Subgraph Structure Membership Inference Attacks against Graph Neural Networks

Xiuling Wang
Stevens Institute of Technology
Hoboken, NJ, USA
xwang193@stevens.edu

Wendy Hui Wang
Stevens Institute of Technology
Hoboken, NJ, USA
hwang4@stevens.edu

ABSTRACT

Graph Neural Networks (GNNs) have been widely applied to various applications across different domains. However, recent studies have shown that GNNs are susceptible to the *membership inference attacks* (MIAs) which aim to infer if some particular data samples were included in the model's training data. While most previous MIAs have focused on inferring the membership of individual nodes and edges within the training graph, we introduce a novel form of membership inference attack called the *Structure Membership Inference Attack* (SMIA) which aims to determine whether a given set of nodes corresponds to a particular target structure, such as a clique or a multi-hop path, within the original training graph. To address this issue, we present novel black-box SMIA attacks that leverage the prediction outputs generated by the target GNN model for inference. Our approach involves training a three-label classifier, which, in combination with shadow training, aids in enabling the inference attack. Our extensive experimental evaluation of three representative GNN models and three real-world graph datasets demonstrates that our proposed attacks consistently outperform three baseline methods, including the one that employs the conventional link membership inference attacks to infer the subgraph structure. Additionally, we design a defense mechanism that introduces perturbations to the node embeddings thus influencing the corresponding prediction outputs by the target model. Our defense selectively perturbs dimensions within the node embeddings that have the least impact on the model's accuracy. Our empirical results demonstrate that the defense effectiveness of our approach is comparable with two established defense techniques that employ differential privacy. Moreover, our method achieves a better trade-off between defense strength and the accuracy of the target model compared to the two existing defense methods.

KEYWORDS

Membership inference attack; Graph Neural Networks; Privacy attacks and defense; Trustworthy machine learning.

1 INTRODUCTION

In recent years, the proliferation of graph data has led to the widespread adoption of Graph Neural Networks (GNNs) as a powerful tool for various machine learning tasks [42, 53, 62]. GNNs have demonstrated exceptional capabilities in modeling complex relationships of graph-structured data in various domains and applications such as social network systems [11, 31], recommendation systems [13, 14], and biological networks [25, 57].

While GNNs have demonstrated remarkable performance across diverse applications, recent studies have shown that they are vulnerable to various privacy attacks, including attribute inference attacks [9, 35, 57], property inference attacks [48, 51, 59, 60], and membership inference attacks [9, 20, 22, 52, 60]. In this paper, we mainly focus on membership inference attacks (MIAs) targeted at GNN models.

In general, MIAs against GNNs aim to infer the presence of specific data objects within the training graphs of the target model. Many existing MIAs focus on either node-level membership (i.e., determining if a particular node exists in the training graph) [22, 36] or edge-level membership (i.e., establishing a connection between two nodes in the training graph) [9, 20, 52, 60]. However, none of these studies have explored structure-level membership, where structures can be more intricate than mere nodes and edges.

In this paper, we consider two fundamental and critical graph structures: *k-cliques* and *k-hop paths* ($k > 1$). A clique represents a subset of nodes within a network that are more densely interconnected among themselves than with the remaining nodes. Meanwhile, a k-hop path indicates that two nodes are connected through k links in the graph. These two structures have been extensively used in various applications such as community detection [15, 50], network measurement [4, 8], and fraud detection [37, 58].

The revelation of the existence of cliques and k-hop paths among a set of target nodes in the training graph can pose significant privacy risks to individuals. Let's consider an online dating network where edges represent romantic relationships among users as an example. In this context, an undirected 2-hop path like $A - B - C$ involving three users A, B, and C can potentially reveal sensitive information. For instance, it may indicate that user B is engaged in or pursuing romantic relationships with both users A and C — a detail that user B may wish to keep confidential. Similarly, inferring a clique among users could disclose not only their potential belonging to the same community but also the strength of cohesion within this group.

Given the extensive application of cliques and k-hop paths in graph analytics and the importance of their privacy, we introduce a novel MIA named *Structure Membership Inference Attack* (SMIA). Briefly speaking, SMIA aims to discern whether a set of k target nodes, in the training graph of the target GNN model, forms either a k-clique or a (k-1)-hop path (i.e., the longest loop-free path).

It is noteworthy that SMIA differentiates from the *Subgraph Inference Attack* (SIA) [60], which focuses on establishing the presence

Proceedings on Privacy Enhancing Technologies 2024(4), 268–290

https://doi.org/10.56553/popets-2024-0116

of a subgraph within the training graph. In contrast, our emphasis is on determining whether the subgraph formed by the provided target nodes encompasses either a clique or a k-hop path. We contend that even the inference of a structure's existence can divulge sensitive information. For instance, inferring the presence of a multi-hop path among professionals employed by rival companies could raise suspicions, even without identifying the specific professionals comprising the path. Moreover, it is worth noting that SIA [60] exclusively considers large subgraphs that constitute a substantial portion of the target graph. It does not address the inference of small subgraphs like k-cliques (triangles) and k-hop paths.

Our contributions. To the best of our knowledge, this work represents the inaugural investigation into the vulnerability of GNN models to SMIAs. We make the following contributions [1].

Problem formulation. We formally define the κ-SMIA problem as a three-label classification task that maps a given set of k target nodes, denoted as V_{att}, into one of three labels. Labels 0 and 1 indicate the presence of a k-clique and a (k-1)-hop path respectively, among the nodes within V_{att} in the training graph, while label 2 indicates the absence of either structure in the training graph.

Attack design. We devise the black-box attacks which infer structure membership using prediction outputs from the target model. These attacks involve constructing the three-class attack classifiers via shadow training [44] and extracting attack features based on the similarity of prediction outputs from the target model. Furthermore, we extend the black-box SMIAs to white-box ones which leverage node embeddings for attack inference. We also extend the scope of SMIAs to encompass SIAs, enabling the inference of the subgraph structure of the target nodes.

Empirical evaluation. Through extensive empirical assessments involving three representative GNN models and real-world graph data, we substantiate the effectiveness of SMIA. Notably, the attack AUC of our attacks can reach up to 0.89 under the non-transfer setting (where training and shadow graphs originate from the same dataset), outperforming three baselines. Remarkably, even in the transfer setting, where shadow and training graphs are drawn from different domains and data distributions, the attack AUC remains substantial, achieving up to 0.83. Additionally, our proposed attacks consistently outperform three baseline methods, including the one that employs the conventional link membership inference attacks [20, 52] to infer the subgraph structure.

Defense design and evaluation. To mitigate GNNs' vulnerability to SMIAs, we propose a defense mechanism that introduces Laplace noise to node embeddings, thereby impacting the posterior outputs of the target model. Notably, we limit perturbation to embedding dimensions of least significance, minimizing the influence on target model accuracy. Empirical assessments demonstrate that our approach achieves defense effectiveness comparable to that of two existing defense techniques. Moreover, our method achieves a better trade-off between defense strength and target model accuracy than the two existing defense methods.

