts to discover neighbours of the candidate node u. The perturbations or poisoning toxicity on u propagate through the graph structure. This effect is further amplified by FedGNN's parameter aggregation across participating clients. The malicious client k can observe outputs of FedGNN on target node set to assess how they are affected.

If a target node v is linked to u, the perturbed messages will propagate bidirectionally. That is, u aggregates information from v, while v absorbs the perturbation from u. Therefore, v's representation and corresponding outputs (e.g., posterior probabilities, confidence scores) will deviate noticeably under sufficient poisoning toxicity. By observing these shifts, the adversary can infer the existence of a link between u and v.

If a target node v is not linked to u, the perturbation cannot directly reach v. v's representation remains stable, and its outputs exhibit negligible changes. Similarly, nodes separated by large hop distances are only weakly affected, which makes FedGNN outputs on these nodes effectively no larger changes compared to linked nodes.

To launch an effective LIA, the attack can be performed when the FedGNN model reaches a near-convergent state. $\mathcal{A}$ first records the baseline outputs (e.g., posterior probabilities or confidence scores) of each node $v \in T$ before perturbing u. Then, $\mathcal{A}$ injects a perturbation into the candidate node u (i.e., by flipping label or applying gradient ascent) with sufficient magnitude to induce noticeable shifts of FedGNN's outputs on v. As illustrated in Fig. 3, given the candidate node u, a target node set T, and suppose the target model FedGNN is almost converged at t-th training round, the attack proceeds as bellows:

Step 1: Baseline recording. At training round $t-1$, $\mathcal{A}$ pulls the global model parameters $\mathbf{W}^{t-1}$, and records FedGNN outputs on each target node $v \in T$ (e.g., posteriors) as baseline values before attacks,

$$O_{t-1}(v) = f(v; \mathbf{W}^{t-1}), \quad \forall v \in T,$$

where $f(\cdot; \mathbf{W}^{t-1})$ denotes the FedGNN prediction function at $(t-1)$-th training round. $O_{t-1}(v)$ is FedGNN outputs for node v at training round $t-1$.

Step 2: Poisoning. Starting from round t, $\mathcal{A}$ poisons candidate node u (i.e., label flipping under a black-box setting or gradient ascent under a white-box setting) for α consecutive rounds $[t, t+\alpha]$. In each poisoned round, $\mathcal{A}$ perturbs its local data, and uploads manipulated local parameters $\mathbf{W}_k^r$ ($r \in [t,\ t+\alpha]$) for global aggregation.

$$\mathbf{W}^r \leftarrow \mathrm{Agg}\big(\{\mathbf{W}_j^r\}_{j=1}^K\big),$$

where $\mathrm{Agg}(\cdot)$ denotes the FL aggregation rule (e.g., FedAvg). The poisoning effect is propagated and accumulated across α rounds.

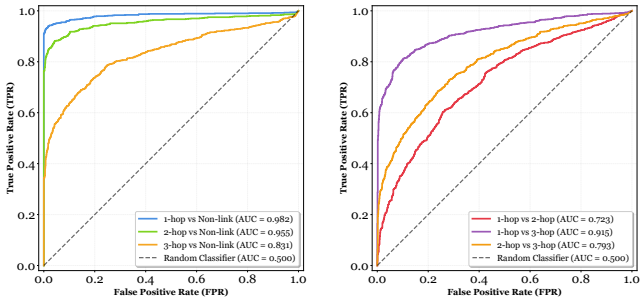
Step 3: Querying. After the poisoning phase ends (i.e., at round $(t+\alpha)$), $\mathcal{A}$ queries the updated global model to obtain post-attack FedGNN predictions on each target node v,

$$O_{t+\alpha}(v) = f(v; \mathbf{W}^{t+\alpha}), \quad \forall v \in T.$$

Step 4: Shift measurement. The output shift is quantified via Kullback–Leibler divergence [47] between pre- and post-perturbation states (i.e., $O_{t-1}(v)$ and $O_{t+\alpha}(v)$). Specifically,

$$\Delta(v) = D_{\mathrm{KL}}\big(O_{t-1}(v) \,\big\|\, O_{t+\alpha}(v)\big), \tag{10}$$

where $D_{\mathrm{KL}}(\cdot\|\cdot)$ denotes the KL divergence between two probability distributions. $\Delta(v)$ is the magnitude of output shift of node v induced by injecting the perturbation into candidate node u. We empirically demonstrate that these KL-based shifts provide strong separability between linked and non-linked node pairs (Fig. 4a) and across different hop distances (Fig. 4b).



(a) ROC for link and non-link node pairs. **(b) ROC for different hop distances.**

Figure 4: ROC of GALIA on Cora dataset under FedAvg.

Step 5: Inference. $\mathcal{A}$ infers the link relationship by comparing $\Delta(v)$ against a decision threshold Γ,

$$\hat{E}(u,v) = \begin{cases} 1, & \text{if } \Delta(v) > \Gamma \quad \text{(linked)}, \\ 0, & \text{otherwise} \quad \text{(not linked)}, \end{cases} \qquad \forall v \in T.$$

where Γ is the link threshold. If node v is linked to u, $\Delta(v)$ will be significantly larger than Γ. Otherwise, if v is not linked to u, the perturbation cannot propagate and $\Delta(v)$ remains close to zero.

As node representations are updated layer by layer through its neighbourhood aggregation, at each layer, a node aggregates feature information from its immediate neighbours, and over multiple layers, this allows information to propagate progressively from increasingly high-hop nodes and enables to capture high-order structural dependencies [45]. However, as the number of layers increases, node representations tend to become indistinguishable [48–50], which causes messages from multiple-hops nodes to be gradually diluted during propagation.

Hence, perturbations injected into the candidate node u also propagate through multiple message-passing layers and weaken as they reach nodes farther away, due to repeated aggregation, transformation, and normalisation across layers. As a result, the impact of the perturbation on high-hop nodes' representations or outputs is less pronounced compared to nodes closely connected to the perturbed node u. This attenuation enables the adversary $\mathcal{A}$ not only to determine whether a target node v is linked to u, but also to estimate the hop distance by measuring the magnitude of output shifts.

Intuitively, nodes closer to the candidate node u are more strongly affected by the perturbation, yielding larger $\Delta(v)$. By comparing these values against a set of thresholds, $\mathcal{A}$ can infer the hop distance. In our work, we consider infer 3-hops neighbour relationships as prior studies on over-smoothing and over-squashing have demonstrated that information from distant nodes quickly vanishes as the number of hops increases. In practice, GNNs usually fail to capture meaningful signals from nodes beyond 6 hops [48].

The hop distance can be inferred as follows,

$$\begin{aligned} &\Delta(v) > \Gamma_1 && \Rightarrow \text{ 1-hop neighbour}, \\ &\Gamma_1 < \Delta(v) < \Gamma_1 && \Rightarrow \text{ 2-hop neighbour}, \\ &\Gamma_3 < \Delta(v) < \Gamma_2 && \Rightarrow \text{ 3-hop neighbour}, \\ &\Delta(v) < \Gamma_3 && \Rightarrow \text{ non-link}. \end{aligned} \tag{11}$$

where Γ_1, Γ_2, Γ_3 are empirically estimated thresholds chosen by the adversary $\mathcal{A}$ to distinguish between 1-hop, 2-hop, 3-hop, and non-linked nodes under the respective attack settings. Since $\mathcal{A}$ participates in the FedGNN training, it has access to its own local subgraph with ground-truth neighbourhood information. By conducting poisoning-based LIA on its controlled local dataset and measuring the output shifts for nodes at known hop distances, $\mathcal{A}$ can derive the thresholds Γ_1, Γ_2, and Γ_3 that are then applied to infer hop distances in global settings.

4.2 Label-flipping link inference attack (LFLIA)

Label-flipping attack is a widely adopted data poisoning method in machine learning [51–54], where an adversary poisons training samples by flipping their labels from a source class to a target class, such as flipping papers' labels "Genetic Algorithms" to "Reinforcement Learning" in Cora dataset for node-level classifications or "Human-Computer Interaction" to "Machine Learning" in Citeseer dataset.

Unlike other data poisoning attacks, label-flipping attack does not require the adversary to know the global data distribution, target model architecture, and the loss function. It is easy to execute, accessible to non-experts and does not require modifying or tampering with FL client software.

We inspire by this to propose a label-flipping link inference attack strategy (LFLIA). In a black-box setting, as the adversary $\mathcal{A}$ has access to its local dataset, $\mathcal{A}$ can conduct targeted poisoning to reveal link relationships. Specifically, given a target node set T, and assuming the client k is the malicious client (i.e., the adversary $\mathcal{A}$), $\mathcal{A}$ aims to infer whether there exists a link between the candidate node $u \in V_k$ and $v \in T$. As $\mathcal{A}$ is a participating client in FedGNN, the attack proceeds within the standard training process of FedGNN as follows.

1. Training data: $\mathcal{A}$ trains under its local dataset D_k, where $D_k = (G_k, Y_k)$, $G_k = (V_k, E_k, \mathbf{X}_k)$, and $u \in V_k$.

2. Training updates before label flipping attack: $\mathcal{A}$ launches label flipping attacks at training round t, where t is the round that FedGNN reaches a near-convergent state. Before attacks, $\mathcal{A}$ participants in the general training on D_k with correct label y_u of candidate node u, and obtains a local model update, referred as $\nabla_{k,1}^{t-1}$,

$$\begin{aligned} \nabla_{k,1}^{t-1} &= \mathbf{W}_k^{t-1} - \mathbf{W}^{t-1} \\ &= \nabla_{\mathbf{W}^{t-2}} \ell_k(\mathbf{W}_k^{t-1}, D_k) \end{aligned} \tag{12}$$

where W^{t-1} and W_k^{t-1} denote the global and local parameters at round $(t-1)$, respectively.

3. Training updates under attacking: For candidate node u, during the training round $r \in [t, t+\alpha]$, $\mathcal{A}$ flips u's label y_u to $\neg y_u$, i.e., the original correct label y_u is incorrectly labelled as the target label $\neg y_u$. Then, $\mathcal{A}$ trains on the poisoned data $(V_{k,u}, E_{k,u}, \mathbf{X}_{k,u}, \neg Y_{k,u})$, and get a local update as $\nabla_{k,2}^{r}$,

$$\begin{aligned} \nabla_{k,2}^{r} &= \mathbf{W}_k^{r} - \mathbf{W}^{r} \\ &= \nabla_{\mathbf{W}^{r-1}} \ell_k(\mathbf{W}_k^{r}, (V_{k,u}, E_{k,u}, \mathbf{X}_{k,u}, \neg Y_{k,u})), \end{aligned} \tag{13}$$

where $(V_{k,u}, E_{k,u}, \mathbf{X}_{k,u}, \neg Y_{k,u}) \in D_k$.

4. Training updates for $(D_k/(V_{k,u}, E_{k,u}, \mathbf{X}_{k,u})$: To maintain the accuracy and stability of the global model, while avoiding the impact on other labels, $\mathcal{A}$ only performs label replacement on u. Nodes that are not labelled flipped, i.e., nodes in $D_k/(V_{k,u}, E_{k,u}, \mathbf{X}_{k,u})$, are trained to get corresponding model update $\nabla_{k,3}^{r}$,

$$\begin{aligned} \nabla_{k,3}^{r} &= \mathbf{W}_k^{r} - \mathbf{W}^{r} \\ &= \nabla_{\mathbf{W}^{r-1}} \ell_k(\mathbf{W}_k^{r}, D_k/(V_{k,u}, E_{k,u}, \mathbf{X}_{k,u})), \end{aligned} \tag{14}$$

where $\mathbf{W}^r$ denotes the current global parameters, $\mathbf{W}_k^r$ denotes the local model parameters obtained by training the malicious client on its local dataset.

5. Training updates for D_k: $\mathcal{A}$ sends its local training data gradient to the server to update the global model. The gradient is calculated as following,

$$\nabla_k^r = \nabla \ell_k(W_k^r, D_k) = \nabla_{k,3}^r - \gamma \nabla_{k,1}^{t-1} + \delta \nabla_{k,2}^r \tag{15}$$

Eq. 15 indicates the direction of the gradient update of the local client k model. To amplify the poisoning to the target node u, it requires to make the gradient update direction of the client k opposite to that of the global model. Hence, LFLIA takes a negative sign for $\nabla_{k,1}^{t-1}$ to promote normal classification before attack, a positive sign for $\nabla_{k,2}^{r}$ to flip u to incorrect label in training phase, and a positive sign for $\nabla_{k,3}^{r}$ to maintain the accuracy of other un-poisoned nodes. The γ and δ are used to balance the impact of gradient updates from malicious client k due to the label flipping impact.

Due to fluctuations induced by the size and direction of the gradients on LFLIA, LFLIA may be detected by robust aggregation algorithms (AGRs) (e.g., Median [55], Multi-Krum [56], Trimmed-Mean [55], Fang [46], and FLTrust [57]) of server-side in FedGNNs. To mitigate the detection from AGRs, we aim at minimising the influence on the global model. We consider the norm clipping of the gradient to limit the size and direction of the gradients by applying 2-norm clipping:

$$\nabla_k^r = \frac{\nabla_k^r}{||\nabla_k^r||_2}, \tag{16}$$

4.3 Gradient-ascending link inference attack (GALIA)

Nasr et al. [31] demonstrated that adversaries can infer membership information based on exploiting the stochastic gradient descent algorithm, where the adversaries push towards ascending the gradients of the global model in a white-box setting. If a data record is a member of the training dataset, the gradient computed on this data record will cause an abrupt reduction after the data holder applies its local gradient descent. The adversary can detect this reduction to distinguish members from non-members.

Motivated by the gradient ascending attack on membership inference attacks, we extend this approach to study its effectiveness for LIAs on FedGNNs in a white-box setting, which we call gradient ascending link inference attack (GALIA).

In our GALIA, $\mathcal{A}$ crafts ascending the gradient direction of the loss function to the candidate node u when FedGNN model reaches a near-convergent state. Then, $\mathcal{A}$ uploads the poisoned updates to the server and observes the FedGNN model's behaviours on each target node v. If the behaviours on v alert, we define there exist a link between v and u; otherwise, we infer that there is no link between v and u. Similar to LFLIA, the training process of the malicious client k follows the general training process of FedGNN which is carried as bellow,

1. Training data: $\mathcal{A}$ trains local model under its local dataset D_k, where $D_k := (G_k, Y_k)$, $G_k = (V_k, E_k, \mathbf{X}_k)$, and $u \in V_k$.