2 PRELIMINARIES

2.1 Subgraph Structures

Essential characteristics of graphs can be discerned through subgroups of nodes/edges represented by subgraphs. Within this context, we explore two fundamental subgraph structures: *k-hop paths* and *k-cliques*.

- *K-hop paths*: A k-hop path denotes a sequence of k adjacent edges connecting distinct nodes. In this paper, we only consider loop-free k-hop paths.
- *K-cliques*: k-clique represents a subgraph comprising k nodes, where each node is interconnected with every other node. Note a 3-clique is occasionally referred to as a triangle. Hence these two terms will be used interchangeably in the paper.

Figure 1 provides illustrative examples of 2-hop and 3-hop paths, as well as 3-cliques and 4-cliques.

2.2 Graph Neural Network

Graph Neural Networks (GNNs) have been widely used for learning over graph-structured data. Given a graph $G(V, E, \mathbf{X})$ where V, E, and $\mathbf{X}$ denote the nodes, edges, and node features, respectively, a message-passing GNN employs a message-passing mechanism that involves the exchange of messages amongst nodes and aggregation of nodes' neighborhood, which is executed as:

$$z_v^{\ell+1} = \phi(z_v^{\ell}, \bigoplus_{v' \in \mathcal{N}(v)} \psi(z_v^{\ell}, z_{v'}^{\ell})) \tag{1}$$

where ϕ is an update function, ψ is a message function, and $\bigoplus$ is an aggregation function (such as sum or max), z_v^{ℓ} denotes the embedding of node v at l-th iteration, and $\mathcal{N}(v)$ denotes the neighbor nodes of v. The initial embeddings are set as the node features.

After k iterations, the application of a Readout function aggregates node embeddings for subsequent predictions of downstream tasks. In the context of node classification, this Readout function, often implemented as a softmax operation, generates a Posterior Probability Vector (PPV) for each node, in which the i-th element indicates the likelihood that the node belongs to the i-th class.

In this paper, our focus is on three prominent GNN models: Graph Convolutional Network (GCN) [26], SAGE [18], and Graph Attention Network (GAT) [49]. These GNN models employ various aggregation mechanisms: GCN employs symmetric normalization, SAGE employs mean aggregation, and GAT employs attention-based weights for aggregation.

3 PROBLEM FORMULATION

In this section, we define the scope and goal of our problem.

Learning setting. Given the prevalence of GNN models adhering to the transductive paradigm, our study exclusively revolves around transductive GNN models for which all graph nodes are available during training [17]. Furthermore, we consider node classification as the downstream task, where the GNN models output a Posterior Probability Vector (PPV) for each node.

Attack goal. We concentrate on the *Structure Membership Inference Attacks* (SMIAs) against GNNs. These attacks aim to deduce

[1] Our code and datasets are available at the link: https://gitfront.io/r/username/Ww17i3onZ2Ug/SMIA/

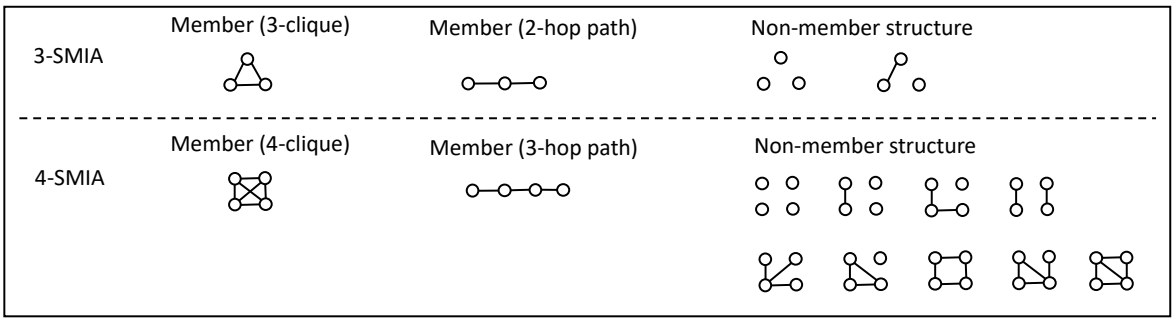


Figure 1: Member and non-member structures of 3-SMIA and 4-SMIA.

whether a set of nodes (attack target) constitute a particular structure within the training graph. Our focus centers on two fundamental subgraph structures: *k-hop paths* and *k-cliques*.

Adversary knowledge. We consider the adversary knowledge $\mathbb{K}$ along two dimensions:

- *Shadow graph* G^S: the adversary possesses a shadow graph G^S which contains its own structure and node features. G^S might originate from a distinct domain than G, featuring different data distribution as well.
- *Target model* Φ: we assume the adversary has only black-box access to Φ, which could be in the form of an API for Machine-Learning-as-a-Service platforms [40, 44]. The adversary can access the posterior classification probabilities of specific nodes when querying the target model.

It is worth noting that in practice, the adversary might possess knowledge of a partial graph which is a subset of the training graph [9, 20, 51]. However, we view this partial graph as a special case of the shadow graph. Additionally, the adversary might have a white-box access to GNN models , i.e., the adversary can access the GNN parameters such as gradients and node embeddings (e.g., in a Federated setting) [9, 51, 60]. The white-box access enables the adversary to derive attack features from GNN parameters, such as node embeddings, as opposed to using posterior probabilities as in black-box attacks. How to extend our attack to the white-box setting will be discussed in Section 5.6.

Problem definition. Given a graph $G(\mathcal{V}, \mathcal{E})$, a GNN model Φ trained on G, a set of k target nodes $V_{\text{att}} \subset \mathcal{V}$, and the adversary's background knowledge $\mathbb{K}$, the adversary's objective is to infer if V_{att} forms either a clique or a multi-hop path in G. The formal *k-node Structure Membership Inference Attack* (κ-SMIA) problem is defined as follows:

*Definition 3.1 (***K-node structure membership inference (κ-SMIA)***).* Given graph $G(V, E)$, GNN model Φ trained on G, and attack target comprising a k-node set $V_{\text{att}} = \{v_1, \ldots, v_k | v_1, \ldots, v_k \in V\}$, the κ-SMIA problem is formulated via the mapping function:

$$f : \Phi(v_1), \ldots, \Phi(v_k), \mathbb{K} \rightarrow \{0, 1, 2\}, \tag{2}$$

where label 0 denotes the absence of both structures within V_{att}, label 1 denotes that V_{att} constitutes a k-clique in G, and label 2 denotes that V_{att} lacks a k-clique but contains a (k-1)-hop path in G. Structures containing either a k-clique or a (k-1)-hop path are termed *member structures*, while the rest are labeled as *non-member structures*.