2. Training updates before gradient ascending attack: In the standard training process, $\mathcal{A}$ participants in the general training process of the FedGNN model with the candidate node u under *gradient descent*, and obtains a local update $\xi_{k,1}^{t-1}$ of client k at $(t-1)$-th round,

$$\begin{aligned}\xi_{k,1}^{t-1} &= \mathbf{W}_k^{t-1} - \mathbf{W}^{t-1} \\ &= \nabla_{\mathbf{W}^{t-2}} \ell_k(\mathbf{W}_k^{t-1}, D_k)\end{aligned} \tag{17}$$

3. Training updates under attacking: For the target node u, at each training round $r \in [t, t+\alpha]$, The $\mathcal{A}$ performs *gradient ascent* on u's loss to amplify its representation shift. Specifically, using the current global parameters $\mathbf{W}^r$, $\mathcal{A}$ trains its local model but reverses the loss objective on u via gradient ascending on u. $\mathcal{A}$ then obtains corresponding model update $\xi_{k,2}^r$,

$$\begin{aligned}\xi_{k,2}^r &= \mathbf{W}_k^r - \mathbf{W}^r \\ &= \nabla_{\mathbf{W}^{r-1}}\Big[- \ell_k\big(\mathbf{W}^r,\ V_{k,u}, E_{k,u}, \mathbf{X}_{k,u}, Y_{k,u}\big)\Big] \\ &= -\nabla_{\mathbf{W}^{r-1}} \ell_k\big(\mathbf{W}^r,\ V_{k,u}, E_{k,u}, \mathbf{X}_{k,u}, Y_{k,u}\big).\end{aligned} \tag{18}$$

4. Training updates for $D_k/(V_{k,u}, E_{k,u}, \mathbf{X}_{k,u})$: Each node in $D_k/(V_{k,u}, E_{k,u}, \mathbf{X}_{k,u})$ are trained under gradient descent, where the local update for $D_k/(V_{k,u}, E_{k,u}, \mathbf{X}_{k,u})$ at r-th round is $\xi_{k,3}^r$,

$$\begin{aligned}\xi_{k,3}^r &= \mathbf{W}_k^r - \mathbf{W}^r \\ &= \nabla_{\mathbf{W}^{r-1}} \ell_k(\mathbf{W}_k^r, D_k/(V_{k,u}, E_{k,u}, \mathbf{X}_{k,u})),\end{aligned} \tag{19}$$

5. Training updates for D_k: $\mathcal{A}$ sends its composed local gradient to the server for aggregation. The final gradient uploaded at round r is,

$$\xi_k^r = \nabla \ell_k(W_k^r, D_k) = \xi_{k,3}^r - \gamma \xi_{k,1}^{t-1} + \delta \xi_{k,2}^r \tag{20}$$

where γ and δ balance the influence of the pre-attack baseline and the ascent component on u, respectively.

Eq. 20 specifies the update direction of malicious client k. To amplify the poisoning on u, $\mathcal{A}$ uses a negative sign on $\xi_{k,1}^{t-1}$ to follow the standard stabilising descent direction learned before GALIA, while $\mathcal{A}$ uses a positive sign on $\xi_{k,2}^r$ as $\xi_{k,2}^r$ already encodes *ascent* on u via the negative loss. $\xi_{k,3}^r$ has a positive sign to preserve accuracy on un-poisoned nodes. Similar to LFLIA, we also consider the norm clipping of the gradient to limit the size and direction as,

$$\xi_k^r = \frac{\xi_k^r}{||\xi_k^r||_2}, \tag{21}$$

For both LFLIA and GALIA, $\mathcal{A}$ repeats the poisoning-based LIA for α rounds. By monitoring behavioural changes on each target node $v \in T$, $\mathcal{A}$ infers whether an edge (u, v) exists via Eq. 11.

5 Evaluation

5.1 Experimental Setup

We implement the FedGNNs model using the PyTorch framework[1] with PyTorch Geometric[2]. All the experiments are conducted on a workstation running Window 11, equipped with an Intel Core i7-14700K CPU, 64 GB DDR5 RAM, NVIDIA RTX 3090 GPU, CUDA 12.6. We use the Adam optimiser with learning rate of 0.001. To ensure statistical significance and reliability, each experiment is repeated five times, and the results are averaged.

Datasets. All the experiments use public, anonymised datasets[3], including three homogeneous graph datasets (Cora, Citeseer, PubMed) and two heterogeneous graph datasets (DBLP, IMDB) for node classification tasks, which are popularly used as benchmark datasets for graph learning [59]. Our goal is to reveal FedGNN privacy risks to inform defense design, not to enable real-world exploitation. Table 1 shows the statistics of these datasets, respectively. For each

[1] https://pytorch.org/
[2] https://pytorch-geometric.readthedocs.io/en/latest/index.html
[3] https://pytorch-geometric.readthedocs.io/en/latest/cheatsheet/data_cheatsheet.html

Table 1: Statistics of datasets

Dataset	Type	Relation (A–B)	Nodes A	Nodes B	Edges	Features	Classes
Cora [25]	Homo.	Citation	2,708	–	10,556	1,433	7
Citeseer [25]	Homo.	Citation	3,327	–	9,104	3,703	6
Pubmed [25]	Homo.	Citation	19,717	–	88,648	500	3
DBLP [58]	Hetero.	Paper–Author	14,328	4,057	19,645	334	4
		Paper–Conf	14,328	20	14,328		
		Paper–Term	14,327	8,789	88,420		
IMDB [58]	Hetero.	Movie–Actor	4,780	5,841	14,340	1,232	3
		Movie–Director	4,780	2,269	4,780		

dataset, we randomly sample 80% of the data record as the training dataset and the remaining 20% as the test dataset.

GNN architectures. We evaluate our poisoning-based attacks across three GNN architectures: GCN [40], GAT [41], and GraphSAGE [2].

Defences. To account for Byzantine-robust aggregation in federated learning, we adopt five commonly used methods: Median [55], Multi-Krum [56], Trimmed-Mean [55], Fang [46], and FLTrust [57]. To evaluate attack performance under defence methods, we test our attacks under FoolsGold [60], and FL Detector [61]. Both defences are recognised representative defences against poisoning behaviours in FL, and have been used as a benchmarks against poisoning attacks [62, 63].

We consider these defence approaches as they share a common detection principle, which measures the directional similarity (e.g., cosine similarity or angle difference) between client updates to identify anomalous or malicious behaviours. This mechanism is highly relevant to our poisoning-based LIAs because our attacks alter the gradients of the target node in a characteristic direction, which makes these defences the most appropriate for evaluating whether perturbation directions can be detected or filtered.

Overlapping rate. As there are diverse scenarios involving distributed graphs. The relationships between nodes can span across different clients. For example, in citation networks, papers (nodes) can be distributed across different research institutions (clients), where the relationships between nodes represent the citations between papers. Each paper belongs to a specific institution but may be cited by papers from other institutions. In our LFLIA and GALIA experiments, we set the overlapping rate is 0.3.

Number of clients. In our experiments, there are 5 participating clients, including 1 malicious client (i.e., adversary $\mathcal{A}$).

Attack timing. We run each experiments for 500 epochs, and launch the attacks starting with the 250th epoch and end with the 290th epoch. This timing is chosen because the 250th epoch marks the point at which the model has almost converged, making it an ideal moment to evaluate the attack impact during a stable training phase.

Metrics. We use *precision* to quantify the fraction of inferred neighbours that are true neighbours with the correct hop distance (i.e., 1-hop, 2-hop, and 3-hop neighbours). *Recall* is used to measure the fraction of true neighbours whose hop distance are successfully detected.

Baselines. We compare our poisoned-based LIAs with two baselines: LINKTELLER [27] and attack-2 in SLA [25]. Both attacks assume the adversary only has the knowledge of node attributes, and aims to infer linked node pairs. These two attacks are black-box LIAs, the same as our LFLIA scenario where the adversary only has knowledge of its own local dataset and knows the node features from the public online resource. To further explore the privacy vulnerabilities of FedGNNs under stronger adversary's capability, our GALIA performs under a white-box setting to leverage more accessible information to steal link privacy in the FedGNN.

Target set and candidate node selection. In our experiments, we construct a target node set T by randomly sampling nodes from the entire global graph excluding the malicious client's local subgraph. For candidate node u, we select u from the local subgraph of the malicious client. To ensure that the observed output changes of FedGNN on target nodes are induced by the propagated perturbations from poisoned u, u is selected from isolated nodes in the subgraph of the adversary. If isolated nodes do not exist in the adversary's subgraph, we create an isolated node (i.e., auxiliary node) as u to the subgraph.

5.2 Performance of LFLIA and GALIA on FedGNN

Table 2 shows attack performance of LIAs in terms of precision and recall compared with the baselines across five datasets, three GNN architectures, and three aggregation strategies. The results consistently show that both LFLIA and GALIA outperform SLA and LTA across all datasets, models, and aggregation rules. GALIA in particular achieves the highest precision and recall in all cases. On average, GALIA improves precision by more than 15% compared with LTA, and maintains stable recall above 90% in most FedGCN settings.

Impact of GNN architectures. GNN architectures strongly affect the attack success rate. FedGCN is the most vulnerable in most of settings, with GALIA reaching 95.61% precision and 93.28% recall on Citeseer under FedAvg. This may be attributed to GCN's neighbourhood aggregation, which captures richer structural information within its receptive fields, making link inference easier for gradient-based attacks.

FedGraphSage generally yields lower precision and recall across all methods. GALIA obtains 94.31% precision and 98.31% recall on Cora with FedAvg due to fixed neighbour sampling reducing available structural information in FedGraphSage. Nonetheless,

Table 2: Attack performance of LIAs. pr.: precision. re.: recall. Baselines: stealing link inference attack (SLA) [25], LinkTeller (LTA) [27]. Ours: label-flipping link inference attack (LFLIA), gradient-ascending link inference attack (GALIA). "_": best results, *bold*: second best .

AGRs		FedAvg						Fang						FLTrust					
Dataset	Method	GAT		GCN		GraphSage		GAT		GCN		GraphSage		GAT		GCN		GraphSage	
		pr	re	pr	re	pr	re	pr	re	pr	re	pr	re	pr	re	pr	re	pr	re
Cora	SLA	0.3649	0.4576	0.3784	0.4746	0.3716	0.4661	0.2230	0.2797	0.2365	0.2966	0.2297	0.2881	0.2027	0.2542	0.2162	0.2712	0.2095	0.2627
	LTA	0.7538	0.8305	0.7692	0.8475	0.7615	0.8390	0.5923	0.6525	0.6077	0.6695	0.6000	0.6610	0.5692	0.6271	0.5846	0.6441	0.5769	0.6356
	LFLIA	**0.9262**	**0.9576**	**0.9350**	**0.9746**	**0.9344**	**0.9661**	**0.7541**	**0.7797**	**0.7705**	**0.7966**	**0.7623**	**0.7881**	**0.7295**	**0.7542**	**0.7459**	**0.7712**	**0.7377**	**0.7627**
	GALIA	**0.9426**	**0.9746**	**0.9512**	**0.9915**	**0.9431**	**0.9831**	**0.7705**	**0.7966**	**0.7869**	**0.8136**	**0.7787**	**0.8051**	**0.7459**	**0.7712**	**0.7623**	**0.7881**	**0.7541**	**0.7797**
Citeseer	SLA	0.3516	0.4639	0.3672	0.4845	0.3594	0.4742	0.1563	0.2062	0.1641	0.2165	0.1563	0.2062	0.1385	0.1856	0.1462	0.1959	0.1385	0.1856
	LTA	0.7143	0.7732	0.7333	0.7938	0.7238	0.7835	0.5673	0.6082	0.5865	0.6289	0.5769	0.6186	0.5534	0.5876	0.5728	0.6082	0.5631	0.5979
	LFLIA	**0.8800**	**0.9072**	**0.8911**	**0.9278**	**0.8900**	**0.9175**	**0.7172**	**0.7320**	**0.7374**	**0.7526**	**0.7273**	**0.7423**	**0.7071**	**0.7216**	**0.7273**	**0.7423**	**0.7172**	**0.7320**
	GALIA	**0.9000**	**0.9278**	**0.9109**	**0.9485**	**0.9010**	**0.9381**	**0.7475**	**0.7629**	**0.7677**	**0.7835**	**0.7576**	**0.7732**	**0.7700**	**0.7938**	**0.7900**	**0.8144**	**0.7800**	**0.8041**
Pubmed	SLA	0.3169	0.3947	0.3662	0.4561	0.3404	0.4211	0.1761	0.2193	0.1901	0.2368	0.1831	0.2281	0.1620	0.2018	0.1761	0.2193	0.1690	0.2105
	LTA	0.7698	0.8509	0.7857	0.8684	0.7778	0.8596	0.6032	0.6667	0.6190	0.6842	0.6111	0.6754	0.5714	0.6316	0.5873	0.6491	0.5794	0.6404
	LFLIA	**0.9068**	**0.9386**	**0.9160**	**0.9561**	**0.9153**	**0.9474**	**0.6525**	**0.6754**	**0.6695**	**0.6930**	**0.6610**	**0.6842**	**0.6186**	**0.6404**	**0.6356**	**0.6579**	**0.6271**	**0.6491**
	GALIA	**0.9237**	**0.9561**	**0.9328**	**0.9737**	**0.9244**	**0.9649**	**0.7217**	**0.7281**	**0.7391**	**0.7456**	**0.7304**	**0.7368**	**0.6870**	**0.6930**	**0.7043**	**0.7105**	**0.6957**	**0.7018**
DBLP	SLA	0.2984	0.3742	0.3121	0.3914	0.3052	0.3828	0.2027	0.2542	0.2162	0.2712	0.2095	0.2627	0.1827	0.2291	0.1962	0.2460	0.1895	0.2376
	LTA	0.7231	0.7965	0.7385	0.8135	0.7308	0.8050	0.5692	0.6271	0.5846	0.6441	0.5769	0.6356	0.5385	0.5932	0.5538	0.6102	0.5462	0.6017
	LFLIA	**0.8525**	**0.8814**	**0.8689**	**0.8983**	**0.8607**	**0.8898**	**0.7295**	**0.7542**	**0.7459**	**0.7712**	**0.7705**	**0.7966**	**0.7077**	**0.7314**	**0.7231**	**0.7475**	**0.7541**	**0.7797**
	GALIA	**0.8852**	**0.9153**	**0.9016**	**0.9322**	**0.8934**	**0.9237**	**0.7623**	**0.7881**	**0.7787**	**0.8051**	**0.7377**	**0.7627**	**0.7459**	**0.7712**	**0.7623**	**0.7881**	**0.7154**	**0.7397**
IMDB	SLA	0.2869	0.3597	0.3006	0.3769	0.2937	0.3683	0.1912	0.2397	0.2047	0.2566	0.1979	0.2481	0.1712	0.2146	0.1847	0.2315	0.1780	0.2231
	LTA	0.7038	0.7750	0.7192	0.7920	0.7115	0.7835	0.5500	0.6059	0.5654	0.6229	0.5577	0.6144	0.5192	0.5720	0.5346	0.5889	0.5269	0.5805
	LFLIA	**0.8328**	**0.8609**	**0.8492**	**0.8778**	**0.8410**	**0.8693**	**0.7098**	**0.7339**	**0.7590**	**0.7846**	**0.7180**	**0.7422**	**0.6885**	**0.7117**	**0.7426**	**0.7677**	**0.6962**	**0.7198**
	GALIA	**0.8656**	**0.8948**	**0.8819**	**0.9117**	**0.8738**	**0.9032**	**0.7426**	**0.7677**	**0.7262**	**0.7507**	**0.7508**	**0.7761**	**0.7262**	**0.7507**	**0.7038**	**0.7276**	**0.7180**	**0.7422**

LFLIA and GALIA still substantially outperform the baselines under FedGraphSage, highlighting their adaptability.

FedGAT, with GAT attention mechanism, shows a resilience to inference attacks by focusing on relevant neighbours and reducing noise from non-neighbour nodes. GALIA leverages gradient distance estimation to bypass this selective weighting effectively, resulting in good precision and recall even within the FedGAT architecture.