We do not assume that the adversary must possess any prior knowledge of the structure of G to make the selection of V_{att}. Figure 1 illustrates member and non-member structures for 3-SMIA and 4-SMIA. It is important to clarify that we exclude the last three non-member structures of 4-SMIA containing any 3-hop paths, as our consideration is confined to loop-free k-hop paths.

Notably, we solely infer the membership of the targeted structure, without delving into the specific subgraph structure composed by the target nodes within the training graph. As an example, we do not differentiate between the nine non-member structures within 4-SMIA (Figure 1). The extension from the Structure Membership Inference Attack (SMIA) to the Structure Inference Attack (SIA) will be discussed in Section 5.7.

It is noteworthy that both (k-1)-hop paths and k-cliques necessitate the complete utilization of the given k nodes. The (k-1)-hop path signifies the longest loop-free multi-hop path among the k nodes, while the k-clique represents the largest clique within these k nodes. Notably, we exclude structures with fewer nodes, such as k'-cliques and (k'-1)-hop paths with any $k' < k$, as these can be addressed by launching k'-SMIA, utilizing an input node set containing $k' < k$ nodes.

4 ATTACK DESIGN

Intuitively, a straightforward approach for performing an SMIA would involve adapting a link membership inference attack [20, 52] to deduce the link status for each node pair within the attack target node set V_{att}. With the predicted link statuses of all node pairs, it becomes possible to reconstruct the structure of V_{att} probabilistically, allowing the inference of whether a multi-hop path or a clique is present within the reconstructed structure. While conceptually sound, this method's effectiveness, as our empirical study (Section 5) will demonstrate, is limited by the accumulation of uncertainties of the predicted edges.

Attack overview. The intrinsic message-passing mechanism of GNN models leads to the nodes' prediction outputs heavily relying on their local neighborhoods. Distinct subgraph structures within these neighborhoods can induce unique behaviors among target nodes, which the attacks can leverage to differentiate between member and non-member structures. Built upon this insight, our SMIAs are designed to bypass the use of link membership inference attacks entirely. Instead, these SMIAs directly infer structure membership from the posterior probabilities of all nodes within the attack target.

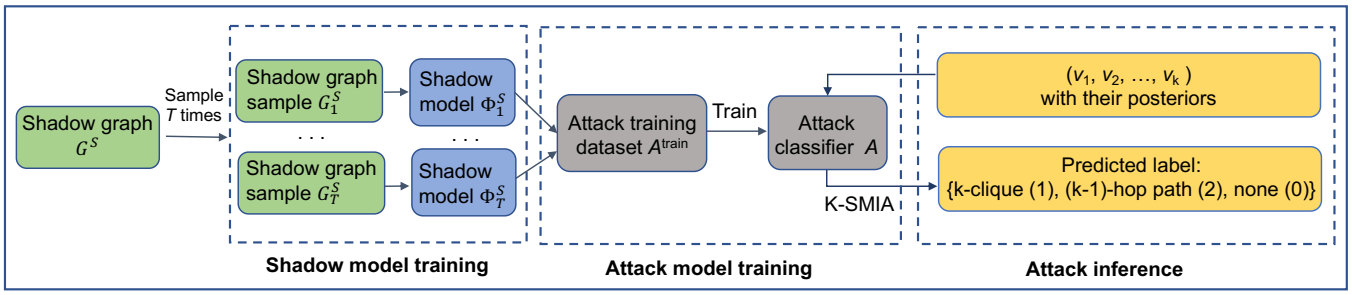


Figure 2: Overview of κ-SMIA.

In particular, we define our attack model $\mathcal{A}$ as a supervised classifier equipped with three labels (label definitions found in Definition 3.1). Training the 3-label classifier $\mathcal{A}$ necessitates obtaining labeled training data – the ground truth of membership of cliques and multi-hop paths. The primary challenge arises in generating such attack training data, particularly when the adversary lacks white-box access to the target GNN model. To address this challenge, we adopt the widely employed *shadow training technique*, utilized by existing link membership inference attacks [20, 44]. This involves training a set of shadow models to simulate the target model's behavior. It is important to note that, when the adversary possesses knowledge of a subset of the training graph, the attack model can be directly trained using the target model's predictions on those known samples [33, 34].

Figure 2 illustrates the framework of $\mathcal{A}$The attack is designed as a three-phase process described as below:

- **Shadow model training phase**: The adversary trains a set of shadow models on shadow graphs to mimic the behaviors of the target model.
- **Attack model training phase**: The adversary generates a training dataset A^{train} from the output of the shadow models on the shadow graph, and trains an attack classifier $\mathcal{A}$ on A^{train}.
- **Attack inference phase**: The adversary employs the trained classifier $\mathcal{A}$ to predict the membership status of the target node set V_{att}.

Next, we explain the details of each phase.

Shadow model training phase. In this phase, the adversary trains multiple shadow models, denoted as $\Phi_1^S, \ldots, \Phi_T^S$. The training dataset for each shadow model G_i^S is generated by random sampling a subset of the shadow graph G^S. To ensure an accurate emulation of the target model's behavior, we train the shadow models in the way that their performance closely matches that of the target model on the same training dataset. We achieve this by using the output of the target model as the input of the shadow models.

Attack model training phase. In this phase, the adversary constructs the attack training dataset, A^{train}. The generation of A^{train} follows four steps. The pseudo code of generation of A^{train} can be found in Appendix A. First, for each shadow graph G_i^S, the adversary selects a set of k-node sets $\mathbf{S}$. Each node set $V_s \in \mathbf{S}$ comprises k nodes randomly selected from G_i^S. Second, for each node set $V_s \in \mathbf{S}$, the adversary obtains the Posterior Probability Vector (PPV) for each node in V_s output by the shadow model Φ_i^S which was trained on the shadow graph G_i^S. There will be k PPVs for V_s. Third, these k PPVs are "aggregated" into a single vector which will serve as the attack feature vector $\mathbf{x}$. Specifically, the adversary measures the pairwise similarity of the k PPVs and obtains $\binom{k}{2}$ pairwise similarity values accordingly. Next, the adversary sorts these similarity values in ascending order, and concatenates the sorted values into a vector $\vec{v}$. We will demonstrate in Appendix B that the time cost associated with the sorting operation is negligible compared to that of training the attack classifier. To fully capture the distinction between different subgraph structures, we consider $d > 1$ multiple similarity metrics to measure the distance between the PPVs. In this paper, we consider three similarity metrics (i.e., $d = 3$), namely, *Dot product*, *Cosine similarity*, and *Euclidean distance*, that have been widely used for measurement of vector similarity. Therefore, there will be d sorted vectors accordingly, where each vector corresponds to a similarity metric. These d vectors are then concatenated into one vector, acting as the attack feature $\mathbf{x}$. Therefore, for any given k-node set V_s the dimension of its corresponding attack feature $\mathbf{x}$ is $\binom{k}{2} \times d$, where d is the number of similarity metrics. As the dimension of $\mathbf{x}$ is independent of the number of labels, this feature construction method enables the attack to be launched under the transfer setting where the shadow graph and the target graph originating from different domains are associated with different numbers of labels. After the adversary generates the feature $\mathbf{x}$ of the node set V_s, he associates $\mathbf{x}$ with its label $\mathbf{y}$. In particular, $\mathbf{y} = 1$ if V_s forms a k-clique in the shadow graph G_i^S, $\mathbf{y} = 2$ if V_s contains a (k-1)-hop path, and $\mathbf{y} = 0$ otherwise. Finally, the adversary adds the newly formed data sample $(\mathbf{x}, \mathbf{y})$ to A^{train}. In our empirical study, we ensure A^{train} is balanced, i.e., each class has the same number of samples.

After A^{train} is generated, the adversary proceeds to train the attack classifier $\mathcal{A}$ on A^{train}. In this paper, we consider three types of classifiers, namely Multi-layer Perceptron (MLP), Random Forest (RF), and Linear Regression (LR). Their performance will be presented later (Section 5).

Attack inference phase. At inference time, the adversary employs the same methodology as the generation of training dataset A^{train} to derive the feature $\mathbf{x}_{att}$ for the target node set V_{att}, utilizing the same similarity functions. It is important to note that, unlike the attack features of the training data V_s that uses the probability output by the shadow model, the adversary employs the posterior probability output of V_{att} by the target model to calculate $\mathbf{x}_{att}$.

Finally, the adversary feeds $\mathbf{x}_{att}$ into $\mathcal{A}$ to obtain predictions. The label associated with the highest probability will be selected as the inference output.

5 EVALUATION OF ATTACK PERFORMANCE

In this section, we assess the performance of SMIA. For simplicity, we focus on two specific input configurations of V_{att} in our experiments: (1) **3-SMIA** where V_{att} comprises three nodes, and the goal is to infer if V_{att} forms either a 3-clique (triangle) or a 2-hop path; (2) **4-SMIA** where V_{att} involves four nodes, intending to infer if V_{att} constitutes either a 4-clique or a 3-hop path.

5.1 Experimental Setup

All the experiments are executed on NVIDIA A100-PCIE-40GB. All the algorithms are implemented in Python along with PyTorch. All the reported results are averaged over 10 repetitions of the experiments.

Datasets. We use three real-world graph datasets, namely *Citeseer*, *Google+*, and *Lastfm* datasets, which are popularly used for graph learning [29, 53, 56]. Table 1 provides the statistical information of these datasets. More details of these datasets can be found in Appendix C.

Target GNN models. We consider three representative GNN models, namely GCN [26][2], SAGE [18], and GAT [49][3]. Each model is equipped with two hidden layers, with each hidden layer containing 64 neurons. We conduct training for 1,500 epochs with the early stopping tolerance value set as 50.

Shadow graphs. We consider two different settings of the shadow graph: (1) **Single-dataset setting** — both the shadow graph and the target graph are sampled from the same dataset; (2) **Data transfer setting** — the shadow graph and the target graph are drawn from different datasets, which may possess varying distributions and domains. For instance, the target dataset might be derived from the Citeseer dataset, which represents a citation graph, while the shadow dataset could originate from the Lastfm dataset, characterizing a social network graph of music users.

Shadow models. We consider two settings of the shadow model: (1) **Non-transfer setting** — both the shadow model and the target model have the same GNN architecture; (2) **Model transfer setting** — the shadow model and the target model have different architectures. For example, the target model and the shadow GNN model can be GCN and SAGE, respectively. For each shadow model, we randomly sampled a subgraph from the shadow graph as its training data, ensuring the performance of the shadow model is similar to that of the target model on the same shadow graph.

Attack classifier setup. In our experiments, we employ three types of distinct attack classifiers: *Multi-layer Perceptron (MLP)*, *Random Forest (RF)*, and *Support Vector Machine (SVM)*. The MLP model comprises three hidden layers, each consisting of 64, 32, and 16 neurons, respectively. We employ the Rectified Linear Unit (Relu) as the activation function for the hidden layers, while the Sigmoid function governs the output layer. Our training regimen spans 1,000 epochs, employing a learning rate of 0.001, the cross-entropy loss, and the Adam optimizer. For the RF classifier, we cap the maximum depth at 150. For the SVM classifier, we leverage the radial basis function (RBF) kernel that is coupled with a regularization parameter set to 1. Additionally, we set the degree of the polynomial kernel to three, and set the kernel coefficient γ as 1. We implemented the three classifiers using the sklearn package[4], a toolkit that is widely utilized for machine learning.

Attack training and testing data. First, we create a data pool that contains the labeled data samples. The data samples of each class are generated as follows:

Non-member samples (Class 0): We randomly sample a set of node sets from the given dataset, where each sampled node set includes neither a clique nor a $(k-1)$-hop path ($k = 3$ for 3-SMIA and $k = 4$ for 4-SMIA). These node sets are associated with the label 0 and are included in the data pool.

Member samples (Classes 1 and 2): We randomly sample a number of node sets (triples for 3-SMIA and quadruples for 4-SMIA) from the training graph, where each node set is either a clique or a k-hop path. Each node set is labeled with 1 if it is a clique in the training graph and 2 otherwise. These node sets are added to the data pool.

Once the data pool is constructed, we divide it into the attack training dataset A^{train} and the attack testing data set A^{test}, using a 7:3 split ratio. We ensure that both A^{train} and A^{test} are balanced, with an equal number of samples in all classes. Additionally, we guarantee an equal number of samples for all non-member structures in both A^{train} and A^{test}.

Metrics. Regarding the target model performance, we utilize the Area Under the Curve (AUC) for node classification.

As for assessing the attack's effectiveness, we employ the following three metrics to evaluate the performance of the attack classifier: (1) *Balanced accuracy (BA)*: BA is a well-known metric for multi-class classification. It is computed as the average of recalls of all the classes. Given our balanced testing data, BA tends to align closely with the classification accuracy, which is the ratio of correctly predicted samples to the total number of tested samples. (2) *Area Under the Curve (AUC)*: The AUC metric is measured over various threshold settings of the attack classifier, using the true positive rate (TPR) and false positive rate (FPR). (3) *True-Positive Rate at False-Positive Rates (TPR@FPR)*: This metric, as introduced in [6], measures the true positive rate when the false positive rate is set at a specific value. In our experiments, we set the FPR at 1%.

It is worth noting that our evaluation goes beyond the overall performance of the attack classifier; we also measure the performance of each individual class.

Baselines. We consider three baseline methods for comparison with our approach:

- **Baseline-1 (Ensemble of sub-attacks with single similarity metric).** This baseline employees an *ensemble* of three sub-attack classifiers, each utilizing a single similarity metric. The sub-attack features are derived from one of the three similarity metrics (Dot product similarity, Cosine similarity, and Euclidean distance) that are used by our approach. The prediction label for each testing sample is determined by majority voting among the labels generated by the three sub-attack classifiers.
- **Baseline-2 (Concatenation of posterior vectors).** In this baseline, the attack features are obtained by directly concatenating the posterior vectors of the nodes, rather than concatenating the similarity of posterior vectors as in our approach.