Impact of FL aggregations. Without aggregation rules (AGRs), GALIA achieves an average recall of approximately 84%. In contrast, when AGRs such as Fang or FLTrust are applied, the attack success declines notably. These AGRs enhance robustness by selectively aggregating benign client updates or employing trusted reference gradients, thereby diminishing the efficacy of link inference attacks.

In particularly, on the Pubmed dataset with a GraphSage model under FLTrust, GALIA's precision drops to 69.57%, whereas LTA and SLA attain only 57.94% and 16.90%, respectively. This reduction may stem from Pubmed's larger and denser graph structure, which dilutes the propagation of adversarial perturbations. Therefore, while robust AGRs decrease the attack success of our LIAs, they do not completely prevent link inference. This vulnerability indicates that structural perturbations and gradient ascent strategies retain efficacy against AGRs, particularly in graphs with complex or dense topology (see Appendices A.1 for more results in AGRs of Medium, Trimmed-Mean and Multi-Krum).

Impact of dataset characteristics. Our attacks exhibit consistent behaviour across both homogeneous and heterogeneous graphs with similar performance trends. Although homogeneous graphs show slightly higher vulnerability, the core attack mechanisms remain effective in both settings. As shown in Table 2, homogeneous datasets (Cora, Citeseer, PubMed) generally yield higher attack precision and recall than heterogeneous datasets (DBLP, IMDB). For example, GALIA reaches peak precision of 95.12% on Cora under FedAvg, compared to 88.52% on DBLP and 86.56% on IMDB.

This 6%–8% performance gap can be attributed to the increased structural complexity of heterogeneous graphs, where diverse node and edge types introduce additional relational noise that partially masks the injected perturbation signals. Nevertheless, the preservation of effective attack performance across different graph structures confirms that our methods leverage universal structural vulnerabilities in FedGNNs.

Impact of defence methods. Our analysis reveals distinct defence effectiveness against link inference attacks in FedGNN, shown as in Fig. 5. FLDetector emerges as the more potent defence, achieving significant performance reductions with approximately 23%-35%, with stronger impact on GALIA with around 29%-35% reduction than LFLIA with 23%-29% reduction.

The reason behind this may be that FLDetector's enhanced sensitivity to gradient manipulation information, while GALIA's explicit gradient ascent creates more detectable anomalies, resulting in performance drops from 0.9426 to 0.6928 on Cora and from 0.8656 to 0.6213 on IMDB. In contrast, FoolsGold demonstrates moderate defence capability with consistent 9%-12% performance reductions across all datasets, reflecting its gradient similarity-based detection approach.

The defence effectiveness exhibits dataset-dependent characteristics, with heterogeneous graphs (DBLP, IMDB) showing greater vulnerability reduction under FLDetector with 32.9%-33.5% for GALIA, compared to homogeneous graphs. However, even the strongest

defence only partially mitigates the attacks, as precision values remain above 0.62 across all scenarios.

This underscores a critical limitation: while sophisticated defences can significantly degrade attack performance, they cannot completely eliminate the threat, highlighting the need for multi-layered protection strategies in practical FedGNN deployments. The empirical evidence establishes that current poisoning-aware defences provide substantial but incomplete protection against link inference attacks in federated graph learning.

Global model performance. To evaluate whether the poisoning-based LIAs influence on the downstream learning task, we analyse the global model's training accuracy before, during, and after the poisoning interval. As shown in Fig. 5, during the attack period [250, 290], the training accuracy of global FedGNN model under GALIA without defences drops from approximately 95% to approximately 71%, while with FoolsGold reduces accuracy to approximately 77% and with FLDetector reduces accuracy approximately 82.5%.

It can be seen that GALIA can disrupts the downstream learning task even under defensive measures. However, after stopping attack and continued training, the training accuracy of global model on the main task re-converges to around 83%-85%. Therefore, it is not easy to clear all these the propagation of adversarial influence through message passing.

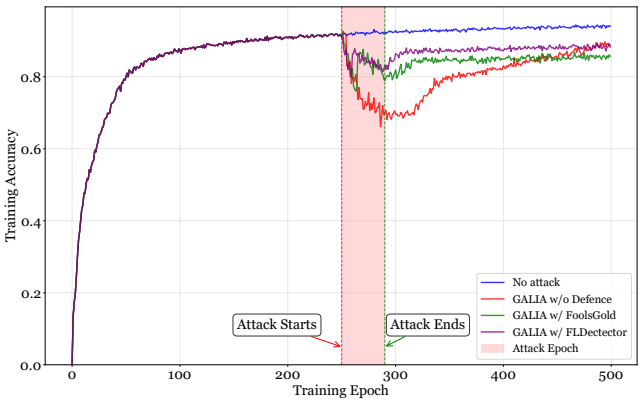


Figure 5: Precision under overlapping rate analysis of GALIA performances on Cora dataset with FedGCN under FedAvg.

6 Ablation Study

We conduct ablation studies on the impact of overlapping rate, number of adversaries, and attack launch timing.

6.1 Overlapping rate across clients

We evaluate LIAs on FedGNNs with varying overlapping rate ranging from 0.2 to 0.4. Fig. 6 shows the GALIA attack performance on Cora dataset with FedGCN under varying overlapping rate. It can be seen that node overlap rate is a critical factor influencing the success rate of LIAs. Higher overlapping rate exposes more cross-client edges and shared neighbourhoods, which amplifies message-passing information and the structural information available for inference.

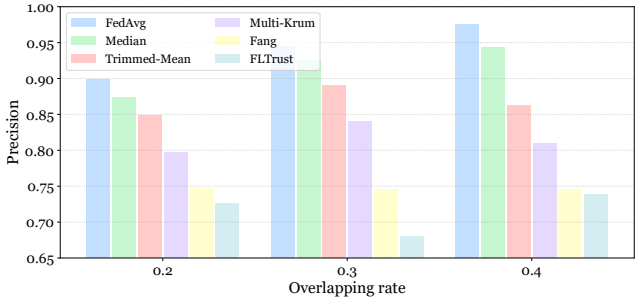


Figure 6: Precision under overlapping rate analysis of GALIA performances on Cora dataset with FedGCN.

With varying the overlap rate from 0.2 to 0.4, all attack methods, especially GALIA, show significantly improved attack success rates, with GALIA achieving up to 97.5% against FedGCN model under FedAvg. The behind reasons might fundamentally stem from higher node overlap providing adversaries with richer cross-client information flow, which enables more accurate inference of the global graph structure. Specifically, overlapping nodes can be the information bridges, which allow adversaries to piece together local graph topological knowledge from different clients and reconstruct a more complete graph structure.

Additionally, more overlapping nodes strengthens feature and representation correlations in model updates, which adversaries can exploit by analysing gradients or parameter shifts. GALIA performs better than LFLAI under FedAvg. On Cora dataset under FedGCN, the precision is 8.3% for GALIA, precision is 6.5% for LFLIA from overlapping rate ranging from 0.2 to 0.4. GALIA performs better as it can utilise these gradient information and leverage graph attention mechanisms to capture higher-order relationships between nodes.

Meanwhile, robust aggregators also show varied resilience under increasing overlap. Under FLTrust on Cora with FedGCN, GALIA's precision rises from 72.6% to 73.9%, indicating that FLTrust mitigates information leakage by filtering anomalous gradients in some extent. However, the persistent vulnerability highlights that even robust AGRs cannot fully defend attacks under high overlapping rate. Similar trends are observed on Citeseer and Pubmed datasets, where overlap consistently amplifies attack efficacy (see Appendix A.4.1).

6.2 Number of adversaries

Fig. 7 represents the attack performance of GALIA on FedGCN with varying the number of adversaries (i.e., malicious clients) with a overlapping rate of 0.3 and 15 clients. We vary the number of malicious clients from 1 to 10. It can be seen that the attack precision of GALIA improves as the number of malicious clients increases from 1 to 10 across three datasets and aggregation strategies. Under FedAvg, the attack precision exhibits a positive trend. On the Cora dataset, the precision increases from approximately 0.91 with a single malicious client to above 0.96 when 10 malicious clients are participating in training process. A similar trend is observed on Citeseer and PubMed, where the attack precision improves by approximately 5%–7% as the number of adversaries increases.

Table 3: Attack performance under FedAvg with poisoning-aware defences. pr.: precision. re.: recall.

Defences		No defence						FoolsGold						FLDetector					
Dataset	Method	GAT		GCN		GraphSage		GAT		GCN		GraphSage		GAT		GCN		GraphSage	
		pr	re	pr	re	pr	re	pr	re	pr	re	pr	re	pr	re	pr	re	pr	re
Cora	LFLIA	0.9262	0.9576	0.9350	0.9746	0.9344	0.9661	0.8402	0.8687	0.8491	0.8841	0.8485	0.8768	0.7337	0.7584	0.7408	0.7720	0.7402	0.7649
	GALIA	0.9426	0.9746	0.9512	0.9915	0.9431	0.9831	0.8551	0.8842	0.8640	0.8996	0.8633	0.8923	0.6928	0.7158	0.7002	0.7289	0.6993	0.7224
Citeseer	LFLIA	0.8800	0.9072	0.8911	0.9278	0.8900	0.9175	0.7964	0.8220	0.8063	0.8404	0.8054	0.8310	0.6836	0.7054	0.6927	0.7215	0.6917	0.7132
	GALIA	0.9000	0.9278	0.9109	0.9485	0.9010	0.9381	0.8150	0.8409	0.8251	0.8598	0.8162	0.8506	0.6507	0.6718	0.6598	0.6873	0.6587	0.6803
Pubmed	LFLIA	0.9068	0.9386	0.9160	0.9561	0.9153	0.9474	0.8225	0.8503	0.8309	0.8672	0.8303	0.8594	0.7155	0.7396	0.7245	0.7553	0.7239	0.7485
	GALIA	0.9237	0.9561	0.9328	0.9737	0.9244	0.9649	0.8378	0.8660	0.8464	0.8831	0.8390	0.8752	0.6805	0.7034	0.6890	0.7177	0.6876	0.7112
DBLP	LFLIA	0.8525	0.8814	0.8689	0.8983	0.8607	0.8898	0.7732	0.7997	0.7876	0.8145	0.7803	0.8069	0.6505	0.6727	0.6631	0.6854	0.6567	0.6789
	GALIA	0.8852	0.9153	0.9016	0.9322	0.8934	0.9237	0.8028	0.8303	0.8175	0.8454	0.8102	0.8378	0.6365	0.6580	0.6481	0.6705	0.6418	0.6635
IMDB	LFLIA	0.8328	0.8609	0.8492	0.8778	0.8410	0.8693	0.7556	0.7818	0.7704	0.7968	0.7630	0.7892	0.6287	0.6500	0.6410	0.6630	0.6347	0.6563
	GALIA	0.8656	0.8948	0.8819	0.9117	0.8738	0.9032	0.7850	0.8119	0.7995	0.8270	0.7921	0.8195	0.6213	0.6423	0.6330	0.6550	0.6269	0.6481

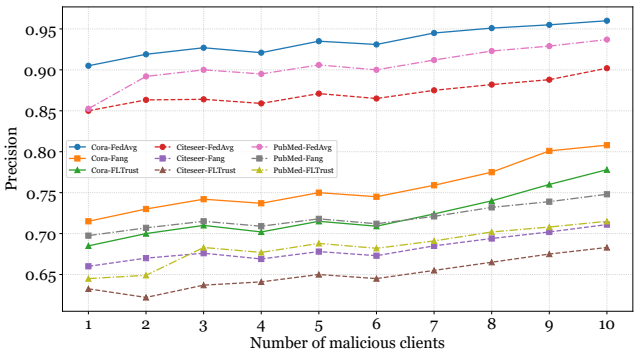


Figure 7: Precision under number of adversaries analysis of GALIA performances with FedGCN.

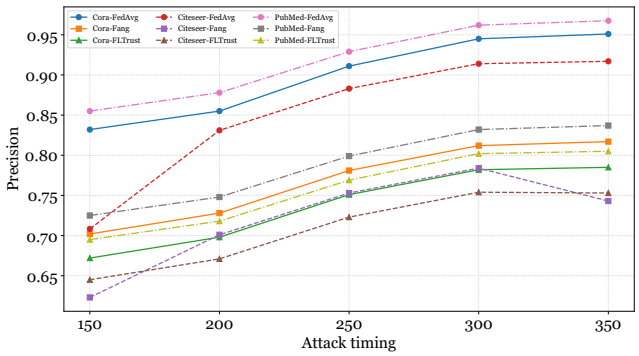


Figure 8: Precision under attack timing analysis of GALIA performances with FedGCN.

Under robust aggregators such as Fang and FLTrust, the attack precision across three datasets is reduced compared to FedAvg. Nevertheless, a slight upward trend with respect to the number of malicious clients can still be observed. In particular, a mild non-monotonic behaviour occurs when the number of malicious clients varies between 4 and 6, where the precision slightly decreases. Additional results under other robust aggregators (i.e., Medium,Trimmed-Mean and Multi-Krum) are provided in Appendix A.4.2.

We analyse that might be from some underlying reasons: 1) Multiple adversaries enable enhanced information aggregation through their distributed observations, creating a collaborative intelligence that reconstructs link information with greater precision; 2) The presence of more malicious clients amplifies gradient information by capturing more model updates; and 3) As more adversaries can obtain increased influence over aggregated updates while simultaneously reducing the effectiveness of conventional defence mechanisms.

6.3 Attack timing

We also investigate the impact of attack launch timing of GALIA. With an overlapping rate of 0.3. Fig. 8 illustrates the impact of attack timing on the performance of GALIA under the FedGCN setting, where the attack timing ratio varies from 150 to 350 epochs. It can be seen that, across all three datasets, the attack precision indicates that poisoning-based LIAs are less effective (e.g., precision is below 63% on Cora with Fang) when the adversary launches GALIA at the early training stages (e.g., 150-200 epochs). This is because the global model has not yet converged, and subsequent clean updates dilute the malicious signal.

When the adversary launches GALIA during FedGNN training epochs 250–300, the attack precision reaches its peak, with improvements of 13.6% on Cora, 11.6% on Citeseer, and 14.4% on PubMed compared to launching at the 150-th epoch. At this stage, node representations have stabilised, allowing poisoned gradients to accumulate and diffuse through message passing while sufficient training rounds remain.

However, when GALIA is launched after the 300-th epoch, the poisoning-based LIA yields no further benefits and performs worse than attacks launched between epochs 250 and 300. This is because the model has converged again. That is, gradients are smaller, learning rates decay, and robust aggregators can filter anomalous updates in some extent due to limited optimisation budget for malicious perturbations to propagate. In terms of aggregators, the adversary consistently achieves the high leakage (approximately 92% at the 350-th epoch) under FedAvg, while FLTrust and Fang can suppress the attacks in some extent (e.g., approximately decreasing from

around 77% at 300-th epoch to approximately 74% at 350-th epoch on Citeseer with Fang).

These vulnerabilities stem from fundamental learning dynamics in GCNs. Early in training, models focus primarily on learning node features rather than structural relationships. As training progresses, they shift toward refining their representations of graph topology, gradually accumulating and stabilising structural knowledge. This progressive encoding of graph structure explains why later attacks are more successful. The model has already captured richer structural dependencies, gradient updates are more stable and structure-focused, and attackers can exploit cleaner signals. Moreover, increasing model confidence inadvertently exposes more definitive information about actual graph connections.