[2]We use the implementation of GCN available at https://github.com/tkipf/pygcn
[3]We use the implementation of both SAGE and GAT available at https://github.com/dmlc/dgl
[4]https://scikit-learn.org/

Dataset	Domain	Nodes	Edges	Classes	2-hop paths	3-cliques	3-hop paths	4-cliques
Google+	Social network	4,417	119,582	2	16,670,586	1,551,859	32,468,232	15,388,770
Lastfm	Social network	7,624	27,806	18	679,080	40,433	1,201,750	65,442
Citeseer	Citation network	3,312	4,732	7	27,174	1,547	185,706	514

Table 1: Description of datasets

- **Baseline-3 (Link inference attack).** Intuitively, whether a subgraph exists in the training graph can be accomplished by reconstructing the subgraph structure. As a subgraph can be reconstructed by inferring all of its links, we design Baseline 3 which employs a Link Inference Attack (LMIA) [21] for subgraph reconstruction. Specifically, first, we construct all the "possible worlds" of the subgraphs that can be constituted from the nodes in V_{att}. Each possible world W is constructed by randomly assigning a 0 or 1 value to each node pair $v_i, v_i \in V_{\text{att}}$, where 1 indicates that v_i and v_j are linked in W (denoted as $(v_i, v_j) \in W$), and 0 otherwise (denoted as $(v_i, v_j) \notin W$). Then for each possible world W of the subgraph, we compute its probability $Prob$ as:

$$Prob = \Pi_{(v_i,v_j)\in W} p_{i,j} \times \Pi_{(v_i,v_i)\notin W}(1 - p_{i,j}),$$

  where $p_{i,j}$ is the link prediction probability inferred by LMIA [21]. After we compute the probability of all the possible worlds, we pick the one with the highest probability as the inferred structure, and assign the membership label (0, 1, or 2) based on the structure of the picked one.

Attack setup. Among the three categories of attack classifiers explored in our experiments (MLP, SVM, RF), we observed that the MLP attack consistently exhibits the highest attack accuracy across most settings. Therefore, we primarily showcase the results obtained using the MLP attack classifier in the following discussions. The details regarding the performance of the three attack classifiers can be found in Appendix D.1.

Additionally, as we observed that the attacks leveraging the concatenation of all three similarity metrics (Cosine similarity, Dot Product, and Euclidean distance) outperform those employing either a single metric or the combination of any two metrics, we present the results of the attacks that concatenate the three similarity metrics in the following discussions. More details can be found in Appendix D.2.

We also observed that the number of shadow models does not impact the attack accuracy. Therefore, we used only one shadow model in the experiments. The results of attack performance for various number of shadow models can be found in Appendix D.3.

5.2 Attack Effectiveness

Before initiating the attacks, we assess the performance of the GNN models to confirm their suitability as targets. The outcomes reveal the strong performance of the three GNN models, with their AUC values ranging from 0.73 to 0.95. Detailed results are available in Appendix E. With the GNN models demonstrating satisfactory performance, they are deemed suitable for the ensuing attacks.

5.2.1 Non-transfer Setting. In this section, we present the attack performance for the non-transfer setting.

Overall attack performance. Figure 3 (a) - (c) presents the balanced accuracy (BA) of 3-SMIA. A glance at the results shows that the attack accuracy falls in the range [0.52, 0.75], which is substantially above the random guessing threshold of 0.33 across all settings. This demonstrates the efficacy of 3-SMIA against GNNs. Furthermore, 3-SMIA outperforms all three baseline methods in terms of BA across all settings. Notably, it significantly surpasses Baseline-3, whose BA ranges from 0.25 to 0.37, a largely unacceptable range. This provides strong evidence that the link membership inference attacks cannot be directly employed to handle SMIA. We attribute the suboptimal performance of Baseline-3 to its inability to effectively capture the high-order connectivity among nodes. While link membership inference attacks can infer first-order connectivity between nodes, extending this inference to higher-order connectivity proves challenging. The straightforward multiplication of link inference probabilities exacerbates the uncertainty inherent in first-order connectivity inference, resulting in significant errors when inferring the membership of subgraphs.

It is worth noting that even though the superiority of 3-SMIA over Baseline-1 and Baseline-2 in terms of attack performance is marginal, 3-SMIA still holds distinct advantages over both. First, Baseline-1 is more time-consuming than ours as it requires the training of three sub-attack classifiers, while ours only requires one. Furthermore, unlike our method, Baseline-2 cannot be applied in the transfer setting. This limitation arises because the features of the attack training dataset and attack testing dataset would have different dimensions if the target and shadow graphs have a different number of class labels.

Figures 3 (d) - (f) present the BA of 4-SMIA. Similar to 3-SMIA, 4-SMIA demonstrates considerable effectiveness, maintaining a BA range of 0.46 to 0.6, well above the random guessing threshold (0.33) in all scenarios. Moreover, 4-SMIA outperforms the three baseline methods in terms of attack accuracy in most of the settings, particularly excelling in comparison to Baseline-3.

Beyond BA, we measure both AUC and TPR@1%FPR of 3-SMIA and 4-SMIA. Further details can be found in Appendix F.1. We observe that both AUC and TPR@1%FPR results are remarkably high. For instance, the AUC of 3-SMIA falls within the range of 0.67 to 0.89, significantly surpassing the AUC of random guessing (0.5). Furthermore, both 3-SMIA and 4-SMIA consistently outperform the baseline in terms of both AUC and TPR@1%FPR across all settings.

Attack performance of individual classes. So far we have demonstrated the overall performance of the attack classifiers. Next, we take a closer look of the attack performance by assess the accuracy of individual classes.

Table 2 presents the balanced attack accuracy (BA) of both 3-SMIA and 4-SMIA for each class. We have the following observations.

2-hop path members: The 2-hop path class obtains the lowest accuracy for 3-SMIA. This can be attributed to the fact that when two out of the three given nodes are connected, they are likely to have similar prediction outputs, resulting in the third node being

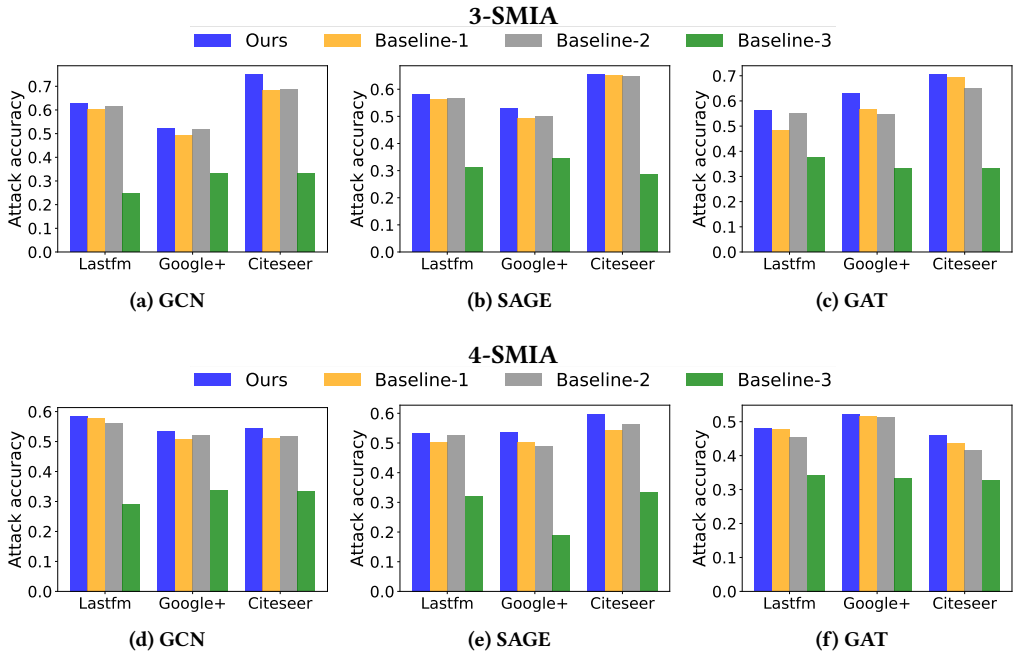


Figure 3: Balanced attack accuracy of 3-SMIA and 4-SMIA (Non-transfer setting).

Class	Lastfm			Google+		
	GCN	SAGE	GAT	GCN	SAGE	GAT
3-clique	0.67	0.61	0.64	0.59	0.52	0.68
2-hop path	0.37	0.44	0.35	0.36	0.39	0.5
Non-member	0.73	0.79	0.67	0.63	0.53	0.69

(a) 3-SMIA

Class	Lastfm			Google+		
	GCN	SAGE	GAT	GCN	SAGE	GAT
4-clique	0.8	0.68	0.75	0.72	0.82	0.6
3-hop path	0.62	0.43	0.51	0.49	0.42	0.49
Non-member	0.35	0.42	0.4	0.38	0.39	0.37

(b) 4-SMIA

Table 2: Balanced attack accuracy of individual classes (non-transfer setting). The highest and lowest attack accuracy for each GNN model and each class is marked with olive color and orange color respectively.

perceived as similar to both. This phenomenon reduces the accuracy of the 2-hop path class in 3-SMIA.

4-Clique members: The 4-clique class exhibits the highest accuracy for the 4-SMIA attack. This is because 4-cliques require all nodes to be connected, leading to the high similarity among all node pairs. Consequently, 4-cliques are more distinguishable and recognizable by 4-SMIA compared to the other two classes.

Non-members: The non-member class exhibits the highest accuracy for 3-SMIA across all the settings. However, it receives the lowest accuracy for 4-SMIA. This difference in behavior is likely due to the characteristics of the non-member structures for both attacks. In 3-SMIA, the non-member structures have at most one edge (and thus one pair of nodes with higher similarity), making them distinguishable from member structures. In 4-SMIA, some non-member structures have the same number of edges as member structures, making them less distinguishable from member structures based on the similarity of node pairs.

Beyond BA, we measure both AUC and TPR@1%FPR of member classes. In line with the results from BA, both 3-SMIA and 4-SMIA demonstrate superior AUC and TPR@1%FPR performance on k-cliques compared to k-hop paths. Further details and discussions can be found in Appendix F.1.

5.2.2 Transfer Setting. We consider two different transfer settings: (1) the **database transfer setting** where the target data and the shadow data are sampled from different datasets; and (2) the **model transfer setting** where the target model and shadow model are of different architecture. We report the attack performance for both transfer settings below.

Dataset transfer setting. Figure 4 (a) - (c) present the attack performance of 3-SMIA under the dataset transfer setting. The results of 4-SMIA under the dataset transfer setting can be found in Appendix F.2. The results demonstrate the attack's continued effectiveness under the data transfer setting. For instance, the attack AUC of 3-SMIA spans a range of 0.62 to 0.89 when the shadow graph is different from the target graph. Notably, even in scenarios where the target graph and the shadow graph originate from different domains, such as the target graph sampled from the Citeseer dataset and the shadow graph sampled from the Lastfm dataset, the AUC of 3-SMIA can be up to 0.83 (Figure 4 (a)). This demonstrates that 3-SMIA can learn the knowledge of distinguish member and non-member structures from the shadow graph and effectively transfer this acquired knowledge to the target graph.

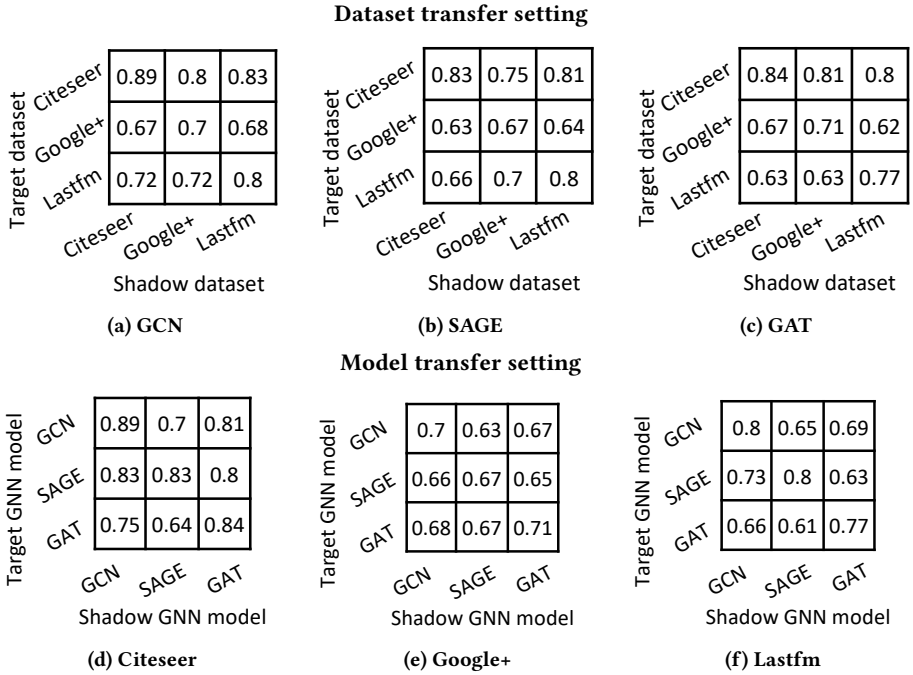


Figure 4: Attack AUC of 3-SMIA under dataset and model transfer setting.