7 Ethical Considerations

This work studies privacy inference risks in federated graph neural networks through controlled poisoning-based attacks. All experiments are conducted on publicly available benchmark datasets, without involving real users or sensitive personal information.

8 Discussion

In this work, we bridged a gap in the study of link inference attacks on FedGNNs by proposing two poisoning attack strategies, LFLIA and GALIA. Both exploit the message-passing mechanism inherent in GNNs and the aggregation process in federated learning to amplify privacy leakage. Our empirical results demonstrate that GALIA consistently outperforms baseline attacks (SLA and LTA) across different federated aggregators.

A natural question then arises: how can such attacks be defended against? Prior studies on privacy protection in GNNs and FL have introduced a range of defences, such as differential privacy (DP) and anomaly-based detection. For example, Edge-DP adds Laplacian noise to links, while other DP approaches inject noise into GNN aggregation functions [27, 33, 64–66]. LoDen [33] detects suspicious updates locally using statistical anomaly detection, and FLDetector [61] filters out clients whose model updates deviate from predicted behaviour. Similarly, PoisonedFL [67] introduces consistency constraints across malicious clients to counter poisoning strategies.

However, our study highlights why these existing approaches may be insufficient in defending against our proposed LIAs. The core difficulty lies in the message-passing mechanism of GNNs, which allows the influence of poisoned samples to propagate to their neighbours and accumulate over training iterations. Even if a method like LoDen successfully detects and discards a poisoned sample, the malicious signal may already have been diffused through the neighbourhood during earlier rounds. Once this toxicity has been embedded into the representation space, later filtering of the original sample cannot fully reverse its effect.

This propagation characteristic fundamentally distinguishes our attacks from conventional poisoning scenarios. While prior defences were designed around detecting or constraining malicious updates at the client or sample level, they do not directly address the neighbourhood-level diffusion introduced by message passing within intra-client and cross-client edges in FedGNNs. As a result, our attacks remain effective to some extent in the presence of existing defences.

In addition, we discussed how LIA-inferred structural relations to privacy of graph reconstruction. Based on the structural relations inferred by LALIA and GALIA, incorporating LIA-inferred edges into a breadth-first expansion strategy significantly improves reconstruction quality compared with feature-only baselines by K-nearest neighbours (see Appendix A.2). The results shown that structural relations inferred by LIAs also amplify broader structural leakage risks at graph reconstructions.

These findings point to an urgent need for new defence mechanisms specifically tailored to FedGNN link inference attacks. In particular, future work is significant on investigating approaches that account for the temporal and structural spread of poisoned messages across the graph, rather than focusing solely on local detection. Designing defences, such as potentially through structure-aware anomaly detection, adaptive aggregation, or hybrid DP mechanisms, represents a promising and an important direction for mitigating the privacy risks exposed by our work.

9 Related Work

9.1 Poisoning inference attacks against FL

In traditional machine learning (ML), the integrity of training data can be compromised to breach user privacy through a range of privacy inference attacks [29, 68, 69]. Perturbation-based attacks (e.g., data poisoning attacks and model poisoning attacks) [29, 34, 70–72] have been widely studied as effective privacy leakage attacks by exploiting vulnerabilities in the training process.

In perturbation-based attacks, adversaries directly manipulate training data or model updates. Data poisoning attacks inject carefully crafted samples into the training set to bias the model towards certain decision boundaries, thereby facilitating membership inference (MIA) or property inference (PIA) [37]. Model poisoning attacks tamper with model updates to bias the global optimisation process towards privacy leakage. For example, Mahloujifar et al. [29] demonstrated that injecting small perturbations into a property-dependent dataset can amplify behavioural differences between models trained with and without that property, enabling property inference.

Recent studies show that such poisoning-based inference attacks also pose significant risks in federated learning (FL). In FL, adversaries may act as malicious clients by injecting poisoned data [34] or manipulating model updates [31] to extract sensitive information such as membership, attributes, or properties of participants' data. Both surrogate-based and perturbation-based strategies remain effective in this setting. For instance, Melis et al. [37] demonstrated that an honest-but-curious server can infer membership by monitoring client updates, while property inference can be achieved by observing poisoned updates. Fu et al. [73] showed that malicious participants can infer private label information by exploiting gradient signals and manipulating local optimisers. Bai et al. [74] proposed ProVFL, a property inference framework in vertical FL, where adversaries exploit correlations between the L-norm distribution of intermediate embeddings or gradients and the global fraction

of a target property, thereby amplifying the distinguishability of model outputs with and without that property.

9.2 Link inference attacks against GNNs

Motivated by increasing concerns over structural privacy in graph neural networks (GNNs), recent studies have examined link inference attacks (LIAs) [25, 27, 28, 35, 75, 76] that aim to determine whether an edge exists between two nodes. Since edges encode sensitive relationships (e.g., friendships, transactions, or interactions), successful LIAs allow adversaries to recover private relational information (e.g., user profiles or preferences) and exploit it for malicious purposes such as targeted advertising.

The earliest study of LIAs on GNNs is the stealing link attack (SLA) [25], which provides the first systematic exploration of link leakage in centralised GNNs. SLA considers a range of adversarial knowledge assumptions and proposes several attack strategies, including: (i) querying node-pair link prediction scores, (ii) leveraging node embedding similarity, (iii) training shadow GNNs on auxiliary data, and (iv) learning ML classifiers from known link patterns. These methods demonstrate that even black-box GNN outputs contain sufficient relational signals to reveal hidden edges. Subsequent work by Wu et al. [75] extends SLA to inductive GNNs.

Li et al. [27] proposed LinkTeller, which takes a different perspective. LinkTeller considers a vertically partitioned setting where one party holds node features and another holds edges. By perturbing node features and analysing how messages propagate through the GNN, LinkTeller infers private edges via the perturbed influence analysis.

Recently, several poisoning-based and auxiliary-node-injection LIAs have been developed. VertexSerum [28] applies imperceptible perturbations via projected gradient descent to poison a subgraph during training, thereby amplifying the separability of linked and unlinked node pairs. GNNBleed [35] and Devil in Disguise [76] introduce auxiliary nodes to connect with victim nodes and measure posterior shifts to infer hidden edges without querying victim posteriors directly. LinkThief [36] adopts a hybrid scoring strategy that integrates embedding similarity and structural priors for probabilistic edge reconstruction.

Although existing LIAs have demonstrated effectiveness on GNNs, they often rely on strong and unrealistic assumptions. SLA requires multiple queries and access to graph embeddings. LinkTeller assumes complete control over node features. VertexSerum presumes adversarial control over a partial graph. GNNBleed and Devil in Disguise require adversaries to inject auxiliary nodes and manipulate their features. LinkThief depends heavily on structural priors, which limits its applicability in sparse or heterogeneous graphs. These requirements restrict the practicality of existing LIAs in real-world scenarios where adversaries typically have limited access. A comprehensive and comparative summary of these LIAs can be found in Appendix A.3.

Despite growing work on privacy inference in ML, FL, and GNNs, LIAs in federated GNNs (FedGNNs) remain under explored. As discussed in Section 1, FedGNNs mitigate the centralisation risks of standard GNNs [42, 77], but they also introduce new privacy vulnerabilities. This is because graph-structured data inherently spans across clients, and message passing propagates information through intra-client and cross-client edges [78, 79]. These characteristics introduce new structural leakage pathways and create opportunities for malicious clients to exploit local training dynamics or federated aggregation (e.g., FedAvg) to infer private edges. Existing LIAs do not account for these distributed training behaviours, nor do they explore poisoning-based attacks tailored to federated optimisation.

10 Conclusion

In this work, we investigated the overlooked problem of link privacy leakage in federated graph neural networks (FedGNNs). Unlike traditional federated learning, FedGNNs are vulnerable since message passing mechanism allows perturbations to propagate across both intra-client and cross-client edges for graph data, thereby embedding hidden structural dependencies in aggregated updates and node representations.

To exploit this property, we proposed two poisoning-based link inference attack strategies: label flipping link inference attack (LFLIA) and gradient ascending link inference attack (GALIA). Both strategies leverage crafted perturbations on the candidate node to induce measurable shifts in FedGNN's outputs on target nodes to reveal link privacy between the candidate node and target nodes.

Experimental results across multiple datasets and aggregation settings demonstrated the effectiveness of our attacks. For example, GALIA archived 95.61% attack precision on Citeseer under FedAvg, significantly outperforming SLA [25] (48.45%) and LTA [27] (79.38%). We further analysed the influence of overlapping rates, the number of adversaries, and attack launch timing on LIA performance.

We also evaluated the attack performance of proposed poisoning-based LIAs under existing defence strategies and discussed the limitations of existing defence mechanisms against LIAs on FedGNNs. In particular, we found that although some detection methods could identify poisoned samples, they failed to prevent the propagation of adversarial influence through message passing. We also discussed and demonstrated that the inferred structural relations by LIAs could amplify the privacy of graph reconstruction.

Acknowledgments

The authors used ChatGPT to revise the text, improve flow and correct any typos, grammatical errors, and awkward phrasing.

References

[1] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.
[2] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.
[3] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI open*, vol. 1, pp. 57–81, 2020.
[4] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph neural networks in recommender systems: a survey," *ACM Computing Surveys*, vol. 55, no. 5, pp. 1–37, 2022.
[5] G. Panagopoulos, G. Nikolentzos, and M. Vazirgiannis, "Transfer graph neural networks for pandemic forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 6, 2021, pp. 4838–4845.
[6] S. Zheng, Z. Zhu, Z. Liu, Z. Guo, Y. Liu, Y. Yang, and Y. Zhao, "Multi-modal graph learning for disease prediction," *IEEE Transactions on Medical Imaging*, vol. 41, no. 9, pp. 2207–2216, 2022.
[7] S. Yu, F. Xia, S. Li, M. Hou, and Q. Z. Sheng, "Spatio-temporal graph learning for epidemic prediction," *ACM Transactions on Intelligent Systems and Technology*,

vol. 14, no. 2, pp. 1–25, 2023.
[8] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *The world wide web conference*, 2019, pp. 417–426.
[9] Y. Zhang, H. Gao, J. Pei, and H. Huang, "Robust self-supervised structural graph neural network for social network prediction," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 1352–1361.
[10] A. Act, "Health insurance portability and accountability act of 1996," *Public law*, vol. 104, p. 191, 1996.
[11] U. D. of Health and H. Services. (2024) The hipaa privacy rule. [Online]. Available: https://www.hhs.gov/hipaa/for-professionals/privacy/index.html
[12] E. Union. (2016) General data protection regulation. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679
[13] C. Wu, F. Wu, Y. Cao, Y. Huang, and X. Xie, "Fedgnn: Federated graph neural network for privacy-preserving recommendation," *arXiv preprint arXiv:2102.04925*, 2021.
[14] C. Wu, F. Wu, L. Lyu, T. Qi, Y. Huang, and X. Xie, "A federated graph neural network framework for privacy-preserving personalization," *Nature Communications*, vol. 13, no. 1, p. 3091, 2022.
[15] T. Zhang, C. Mai, Y. Chang, C. Chen, L. Shu, and Z. Zheng, "Fedego: privacy-preserving personalized federated graph learning with ego-graphs," *ACM Transactions on Knowledge Discovery from Data*, vol. 18, no. 2, pp. 1–27, 2023.
[16] B. Yan, Y. Cao, H. Wang, W. Yang, J. Du, and C. Shi, "Federated heterogeneous graph neural network for privacy-preserving recommendation," in *Proceedings of the ACM Web Conference 2024*, 2024, pp. 3919–3929.
[17] H. Yuan, J. Xu, C. Wang, Z. Yang, C. Wang, K. Yin, and Y. Yang, "Unveiling privacy vulnerabilities: Investigating the role of structure in graph data," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 4059–4070.
[18] T. Zhao, H. Hu, and L. Cheng, "Unveiling the role of message passing in dual-privacy preservation on gnns," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2023, pp. 3474–3483.
[19] Y. Yao, W. Jin, S. Ravi, and C. Joe-Wong, "Fedgcn: Convergence-communication tradeoffs in federated training of graph convolutional networks," *Advances in neural information processing systems*, vol. 36, pp. 79 748–79 760, 2023.
[20] Y. Yao, Y. Li, X. Fan, J. Li, K. Liu, W. Jin, Y. Yang, S. Ravi, P. S. Yu, and C. Joe-Wong, "Fedgraph: A research library and benchmark for federated graph learning," *arXiv preprint arXiv:2410.06340*, 2024.
[21] Z. Guo, D. Yao, Q. Yang, and H. Liu, "Hifgl: A hierarchical framework for cross-silo cross-device federated graph learning," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 968–979.
[22] V. Duddu, A. Boutet, and V. Shejwalkar, "Quantifying privacy leakage in graph embedding," in *MobiQuitous 2020-17th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, 2020, pp. 76–85.
[23] H. Zhan, K. Zhang, K. Lu, and V. S. Sheng, "Measuring the privacy leakage via graph reconstruction attacks on simplicial neural networks (student abstract)," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 13, 2023, pp. 16 380–16 381.
[24] Y. Zhuang, C. Shi, M. Zhang, J. Chen, L. Lyu, P. Zhou, and L. Sun, "Unveiling the secrets without data: Can graph neural networks be exploited through {Data-Free} model extraction attacks?" in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 5251–5268.
[25] X. He, J. Jia, M. Backes, N. Z. Gong, and Y. Zhang, "Stealing links from graph neural networks," in *30th USENIX security symposium (USENIX security 21)*, 2021, pp. 2669–2686.
[26] S. Zhang, H. Yin, T. Chen, Z. Huang, L. Cui, and X. Zhang, "Graph embedding for recommendation against attribute inference attacks," in *Proceedings of the web conference 2021*, 2021, pp. 3002–3014.
[27] F. Wu, Y. Long, C. Zhang, and B. Li, "Linkteller: Recovering private edges from graph neural networks via influence analysis," in *2022 ieee symposium on security and privacy (sp)*. IEEE, 2022, pp. 2005–2024.
[28] R. Ding, S. Duan, X. Xu, and Y. Fei, "Vertexserum: Poisoning graph neural networks for link inference," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4532–4541.
[29] S. Mahloujifar, E. Ghosh, and M. Chase, "Property inference from poisoning," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022, pp. 1120–1137.
[30] Y. Chen, C. Shen, Y. Shen, C. Wang, and Y. Zhang, "Amplifying membership exposure via data poisoning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 29 830–29 844, 2022.
[31] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *2019 IEEE symposium on security and privacy (SP)*. IEEE, 2019, pp. 739–753.
[32] Y. Zhang, G. Bai, M. A. P. Chamikara, M. Ma, L. Shen, J. Wang, S. Nepal, M. Xue, L. Wang, and J. Liu, "Agrevader: Poisoning membership inference against byzantine-robust federated learning," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 2371–2382.
[33] M. Ma, Y. Zhang, P. C. M. Arachchige, L. Y. Zhang, M. B. Chhetri, and G. Bai, "Loden: Making every client in federated learning a defender against the poisoning membership inference attacks," in *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security*, 2023, pp. 122–135.
[34] Z. Wang, Y. Huang, M. Song, L. Wu, F. Xue, and K. Ren, "Poisoning-assisted property inference attack against federated learning," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 4, pp. 3328–3340, 2022.
[35] Z. Song, E. Kabir, and S. Mehnaz, "Gnnbleed: Inference attacks to unveil private edges in graphs with realistic access to gnn models," *arXiv preprint arXiv:2311.16139*, 2023.
[36] Y. Zhang, S. Meng, C. Chen, M. Peng, H. Gu, and X. Huang, "Linkthief: Combining generalized structure knowledge with node similarity for link stealing attack against gnn," in *Proceedings of the 32nd ACM International Conference on Multimedia*, 2024, pp. 4947–4956.
[37] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *2019 IEEE symposium on security and privacy (SP)*. IEEE, 2019, pp. 691–706.
[38] X. Yin, Y. Zhu, and J. Hu, "A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, pp. 1–36, 2021.
[39] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.
[40] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.
[41] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.
[42] B. Wang, A. Li, M. Pang, H. Li, and Y. Chen, "Graphfl: A federated learning framework for semi-supervised node classification on graphs," in *2022 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2022, pp. 498–507.
[43] C. He, K. Balasubramanian, E. Ceyani, C. Yang, H. Xie, L. Sun, L. He, L. Yang, S. Y. Philip, Y. Rong *et al.*, "Fedgraphnn: A federated learning benchmark system for graph neural networks," in *ICLR 2021 Workshop on Distributed and Private Machine Learning (DPML)*, 2021.
[44] C. He, E. Ceyani, K. Balasubramanian, M. Annavaram, and S. Avestimehr, "Spreadgnn: Serverless multi-task federated learning for graph neural networks," *arXiv preprint arXiv:2106.02743*, 2021.
[45] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" *arXiv preprint arXiv:1810.00826*, 2018.
[46] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to {Byzantine-Robust} federated learning," in *29th USENIX security symposium (USENIX Security 20)*, 2020, pp. 1605–1622.
[47] T. Van Erven and P. Harremos, "Rényi divergence and kullback-leibler divergence," *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 3797–3820, 2014.
[48] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.
[49] K. Oono and T. Suzuki, "Graph neural networks exponentially lose expressive power for node classification," *arXiv preprint arXiv:1905.10947*, 2019.
[50] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun, "Measuring and relieving the over-smoothing problem for graph neural networks from the topological view," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 04, 2020, pp. 3438–3445.
[51] C. Fung, C. J. Yoon, and I. Beschastnikh, "The limitations of federated learning in sybil settings," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, 2020, pp. 301–316.
[52] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International conference on machine learning*. PMLR, 2020, pp. 5132–5143.
[53] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *Computer security–ESORICs 2020: 25th European symposium on research in computer security, ESORICs 2020, guildford, UK, September 14–18, 2020, proceedings, part i 25*. Springer, 2020, pp. 480–501.
[54] Q. Xu, Z. Yang, Y. Zhao, X. Cao, and Q. Huang, "Rethinking label flipping attack: From sample masking to sample thresholding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 6, pp. 7668–7685, 2022.
[55] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International conference on machine learning*. Pmlr, 2018, pp. 5650–5659.
[56] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in neural information processing systems*, vol. 30, 2017.
[57] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," *arXiv preprint arXiv:2012.13995*, 2020.
[58] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *The world wide web conference*, 2019, pp. 2022–2032.
[59] Z. Yang, W. Cohen, and R. Salakhudinov, "Revisiting semi-supervised learning with graph embeddings," in *International conference on machine learning*. PMLR, 2016, pp. 40–48.

[60] C. Fung, C. J. Yoon, and I. Beschastnikh, "Mitigating sybils in federated learning poisoning," *arXiv preprint arXiv:1808.04866*, 2018.
[61] Z. Zhang, X. Cao, J. Jia, and N. Z. Gong, "Fldetector: Defending federated learning against model poisoning attacks via detecting malicious clients," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 2545–2555.
[62] J. Xu, R. Wang, S. Koffas, K. Liang, and S. Picek, "More is better (mostly): On the backdoor attacks in federated graph neural networks," in *Proceedings of the 38th Annual Computer Security Applications Conference*, 2022, pp. 684–698.
[63] F. Liu, S. Lai, Y. Ning, and H. Liu, "Bkd-fedgnn: A benchmark for classification backdoor attacks on federated graph neural network," *arXiv preprint arXiv:2306.10351*, 2023.
[64] S. Sajadmanesh, A. S. Shamsabadi, A. Bellet, and D. Gatica-Perez, "{GAP}: Differentially private graph neural networks with aggregation perturbation," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 3223–3240.
[65] X. Zhu, V. Y. Tan, and X. Xiao, "Blink: Link local differential privacy in graph neural networks via bayesian estimation," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, 2023, pp. 2651–2664.
[66] Z. Xiang, T. Wang, and D. Wang, "Preserving node-level privacy in graph neural networks," in *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2024, pp. 4714–4732.
[67] Y. Xie, M. Fang, and N. Z. Gong, "Poisonedfl: Model poisoning attacks to federated learning via multi-round consistency," *arXiv preprint arXiv:2404.15611*, 2024.
[68] C. A. Choquette-Choo, F. Tramer, N. Carlini, and N. Papernot, "Label-only membership inference attacks," in *International conference on machine learning*. PMLR, 2021, pp. 1964–1974.
[69] F. Tramèr, R. Shokri, A. San Joaquin, H. Le, M. Jagielski, S. Hong, and N. Carlini, "Truth serum: Poisoning machine learning models to reveal their secrets," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 2779–2792.
[70] X. Tang, S. Mahloujifar, L. Song, V. Shejwalkar, M. Nasr, A. Houmansadr, and P. Mittal, "Mitigating membership inference attacks by {Self-Distillation} through a novel ensemble architecture," in *31st USENIX security symposium (USENIX security 22)*, 2022, pp. 1433–1450.
[71] H. Chaudhari, J. Abascal, A. Oprea, M. Jagielski, F. Tramer, and J. Ullman, "Snap: Efficient extraction of private properties with poisoning," in *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2023, pp. 400–417.
[72] J. Wei, Y. Zhang, L. Yu Zhang, C. Chen, S. Pan, K.-L. Ong, J. Zhang, and Y. Xiang, "Extracting private training data in federated learning from clients," *IEEE Transactions on Information Forensics and Security*, vol. 20, pp. 4525–4540, 2025.
[73] C. Fu, X. Zhang, S. Ji, J. Chen, J. Wu, S. Guo, J. Zhou, A. X. Liu, and T. Wang, "Label inference attacks against vertical federated learning," in *31st USENIX security symposium (USENIX Security 22)*, 2022, pp. 1397–1414.
[74] L. Bai, X. Zhang, S. Zhang, Q. Ye, and H. Hu, "Provfl: Property inference attacks against vertical federated learning," *IEEE Transactions on Information Forensics and Security*, 2025.
[75] Y. Wu, X. He, P. Berrang, M. Humbert, M. Backes, N. Z. Gong, and Y. Zhang, "Link stealing attacks against inductive graph neural networks," *arXiv preprint arXiv:2405.05784*, 2024.
[76] L. Meng, Y. Bai, Y. Chen, Y. Hu, W. Xu, and H. Weng, "Devil in disguise: Breaching graph neural networks privacy through infiltration," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, 2023, pp. 1153–1167.
[77] H. Xie, J. Ma, L. Xiong, and C. Yang, "Federated graph classification over non-iid graphs," *Advances in neural information processing systems*, vol. 34, pp. 18 839–18 852, 2021.
[78] C. Meng, S. Rambhatla, and Y. Liu, "Cross-node federated graph neural network for spatio-temporal data modeling," in *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, 2021, pp. 1202–1211.
[79] H. Zhang, T. Shen, F. Wu, M. Yin, H. Yang, and C. Wu, "Federated graph learning–a position paper," *arXiv preprint arXiv:2105.11099*, 2021.
[80] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017, pp. 3–18.

A Appendices

A.1 Attack performance of LIAs

Table 4 reports the complete attack precision and recall results of baseline LIAs (i.e., SLA and LTA) and our poisoning-based LIAs (i.e., LFLIA and GALIA) across five datasets (Cora, Citeseer, Pubmed, DBLP, and IMDB), three GNN backbones (GAT, GCN, and GraphSage), and six aggregation strategies (FedAvg, Median, Trimmed-Mean, Multi-Krum, Fang, and FLTrust).

Across most settings, our poisoning-based LIAs achieve the better attack performance to infer link privacy compared to baseline LIAs, while robust aggregators (e.g., Multi-Krum, Fang, and FLTrust) generally reduce the attack performance compared with FedAvg but do not eliminate the influence of the propagation of poisoning attacks via message passing mechanisms through cross-client and intra-client edges.

A.2 Performance of subgraph reconstruction

A.2.1 Subgraph reconstruction. We further examine the ability of inferred structural relations by LIAs to reconstruct subgraphs. Specifically, given a new target node set T' and its graph embedding $H_{T'}$. The $\mathcal{A}$ aims to reconstruct the subgraph G_S for T'. The reconstruction process proceeds as follows:

Step 1: Infer structural relationship. For each candidate node $v' \in T'$, $\mathcal{A}$ applies LIAs to get the structural relationships (i.e., 1-hop, 2-hop, and 3-hop). Then, the inferred relations are recorded in a pool

$$\mathcal{P} = \left\{ \left(v', \{N^{(1)}(v'), N^{(2)}(v'), N^{(3)}(v')\} \right) \,\middle|\, v' \in T' \right\}, \quad (22)$$

where $N^{(n)}(v')$ denotes the set of inferred n-hop neighbours of v' for $n \in \{1, 2, 3\}$.

Step 2: Construct subgraph: Based on the inferred structural relationship $\mathcal{P}$, $\mathcal{A}$ performs a Breadth-First Search (BFS). Starting from nodes in T', BFS iteratively adds their inferred 1-hop, 2-hop, and 3-hop neighbours to constructed a subgraph G_S.

Step 3: Calculate embedding-based similarity: $\mathcal{A}$ employs a graph embedding extractor model (e.g., GraphSage [2]) to learn the graph representation H_S of the reconstructed subgraph G_S, and then adopt the cosine similarity to calculate the similarity between the two graph embeddings,

$$\mathrm{Sim}(H_S, H_{T'}) = \frac{H_S \cdot H_{T'}}{\|H_S\|_2 \, \|H_{T'}\|_2}, \quad (23)$$

To evaluate the performance of subgraph reconstruction, we consider a K-Nearest Neighbors (KNN) method as baseline to reconstruct subgraph of target node set T'. For each node in T', the KNN selects its top-k most similar nodes in the feature space to connect to reconstruct the subgraph.

The effectiveness of reconstruction is quantified by$\mathrm{Sim}(H_S, H_{T'})$, a higher value of $\mathrm{Sim}(H_S, H_{T'})$ indicates a greater similarity between the reconstructed and original subgraph embeddings, meaning a more successful reconstruction.

A.2.2 Evaluation. As our poisoning-based LIA relies on training-induced representation shifts, which are weak or absent for nodes that are not the members of training dataset, to obtain T', $\mathcal{A}$ first conduct a existing membership inference attack (MIA) [80] to infer member nodes on a set of attack sample, where the attack sample are randomly sampled from the dataset excluding the local dataset of malicious client in our experiment of subgraph reconstruction. This pre-filtering both improves noise ratio for structural inference and reduces computational cost. These inferred member nodes are referred as T'.

Table 5 shows the accuracy of MIA on the target model FedGNNs under FedAvg with an overlapping rate of 0.5. The MIA achieves above 83% across datasets. Using the MIA-predicted members as

Table 4: Attack performance of LIAs. pr.: precision. re.: recall. Baselines: stealing link inference attack (SLA) [25], LinkTeller (LTA) [27]. Ours: label-flipping link inference attack (LFLIA), gradient-ascending link inference attack (GALIA). "_": best results, *bold*: second best.