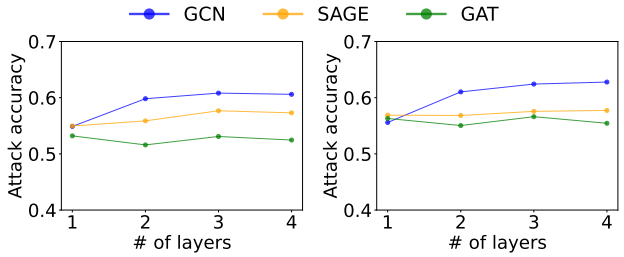


Figure 5: Imapct of GNN model complexity on attack accuracy of 3-SMIA and 4-SMIA (Lastfm dataset).

Model transfer setting. Figure 4 (d) - (f) illustrate the attack performance of 3-SMIA in the context of model transfer. Additionally, the results of 4-SMIA under the model transfer scenario can be found in Appendix F.2. Our observations indicate that the attack AUC of 3-SMIA varies from 0.61 to 0.83 when the shadow model differs from the target model across all the settings. These findings serve to highlight the continued effectiveness of our attack under the model transfer setting.

5.3 Impact of Model Complexity on Attack Performance

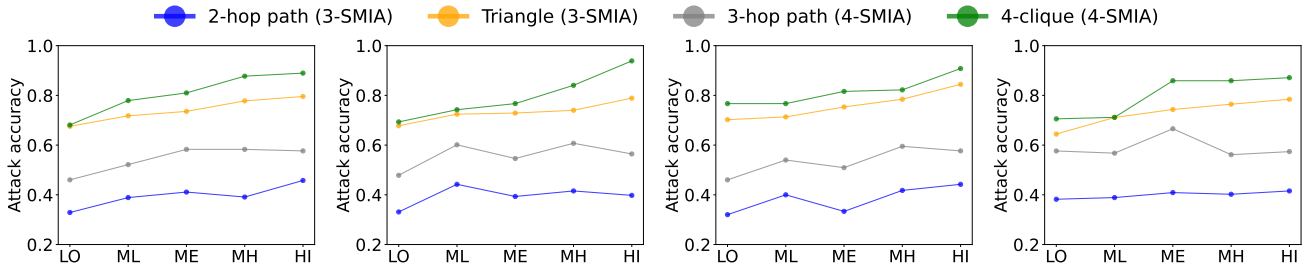
Given that message-passing GNNs capture information from neighboring nodes up to a maximum of K hops, where K corresponds to the number of GNN layers, an important question arises: *Does SMIA's performance vary with the number of GNN layers, given that subgraph membership inference essentially involves inferring neighborhoods within a specific number of hops?* To address this question, we deploy GNNs with varying levels of architectural complexity (i.e., different numbers of layers) and assess the attack performance of κ-SMIA for these GNNs. We manipulate the number of hidden layers from 1 to 4, with each layer containing 64 neurons.

Figure 5 illustrates the attack accuracy for GNN models with different complexities. For both 3-SMIA and 4-SMIA, the attack accuracy of GCN and SAGE undergoes a sharp increase as the number of layers goes from one to two. This is attributed to the fact that 2-layer GNN models encode the 2-hop neighborhood, enabling the attack classifiers to better distinguish between 2-hop paths and non-member structures. A similar trend of increasing attack accuracy persists when the number of layers increases from two to three, as it becomes more accurate at inferring 3-cliques. However, this trend stabilizes when the number of layers exceeds three.

Nevertheless, this pattern is not observed for the GAT model. The discrepancy may be attributed to the fundamental distinction in the neighborhood aggregation mechanism of GAT compared to the other two GNN models. While GCN and SAGE employ MEAN operations for neighborhood aggregation, GAT employs attention-based weights for aggregation. This attention-based mechanism reduces the accumulation effects of large neighborhoods, contributing to the differing behavior observed in the GAT model.

5.4 Impact of Node Importance on Attack Performance

Thus far, we have effectively demonstrated the efficacy of both 3-SMIA and 4-SMIA across all attack targets present in the testing data. However, are these attacks more accurate if the adversary has some prior knowledge of the target graph and thus targets specific nodes, such as those of higher importance (e.g., nodes of



(a) Node degree (b) Node closeness centrality (c) Node betweenness centrality (d) Node feature similarity

Figure 6: Impact of node degree, node closeness, betweenness centrality, and node feature similarity on attack accuracy of 3-SMIA and 4-SMIA (GCN, Lastfm dataset). "LO", "ML", "ME", "MH", "HI" represent the node group of low, medium low, medium, medium high, and high average node degree/closeness/betweenness centrality/feature similarity respectively.

high degrees), compared to nodes of lesser significance? In this part of experiments, we will investigate if our attacks have disparate performance of the target node sets that have distinct properties in their structure and features.

Node importance. We consider three metrics that are commonly employed for gauging node importance in graph analytics:

- **Node degree**: This metric quantifies the number of edges connected to a node.
- **Node closeness centrality**: This metric quantifies the number of the shortest paths that traverse through the node. Nodes with a higher closeness centrality score exhibit shorter distances to other nodes.
- **Node betweenness centrality**: This metric computes the reciprocal of the sum of shortest path lengths between the node in question and all other nodes in the graph. Nodes with higher betweenness centrality wield more control over the network, as more information passes through them.

Importance score of target node set. Given a node set V_{att}, we measure its *importance score* by four distinct metrics: *node degree importance*, *node closeness centrality importance*, *node betweenness centrality importance*, and *feature similarity importance*. The first three types of importance scores are measured as the average of the node degree, node closeness centrality, and node betweenness centrality of all nodes in V_{att}, and feature similarity importance score is measured as the average of the cosine similarity of node features for all node pairs in V_{att}.

In our assessment, we determine the importance score of each node set in A^{test}. Then we rank the node sets within A^{test} in ascending order of their importance scores and evenly partition these ranked samples into five distinct groups: *Low* (LO), *Medium Low* (ML), *Medium* (ME), *Medium High* (MH), and *High* (HI). For each group, we measure its attack accuracy as the average of the attack accuracy of all samples within the group.

Figure 6 illustrates the influence of four types of importance scores on the attack accuracy of 3-SMIA and 4-SMIA against the GCN model trained on the Lastfm dataset. Detailed outcomes for other settings can be found in Appendix F.3. We have several noteworthy observations. First, the accuracy of clique inference (triangle and 4-cliques) consistently increases for targets of higher importance scores for all the four types of importance. This signifies that SMIAs yield greater success when inferring cliques among nodes of heightened significance. This phenomenon is likely attributable to the target model's tendency to memorize more information from the more pivotal nodes, owing to the message-passing and neighborhood aggregation mechanisms intrinsic to GNNs. However, such a trend does not manifest for k-hop path inference. We guess that it is because the presence of disconnected node pairs within V_{att} introduces additional uncertainty into attack inference, potentially explaining the lack of a uniform pattern.