AGRs		FedAvg						Medium						Trimmed-Mean					
Dataset	Method	GAT		GCN		GraphSage		GAT		GCN		GraphSage		GAT		GCN		GraphSage	
		pr	re	pr	re	pr	re	pr	re	pr	re	pr	re	pr	re	pr	re	pr	re
Cora	SLA	0.3649	0.4576	0.3784	0.4746	0.3716	0.4661	0.3176	0.3983	0.3311	0.4153	0.3243	0.4068	0.2973	0.3729	0.3108	0.3898	0.3041	0.3814
	LTA	0.7538	0.8305	0.7692	0.8475	0.7615	0.8390	0.7077	0.7797	0.7231	0.7966	0.7154	0.7881	0.6769	0.7458	0.6923	0.7627	0.6846	0.7542
	LFLIA	**0.9262**	**0.9576**	**0.9350**	**0.9746**	**0.9344**	**0.9661**	**0.8770**	**0.9068**	**0.8862**	**0.9237**	**0.8852**	**0.9153**	**0.8443**	**0.8729**	**0.8607**	**0.8898**	**0.8525**	**0.8814**
	GALIA	**0.9426**	**0.9746**	**0.9512**	**0.9915**	**0.9431**	**0.9831**	**0.8934**	**0.9237**	**0.9024**	**0.9407**	**0.8943**	**0.9322**	**0.8607**	**0.8898**	**0.8770**	**0.9068**	**0.8689**	**0.8983**
Citeseer	SLA	0.3516	0.4639	0.3672	0.4845	0.3594	0.4742	0.3281	0.4330	0.3438	0.4536	0.3359	0.4433	0.2813	0.3711	0.2969	0.3918	0.2891	0.3814
	LTA	0.7143	0.7732	0.7333	0.7938	0.7238	0.7835	0.6857	0.7423	0.7048	0.7629	0.6952	0.7526	0.6286	0.6804	0.6476	0.7010	0.6381	0.6907
	LFLIA	**0.8800**	**0.9072**	**0.8911**	**0.9278**	**0.8900**	**0.9175**	**0.8500**	**0.8763**	**0.8614**	**0.8969**	**0.8600**	**0.8866**	**0.8100**	**0.8351**	**0.8020**	**0.8351**	**0.8000**	**0.8247**
	GALIA	**0.9000**	**0.9278**	**0.9109**	**0.9485**	**0.9010**	**0.9381**	**0.8700**	**0.8969**	**0.8812**	**0.9175**	**0.8713**	**0.9072**	**0.8100**	**0.8351**	**0.8218**	**0.8557**	**0.8119**	**0.8454**
Pubmed	SLA	0.3169	0.3947	0.3662	0.4561	0.3404	0.4211	0.2887	0.3596	0.3380	0.4211	0.3121	0.3860	0.2606	0.3246	0.3099	0.3860	0.2817	0.3509
	LTA	0.7698	0.8509	0.7857	0.8684	0.7778	0.8596	0.7381	0.8158	0.7540	0.8333	0.7460	0.8246	0.7063	0.7807	0.7222	0.7982	0.7143	0.7895
	LFLIA	**0.9068**	**0.9386**	**0.9160**	**0.9561**	**0.9153**	**0.9474**	**0.8390**	**0.8684**	**0.8559**	**0.8860**	**0.8475**	**0.8772**	**0.7797**	**0.8070**	**0.7966**	**0.8246**	**0.7881**	**0.8158**
	GALIA	**0.9237**	**0.9561**	**0.9328**	**0.9737**	**0.9244**	**0.9649**	**0.8803**	**0.9035**	**0.8898**	**0.9211**	**0.8814**	**0.9123**	**0.8362**	**0.8509**	**0.8462**	**0.8684**	**0.8376**	**0.8596**
DBLP	SLA	0.2984	0.3742	0.3121	0.3914	0.3052	0.3828	0.2706	0.3393	0.2849	0.3573	0.2777	0.3483	0.2429	0.3046	0.2571	0.3224	0.2500	0.3135
	LTA	0.7231	0.7965	0.7385	0.8135	0.7308	0.8050	0.6769	0.7455	0.6923	0.7625	0.6846	0.7540	0.6385	0.7035	0.6538	0.7200	0.6462	0.7115
	LFLIA	**0.8525**	**0.8814**	**0.8689**	**0.8983**	**0.8607**	**0.8898**	**0.7869**	**0.8136**	**0.8033**	**0.8305**	**0.7951**	**0.8220**	**0.7377**	**0.7627**	**0.7541**	**0.7797**	**0.7459**	**0.7712**
	GALIA	**0.8852**	**0.9153**	**0.9016**	**0.9322**	**0.8934**	**0.9237**	**0.8197**	**0.8475**	**0.8361**	**0.8644**	**0.8279**	**0.8559**	**0.7377**	**0.7627**	**0.7869**	**0.8136**	**0.7787**	**0.8051**
IMDB	SLA	0.2869	0.3597	0.3006	0.3769	0.2937	0.3683	0.2591	0.3248	0.2734	0.3428	0.2662	0.3338	0.2314	0.2901	0.2456	0.3079	0.2385	0.2990
	LTA	0.7038	0.7750	0.7192	0.7920	0.7115	0.7835	0.6577	0.7245	0.6731	0.7415	0.6654	0.7330	0.6192	0.6820	0.6346	0.6990	0.6269	0.6905
	LFLIA	**0.8328**	**0.8609**	**0.8492**	**0.8778**	**0.8410**	**0.8693**	**0.7672**	**0.7931**	**0.7836**	**0.8100**	**0.7754**	**0.8015**	**0.7180**	**0.7422**	**0.7344**	**0.7592**	**0.7262**	**0.7507**
	GALIA	**0.8656**	**0.8948**	**0.8819**	**0.9117**	**0.8738**	**0.9032**	**0.8000**	**0.8270**	**0.8164**	**0.8439**	**0.8082**	**0.8354**	**0.7508**	**0.7761**	**0.7672**	**0.7931**	**0.7590**	**0.7846**
AGRs		Multi-krum						Fang						FLTrust					
Dataset	Method	GAT		GCN		GraphSage		GAT		GCN		GraphSage		GAT		GCN		GraphSage	
		pr	re	pr	re	pr	re	pr	re	pr	re	pr	re	pr	re	pr	re	pr	re
Cora	SLA	0.2770	0.3475	0.2905	0.3644	0.2838	0.3559	0.2230	0.2797	0.2365	0.2966	0.2297	0.2881	0.2027	0.2542	0.2162	0.2712	0.2095	0.2627
	LTA	0.6538	0.7203	**0.6692**	**0.7373**	0.6605	0.7238	0.5923	0.6525	0.6077	0.6695	0.6000	0.6610	0.5692	0.6271	0.5846	0.6441	0.5769	0.6356
	LFLIA	**0.8197**	**0.8475**	**0.6692**	**0.7373**	**0.6615**	**0.7288**	**0.7541**	**0.7797**	**0.7705**	**0.7966**	**0.7623**	**0.7881**	**0.7295**	**0.7542**	**0.7459**	**0.7712**	**0.7377**	**0.7627**
	GALIA	**0.8361**	**0.8644**	**0.8525**	**0.8814**	**0.8443**	**0.8729**	**0.7705**	**0.7966**	**0.7869**	**0.8136**	**0.7787**	**0.8051**	**0.7459**	**0.7712**	**0.7623**	**0.7881**	**0.7541**	**0.7797**
Citeseer	SLA	0.2734	0.3608	0.2891	0.3814	0.2813	0.3711	0.1563	0.2062	0.1641	0.2165	0.1563	0.2062	0.1385	0.1856	0.1462	0.1959	0.1385	0.1856
	LTA	0.6154	0.6598	0.6346	0.6804	0.6250	0.6701	0.5673	0.6082	0.5865	0.6289	0.5769	0.6186	0.5534	0.5876	0.5728	0.6082	0.5631	0.5979
	LFLIA	**0.7700**	**0.7938**	**0.7822**	**0.8144**	**0.7800**	**0.8041**	**0.7172**	**0.7320**	**0.7374**	**0.7526**	**0.7273**	**0.7423**	**0.7071**	**0.7216**	**0.7273**	**0.7423**	**0.7172**	**0.7320**
	GALIA	**0.7900**	**0.8144**	**0.8020**	**0.8351**	**0.7921**	**0.8247**	**0.7475**	**0.7629**	**0.7677**	**0.7835**	**0.7576**	**0.7732**	**0.7700**	**0.7938**	**0.7900**	**0.8144**	**0.7800**	**0.8041**
Pubmed	SLA	0.2113	0.2632	0.2254	0.2807	0.2183	0.2719	0.1761	0.2193	0.1901	0.2368	0.1831	0.2281	0.1620	0.2018	0.1761	0.2193	0.1690	0.2105
	LTA	0.6746	**0.7456**	0.6905	**0.7632**	0.6825	**0.7544**	0.6032	0.6667	0.6190	0.6842	0.6111	0.6754	0.5714	0.6316	0.5873	0.6491	0.5794	0.6404
	LFLIA	**0.7203**	**0.7456**	**0.7373**	**0.7632**	**0.7288**	**0.7544**	**0.6525**	**0.6754**	**0.6695**	**0.6930**	**0.6610**	**0.6842**	**0.6186**	**0.6404**	**0.6356**	**0.6579**	**0.6271**	**0.6491**
	GALIA	**0.7913**	**0.7982**	**0.8017**	**0.8158**	**0.7931**	**0.8070**	**0.7217**	**0.7281**	**0.7391**	**0.7456**	**0.7304**	**0.7368**	**0.6870**	**0.6930**	**0.7043**	**0.7105**	**0.6957**	**0.7018**
DBLP	SLA	0.2230	0.2797	0.2365	0.2966	0.2297	0.2881	0.2027	0.2542	0.2162	0.2712	0.2095	0.2627	0.1827	0.2291	0.1962	0.2460	0.1895	0.2376
	LTA	0.5923	0.6525	0.6077	0.6695	0.6000	0.6610	0.5692	0.6271	0.5846	0.6441	0.5769	0.6356	0.5385	0.5932	0.5538	0.6102	0.5462	0.6017
	LFLIA	**0.7541**	**0.7797**	**0.7705**	**0.7966**	**0.7623**	**0.7881**	**0.7295**	**0.7542**	**0.7459**	**0.7712**	**0.7705**	**0.7966**	**0.7077**	**0.7314**	**0.7231**	**0.7475**	**0.7541**	**0.7797**
	GALIA	**0.7869**	**0.8136**	**0.8033**	**0.8305**	**0.7951**	**0.8220**	**0.7623**	**0.7881**	**0.7787**	**0.8051**	**0.7377**	**0.7627**	**0.7459**	**0.7712**	**0.7623**	**0.7881**	**0.7154**	**0.7397**
IMDB	SLA	0.2115	0.2652	0.2250	0.2821	0.2182	0.2736	0.1912	0.2397	0.2047	0.2566	0.1979	0.2481	0.1712	0.2146	0.1847	0.2315	0.1780	0.2231
	LTA	0.5731	0.6313	0.5885	0.6483	0.5808	0.6398	0.5500	0.6059	0.5654	0.6229	0.5577	0.6144	0.5192	0.5720	0.5346	0.5889	0.5269	0.5805
	LFLIA	**0.7344**	**0.7592**	**0.7508**	**0.7761**	**0.7426**	**0.7677**	**0.7098**	**0.7339**	**0.7590**	**0.7846**	**0.7180**	**0.7422**	**0.6885**	**0.7117**	**0.7426**	**0.7677**	**0.6962**	**0.7198**
	GALIA	**0.7672**	**0.7931**	**0.7836**	**0.8100**	**0.7754**	**0.8015**	**0.7426**	**0.7677**	**0.7262**	**0.7507**	**0.7508**	**0.7761**	**0.7262**	**0.7507**	**0.7038**	**0.7276**	**0.7180**	**0.7422**

the target set, we then evaluate worst-case subgraph reconstruction on MIA-predicted members of FedGCN. Table 6 summarises the reconstruction performance of our poisoning-based LIAs (LFLIA, GALIA) on Cora, Citeseer, and PubMed with cosine similarity.

As we can see, BFS-GALIA achieves the best reconstruction on all the three datasets, outperforming BFS-LFLIA by Cora (1.8%), Citeseer (2.5%) and Pubmed (8.7%). BFS-GALIA outperforms KNN by Cora (8.3%), Citeseer (12.4%) and Pubmed (11.2%). BFS-LFLIA also improves over KNN by Cora (6.5%), Citeseer (9.9%) and Pubmed (2.5%). These results demonstrates that LIA-inferred structural relationships are enhance subgraph reconstruction over feature-only baseline. The LIA-inferred structural relations guide the BFS expansion along true structural paths to reconstruct neighbourhoods with realistic degree distributions and clustering. In contrast, KNN may enforce a fixed out-degree k based on feature proximity.

Table 5: Membership inference attack accuracy.

Model	Dataset		
	Cora	Citeseer	Pubmed
FedGAT	0.896	0.882	0.841
FedGCN	0.891	0.865	0.837
FedGraphSage	0.915	0.92	0.853

Table 6: Attack performance of subgraph reconstruction measured by cosine distance similarity.

Method	Dataset		
	Cora	Citeseer	Pubmed
BFS-LFLIA	0.876	0.878	0.867
BFS-GALIA	0.894	0.903	0.954
KNN	0.811	0.779	0.842

A.3 Comparative summary of LIAs

Table 7 summarises the key characteristics of LIA techniques and our proposed LIAs. The table contrasts their system settings, attack stages, required adversarial capabilities, use of auxiliary nodes, and core strategies, and highlights key limitations. Most of existing LIAs rely on centralised or vertically partitioned assumptions, often require strong access to global node features or topology, or depend on unrestricted global auxiliary-node injection.

In contrast, our attack operates under realistic federated constraints, where the adversary controls only its own local client and can inject auxiliary nodes only into its local subgraph. Such auxiliary-node injection is also practical in real-world federated settings, as an adversarial client typically has full control over its own local dataset and can legitimately create new records that appear as additional nodes.

A.4 Ablation study

A.4.1 Overlapping rate. Figs. 9, 10 and 11 show attack precision of representative LIAs under three overlapping rates ranging from 0.2 to 0.4 across three datasets (Cora, Citeseer, and PubMed), three GNN backbones (GCN, GAT, and GraphSage), and six aggregators (FedAvg, Median, Trimmed-Mean, Multi-Krum, Fang, and FLTrust).

Overall, increasing the overlapping rate tends to strengthen poisoning-based LIAs, especially under FedAvg, Median, and Trimmed-Mean. For example, under FedAvg with GCN, GALIA improves from 0.899 to 0.975 on Cora, 0.899 to 0.982 on Citeseer, and 0.899 to 0.979 on PubMed as the overlap increases from 0.2 to 0.4. LFLIA shows a similar upward trend (e.g., Cora: 0.874 to 0.939). In contrast, robust aggregators consistently decrease the attack precision, yet the overlap effect is not fully removed (e.g., under FLTrust on Cora with GCN, GALIA varies 0.726 to 0.681 and then to 0.739).

These findings suggest a relationship between overlap and attack performance: higher overlap provides richer cross-client structural consistency (i.e., more shared neighbourhoods) that makes poisoning signals easier to propagate, accumulate and separate linked compared with unlinked pairs, whereas robust aggregation primarily reduces magnitude and can introduce small fluctuations rather than fully eliminating the overlap-driven gain.

A.4.2 Number of malicious clients. In Figs. 12, 13 and 14, it can be seen that there is a strong positive correlation between the number of malicious clients and the precision across all FedGNN architectures.

The precision averaged over models and aggregators increases from 79.5% to 86.0% on Cora, 73.2% to 78.7% on Citeseer and 75.9% to 81.5% on Pubmed, respectively. The precision roughly increases from 0.006 to 0.007 with per additional malicious client. Especially, when the number is more than 6 participants (40%) during the FedGNN trainings, the attack success rate increases (e.g., Cora from 3.38% to 4.71%, Citeseer from 2.37% to 4.94%, and Pubmed from 2.74% to 4.45%).

When there are 10 malicious clients, the best LIA performance is achieved by the FedGCN model with FedAvg aggregator, which are 96.0% on Cora, 90.2% on Citeseer, and 93.7% on PubMed. Across architectures averaged over aggregators, FedGCN achieves higher precision on Cora and Pubmed, with improvements ranging from approximately 6% to 7.5%. In contrast, FedGraphSage slightly outperforms on Citeseer by approximately 5.7%. A small decrement in is observed when the number of malicious clients varies from 4 to 6 under robust aggregators (e.g., Fang and FLTrust), but the overall trend remains upward.

A.4.3 Attack timing. Figs. 15, 16 and 17 show that, across three datasets, GNN architectures, and aggregation strategies. GALIA exhibits a late-stage advantage with respect to attack timing. Specifically, the attack precision generally increases when adversaries launch GALIA during the attack timing from 150-300 training epochs. Then, the attack precision slightly declines at 350-th training epoch. This scenario is stably occurs across GCN, GAT, and GraphSage. That is, GALIA is most effective when adversaries launch poisoning inference attack at FedGNN's convergence stage.

In contrast, attacks launched at early stages (e.g., 150-th epoch) yield weaker attack performance. For instance, under FedAvg, precision improves by approximately 11.9%–15.3% on Cora, 18.6%–20.9% on Citeseer, and 10.7%–18.2% on PubMed when comparing attack timing at 300-th epoch.

We found the inferior performance of early-stage attacks (e.g., at the 150-th epoch) may be attributed to the instability of node representations during initial training. At this stage, model parameters and message-passing representations are still rapidly evolving, and poisoned gradients injected by adversaries are largely overwhelmed or diluted by subsequent clean updates. As a result, the injected perturbations may fail to accumulate into consistent structural signals that can be reliably exploited for link inference.

When attacks launched near the convergence stage (i.e., around 250–300 epochs), GALIA achieves the best attack performance. During the convergence stage, node embeddings have largely stabilised, which allows poisoned gradients to persist across training rounds and diffuse effectively through message passing, thereby amplifying link-dependent differences in posterior behaviour. However, when the attack is launched too late (e.g., at around 350-th epoch), the marginal gain diminishes or slightly degrades, as the optimisation budget becomes limited. That is, gradients are smaller, learning

Table 7: A summarised overview of LIAs and Ours including system settings, attack stages, required adversarial capabilities, use of auxiliary nodes, and core strategies to infer links, and key limitations or unrealistic assumptions.

Approach	System Setting	Attack Stage	Access/Control	Auxiliary Node	Strategy	FedGNN	Limitations
SLA [25]	Centralised GNN	Inference	Link-prediction queries; node features; or shadow data	✗	Shadow models, embedding similarity, ML classifiers	✗	Requires strong access (features/queries/shadow); cannot operate in federated settings; no poisoning ability.
LinkTeller [27]	Vertical GNN	Training/Influence	Full control over node features on one party	✗	Feature perturbation; influence propagation analysis	✗	Assumes unrealistic complete feature control.
VertexSerum [28]	Centralised GNN	Training	Modify features in a global subgraph via PGD	✗	Imperceptible training-stage feature poisoning	✗	Requires global subgraph access.
GNNBleed [35] **Devil [76]**	Centralised GNN	Inference	Inject global auxiliary nodes and modify their edges and features	Global injection	Posterior shift analysis via auxiliary node perturbation	✗	Assumes unrestricted global node injection.
LinkThief [36]	Centralised GNN	Inference	Access to embeddings and model outputs	✗	Similarity, structural priors and probabilistic scoring	✗	Depends on structural priors; ineffective in sparse and heterogeneous graphs.
Ours	**Federated GNN**	Training/inference	Control only its own local dataset and training (black-box or white-box)	Local injection only (auxiliary nodes restricted to attacker's own client)	Local poisoning, local auxiliary nodes, posterior shift across FL rounds	✓	First LIA for FedGNN; auxiliary nodes limited to local clients; relies on aggregated model behaviour.

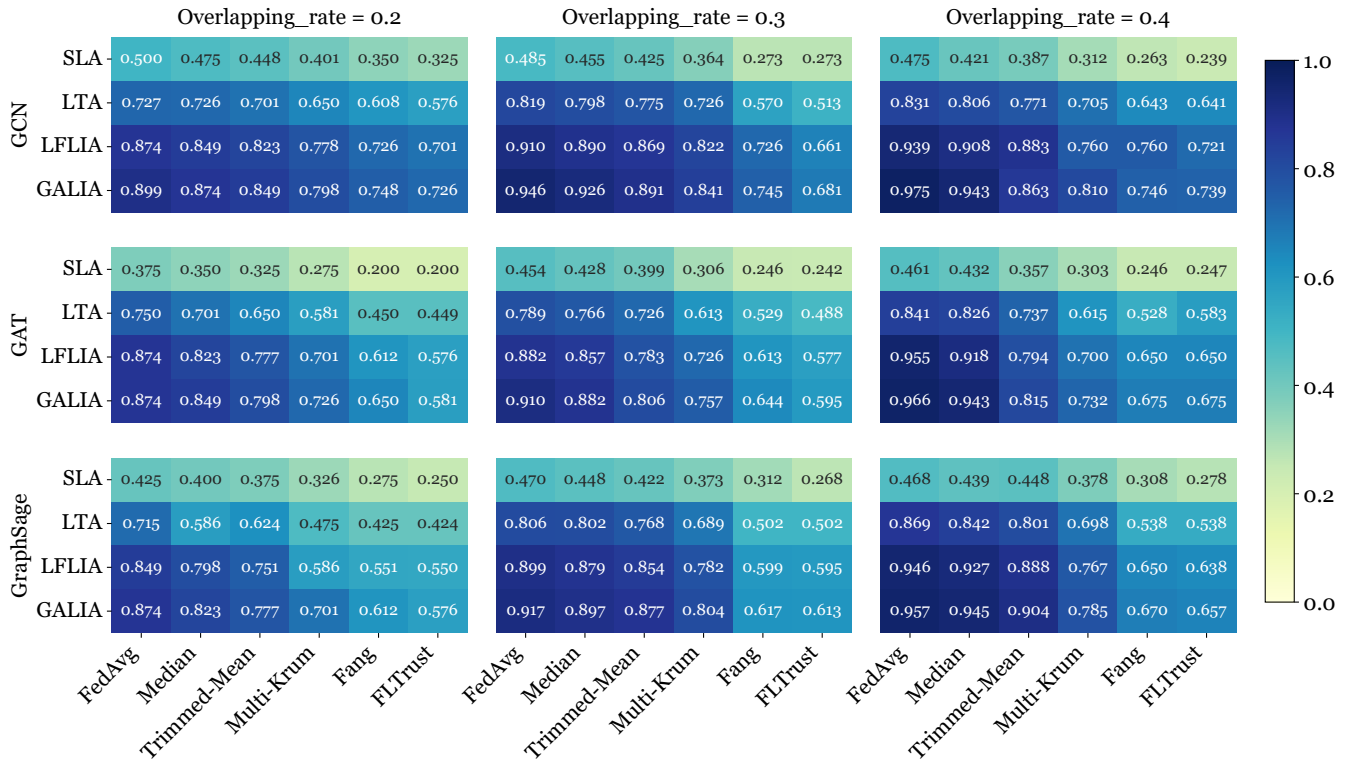


Figure 9: Precision under overlapping rate analysis of link inference attack performances on Cora dataset.

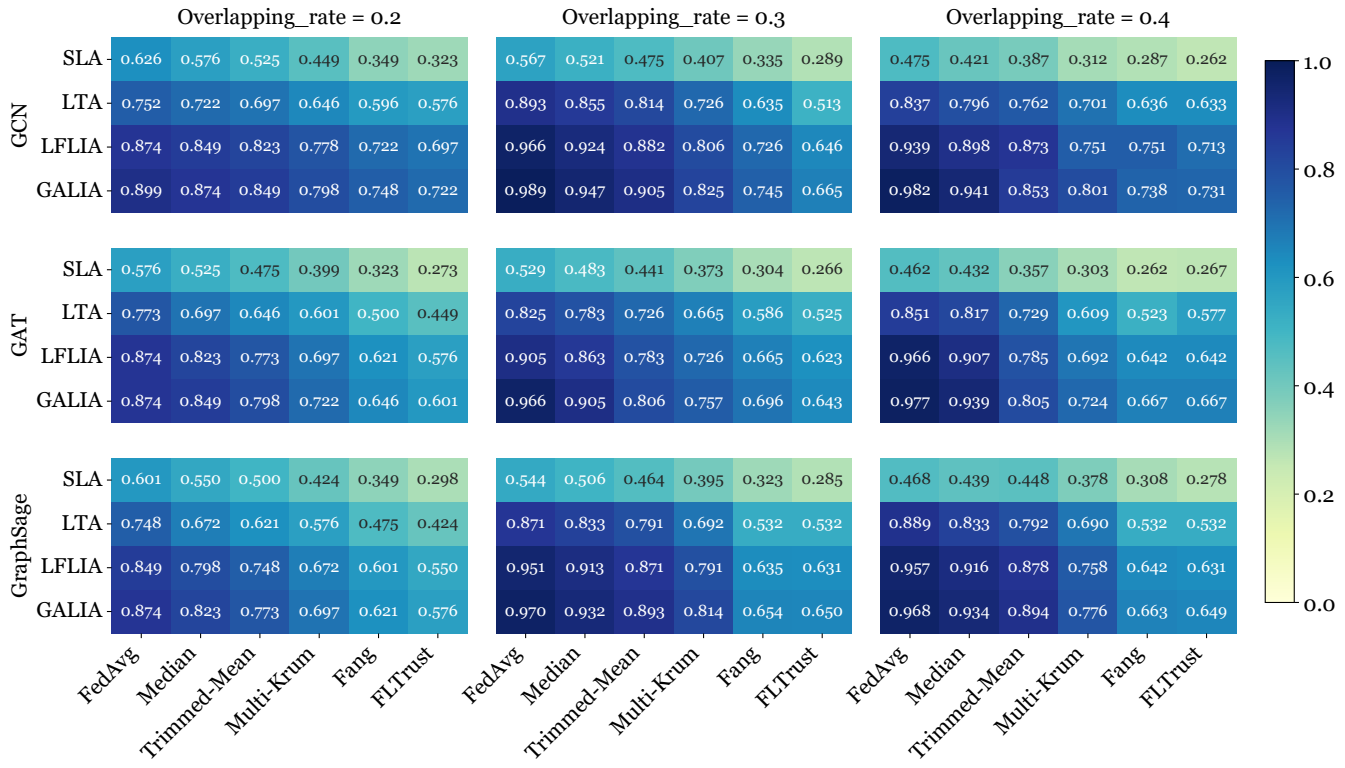


Figure 10: Precision under overlapping rate analysis of link inference attack performances on Citeseer dataset.

rates decay, and robust aggregation mechanisms can easily suppress anomalous updates, leaving insufficient opportunity for poisoning signals to further propagate.

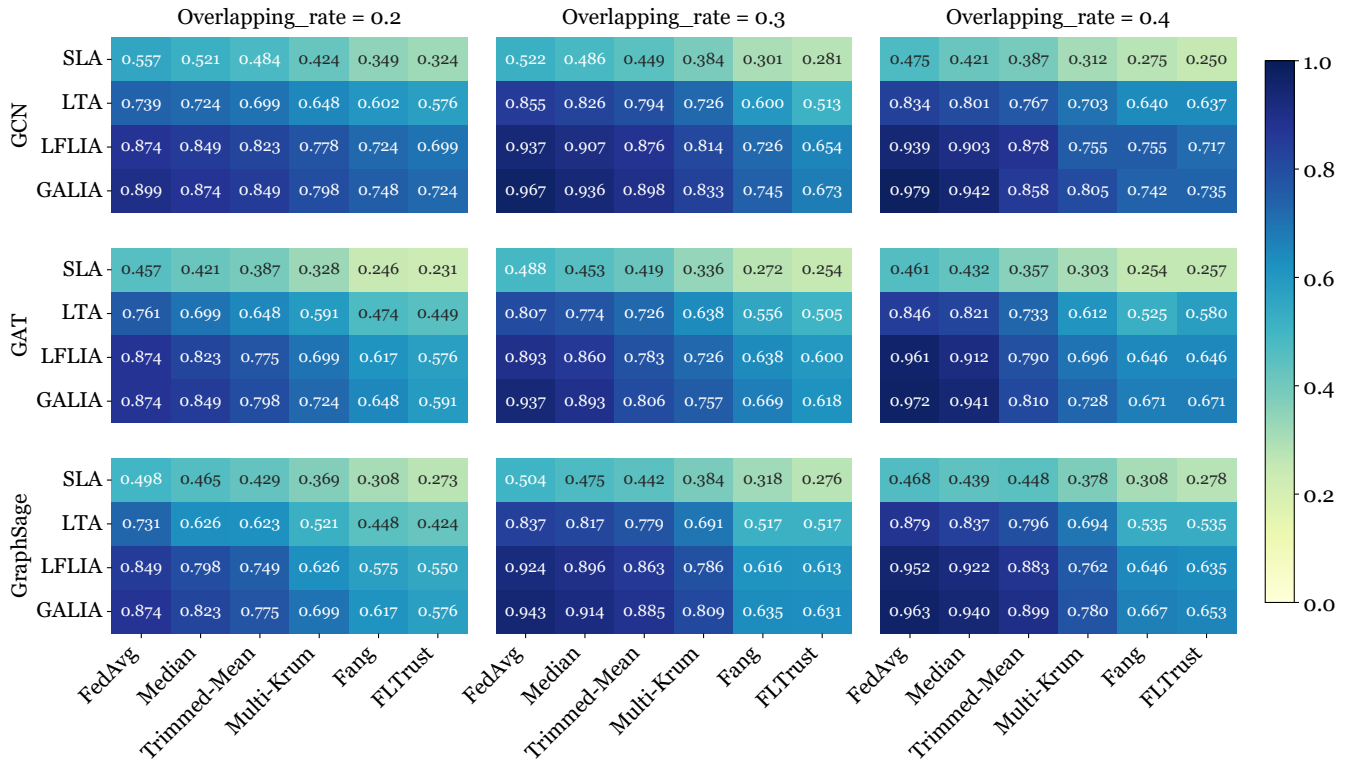
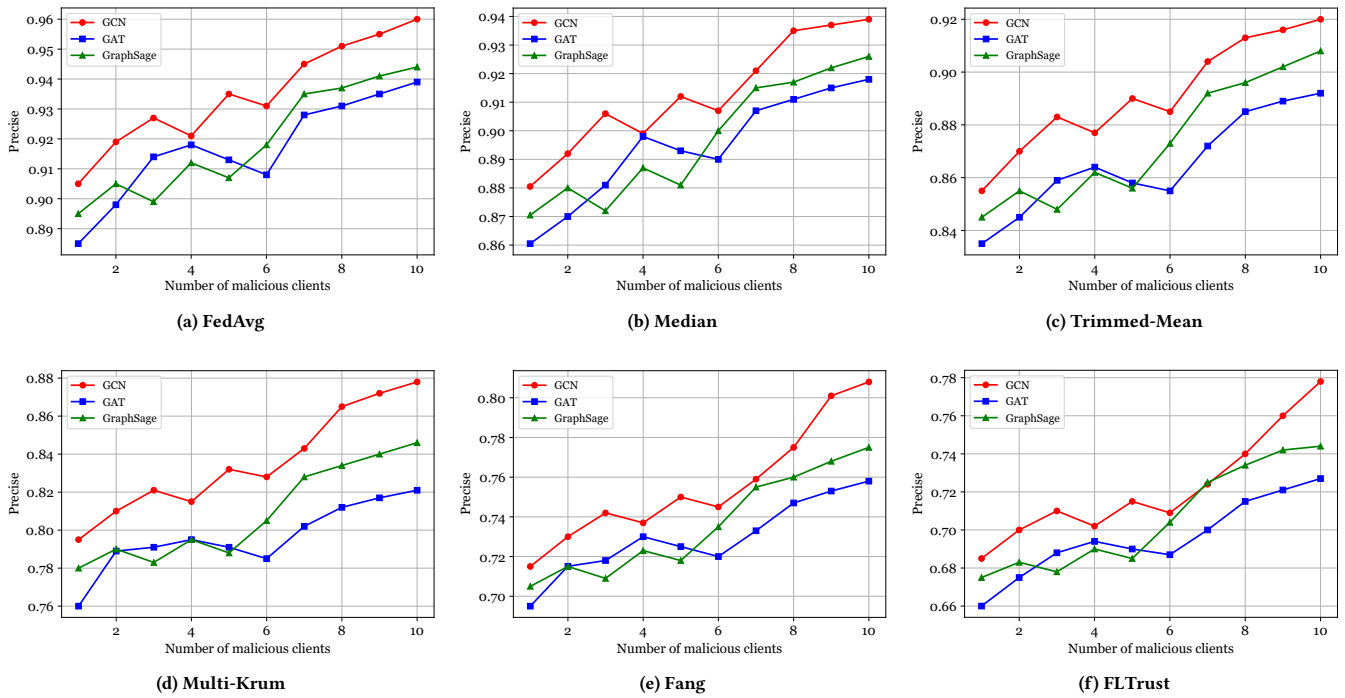


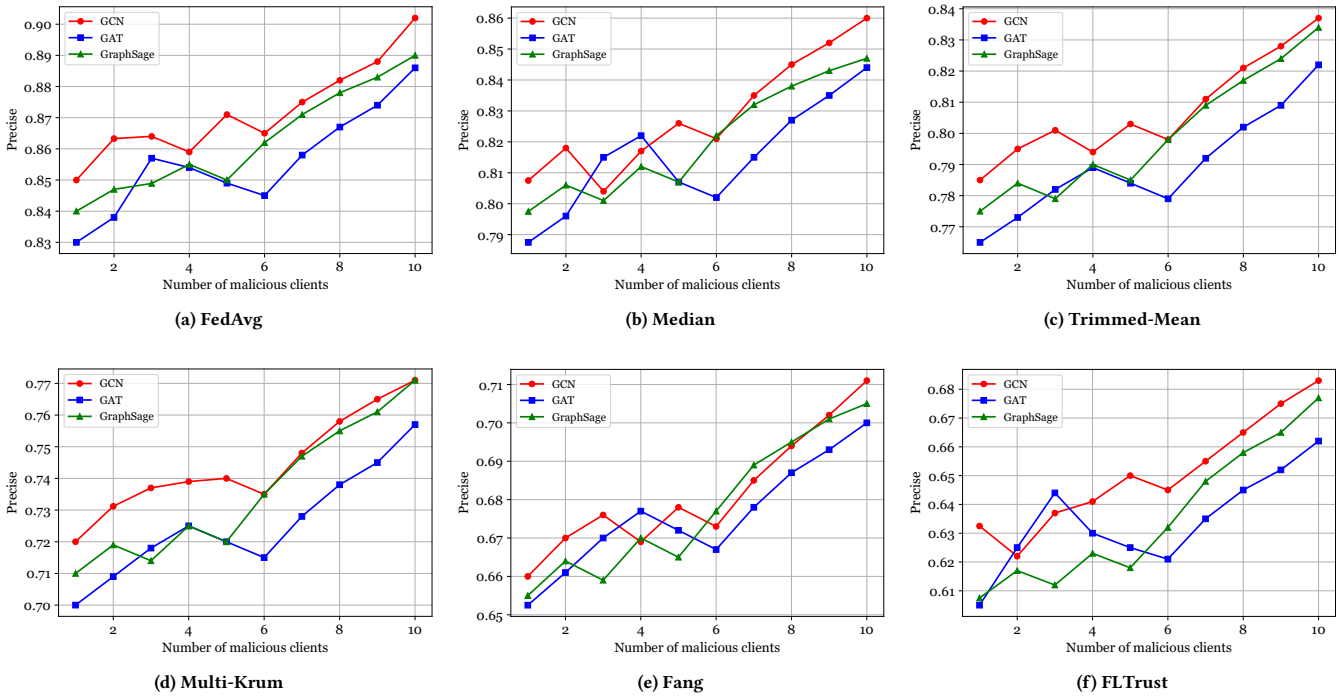
Figure 11: Precision under overlapping rate analysis of link inference attack performances on Pubmed dataset.



(a) FedAvg **(b) Median** **(c) Trimmed-Mean**

(d) Multi-Krum **(e) Fang** **(f) FLTrust**

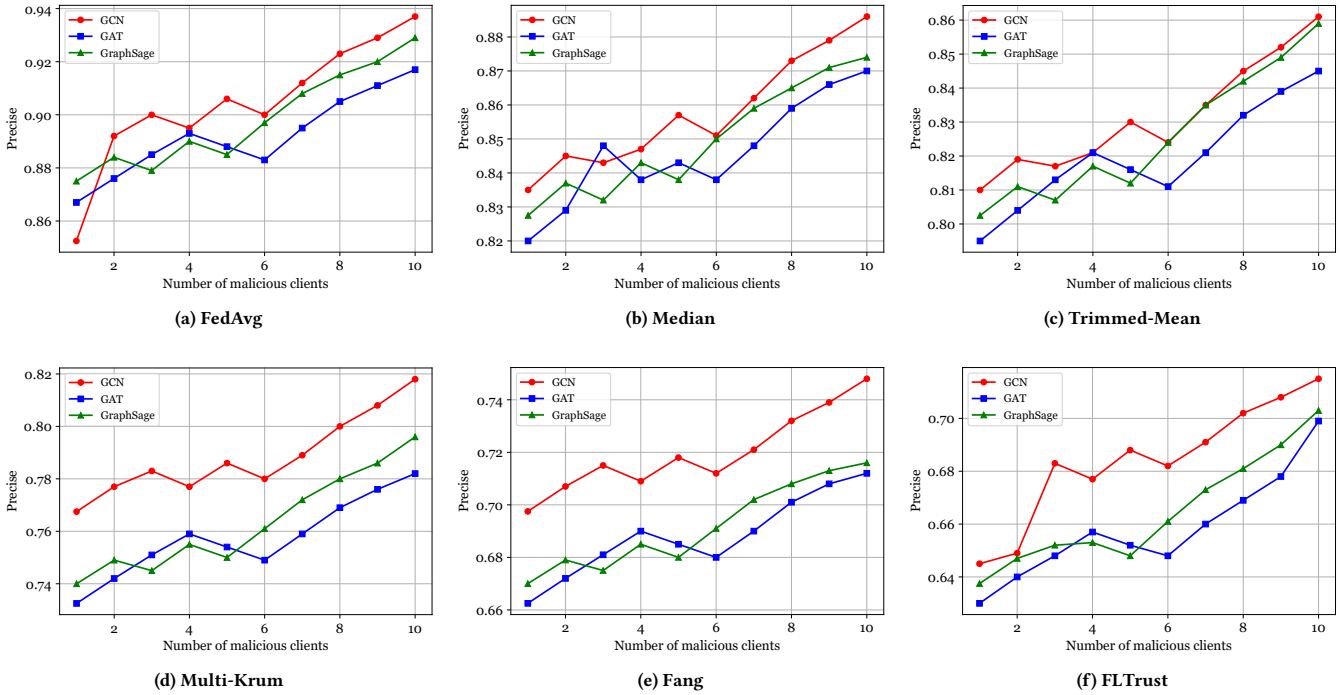
Figure 12: Precision under number of malicious clients analysis of link inference attack performances on Cora dataset.



(a) FedAvg (b) Median (c) Trimmed-Mean

(d) Multi-Krum (e) Fang (f) FLTrust

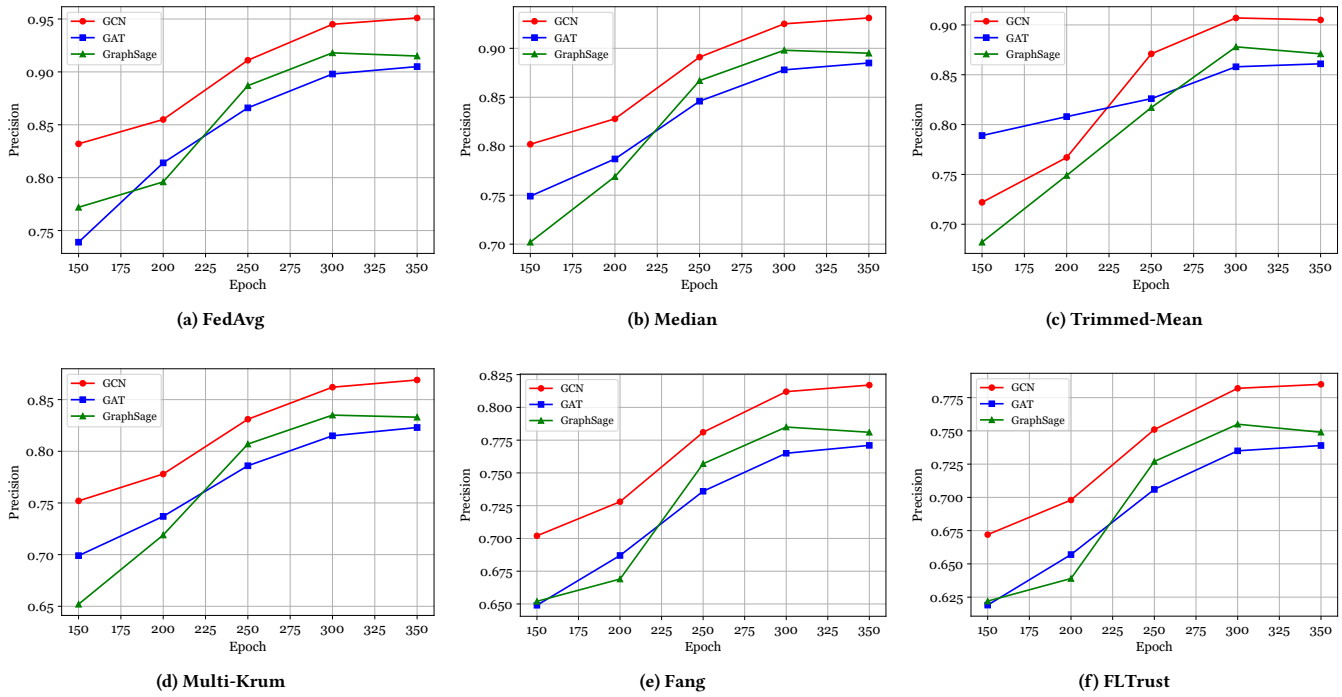
Figure 13: Precision under number of malicious clients analysis of link inference attack performances on Citeseer dataset.



(a) FedAvg (b) Median (c) Trimmed-Mean

(d) Multi-Krum (e) Fang (f) FLTrust

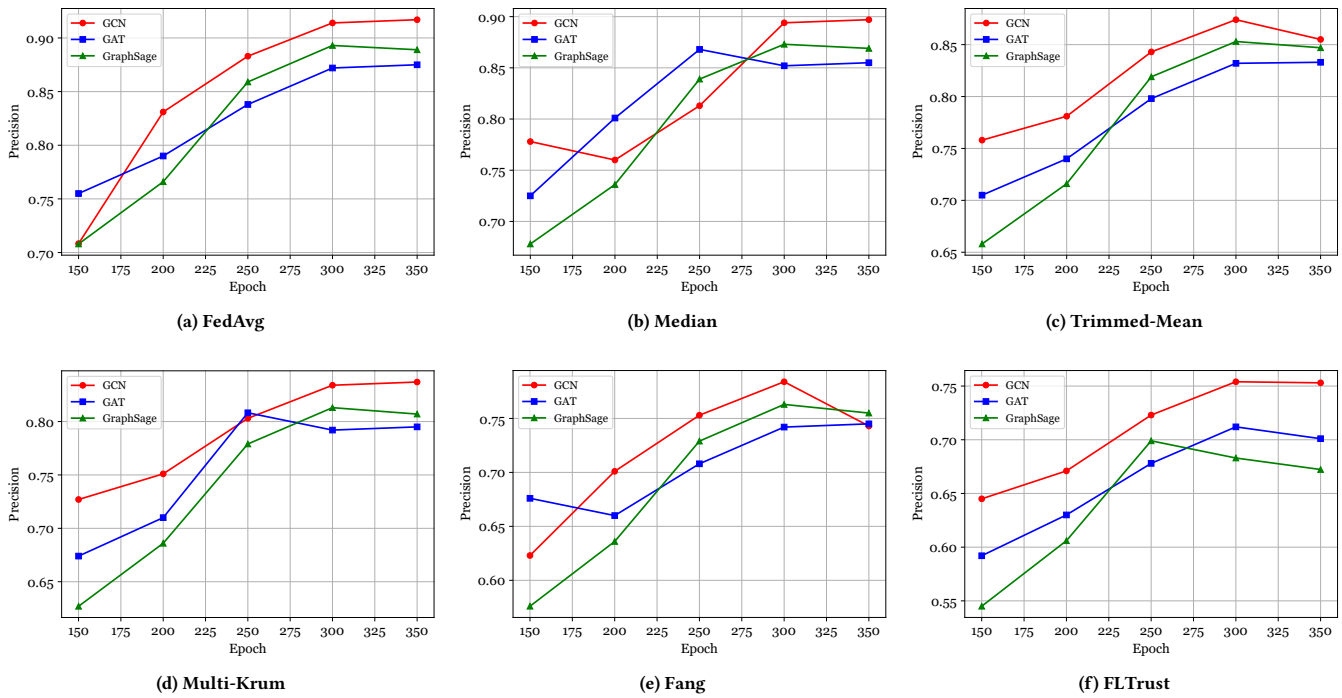
Figure 14: Precision under number of malicious clients analysis of link inference attack performances on Pubmed dataset.



(a) FedAvg (b) Median (c) Trimmed-Mean

(d) Multi-Krum (e) Fang (f) FLTrust

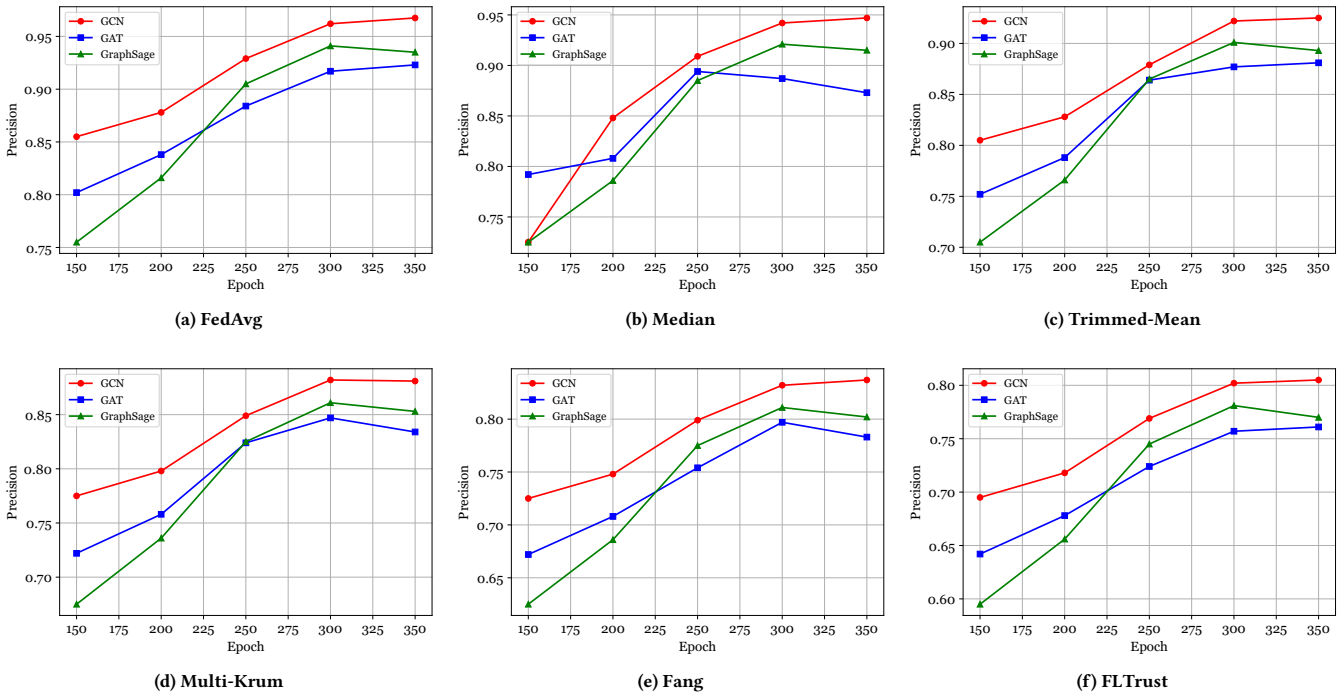
Figure 15: Precision under attack timing analysis of link inference attack performances on Cora dataset.



(a) FedAvg (b) Median (c) Trimmed-Mean

(d) Multi-Krum (e) Fang (f) FLTrust

Figure 16: Precision under attack timing analysis of link inference attack performances on Citeseer dataset.



(a) FedAvg (b) Median (c) Trimmed-Mean

(d) Multi-Krum (e) Fang (f) FLTrust

Figure 17: Precision under attack timing analysis of link inference attack performances on Pubmed dataset.