Dataset	6-SMIA			10-SMIA		
	GCN	SAGE	GAT	GCN	SAGE	GAT
Lastfm	0.54	0.52	0.48	0.49	0.45	0.44
Google+	0.51	0.48	0.50	0.47	0.46	0.45

Table 3: Balanced attack accuracy of 6-SMIA and 10-SMIA under Setting 1 (non-transfer setting). The balanced attack accuracy for random guess is 0.33.

5.5 Inference of Large Subgraphs

In this section, we assess the effectiveness of our attack on larger subgraphs, namely 6-SMIA and 10-SMIA whose attack target contains six and ten nodes, respectively. The three class labels of 6-SMIA correspond to a 6-clique, a 5-hop path, and any other structure, respectively, while the three class labels for 10-SMIA correspond to a 10-clique, a 9-hop path, and any other structure, respectively. The experimental setup is the same as that of 3-SMIA and 4-SMIA (Section 5.1).

Table 3 reports the balanced accuracy (BA) of 6-SMIA and 10-SMIA under the non-transfer setting. The results show that our attacks remain highly effective in inferring these larger subgraphs, achieving an attack accuracy ranging from 0.48 to 0.54 for 6-SMIA and between 0.44 and 0.49 for 10-SMIA. These results significantly outperform the baseline accuracy of 0.33, which represents a random guess scenario. This underscores the capability of our attack to successfully infer large subgraphs.

5.6 Extension to White-box Attacks

In this part of the empirical study, we expand the scope of SMIAs to the white-box setting, wherein the adversary can access the architecture and internal parameters of the target model. Unlike the black-box setting where the attack features are derived from the

Dataset	3-SMIA			4-SMIA		
	GCN	SAGE	GAT	GCN	SAGE	GAT
Lastfm	0.64	0.62	0.61	0.64	0.57	0.67
Google+	0.73	0.61	0.62	0.55	0.55	0.56
Citeseer	0.79	0.68	0.72	0.62	0.59	0.58

Table 4: Balanced attack accuracy of white-box attacks under Setting 1 (non-transfer setting).

Dataset	3-SIA			4-SIA		
	GCN	SAGE	GAT	GCN	SAGE	GAT
Lastfm	0.41	0.44	0.4	0.2	0.23	0.19
Google+	0.36	0.42	0.34	0.19	0.18	0.18
Citeseer	0.41	0.43	0.42	0.21	0.26	0.23

Table 5: Balanced attack accuracy of the Structure Inference Attack (SIA).

posterior probabilities, the attack features of the white-box attack will be derived from node embeddings.

Attack design. As the adversary under the white-box setting has direct access to the parameters of the target model, there is no need to utilize shadow models. Consequently, the white-box attacks involve only two primary phases: *attack model training* and *attack inference*. Both phases closely resemble the phases of attack model training and attack inference of our black-box approach (Section 4) with a minor modification: the attack features are now derived from the similarity of final node embeddings instead of the similarity of posterior outputs. Specifically, for a given similarity metric, the pairwise similarity values of all pairs of the nodes in the sampled graph are sorted in ascending order. There will be d sorted lists for d metrics. These d vectors are then concatenated into one vector, acting as the attack feature $\mathbf{x}$. We use the same three similarity metrics (Cosine similarity, Dot product, and Euclidean distance) as the black-box setting.

Evaluation. Table 4 presents the balanced attack accuracy of the white-box attack under Setting 1 (non-transfer setting). The results reveal that the accuracy of the white-box attack surpasses that of the black-box attack (Figure 3). For instance, when considering SAGE as the target model and utilizing the Citeseer dataset as the training data, the black-box 3-SMIA attack achieves an attack accuracy of up to 0.65 (Figure 3 (b)), while the white-box attack achieves an even higher attack accuracy of 0.68 (Table 4). The higher accuracy of the white-box attack is anticipated, as the attack can extract more information from the original training data through their access to node embeddings.

Beyond the non-transfer setting, we measure the attack AUC of the white-box attack under both data transfer and model transfer settings. The results can be found in Appendix F.4.

5.7 Extension to Structure Inference Attack (SIA)

Up to this point, we have demonstrated the effectiveness of our κ-SMIA approaches in inferring whether the given target nodes contain a clique or a multi-hop path (for instance, whether there exists a multi-hop path among the nodes V_A, V_B, V_C). However, in some cases, the adversary might seek to discover specific details of the structure itself (for example, whether the multi-hop path is structured as $V_A - V_B - V_C$ or $V_C - V_A - V_B$).

Unfortunately, the existing subgraph inference attack [60] cannot be directly applied to our setting as it utilizes the graph embedding (i.e., the embedding of the whole graph) but not posteriors of individual nodes to derive attack features. Thus, we extend the scope of SMIAs to the Structure Inference Attacks (SIAs), which aim to infer the subgraph formed by the target nodes in the training graph.

Attack design. Our SIA attack consists of two major steps:

- **Inference of number of edges**: the adversary first infers the number of edges in the subgraph of V_{att}.
- **Inference of edges**: based on the inferred number of edges, the adversary further infers the edges (and thus the structure) of V_{att}.

Next, we describe the details of each step.

Inference of number of edges. Intuitively, given k target nodes, there are $\binom{k}{2} + 1$ possibilities of the number of edges among them. Thus, we build a ℓ-class classifier to predict the number of edges among the given target nodes, where $\ell = \binom{k}{2} + 1$. This way, the 3-label classifier of κ-SMIA can be easily extended to κ-SIA. The training data A^{train} will be reconstructed to accommodate $\binom{k}{2} + 1$ classes for κ-SIA, while the features remain the same as κ-SMIA.

Inference of edges. Building upon the inferred number of edges from Step 1, the adversary proceeds to identify these edges. Specifically, for each node pair (v_i, v_j) of V_{att}, the adversary computes the similarity between the posterior probability vectors of v_i and v_j using a distance function. We use three distance functions (Dot product similarity, Cosine similarity, and Euclidean distance), which are the same ones used by SMIA. These node pairs are then ranked based on their similarity scores, and the top-t node pairs with the highest similarity values are selected, where t is the number of edges estimated by Step 1.

Compare with the link membership inference attack (LMIA) [20], our SIA attack is fundamentally different from [20]. First, LMIA and SIA have different attack goals. While LMIA aims to infer if two nodes were connected in the training graph, SIA aims to infer the specific structure among a set of $k > 2$ nodes. Second, the design of LMIA and SIA differs fundamentally. LMIA trains a binary classifier using features derived from the similarity between the posteriors of nodes for inference. In contrast, SIA follows a different approach: it first infers the number of edges t among the k target nodes and then selects the top t node pairs from the k nodes with the highest similarity between their posteriors as these edges, thereby constructing the inferred subgraph structure.

Empirical evaluation. Similar to SMIA, we investigate two variants of the SIA attacks: 3-SIA and 4-SIA which differ in the number of target nodes (3 or 4). We evaluate the performance using three distance functions: Dot product similarity, Cosine similarity, and Euclidean distance for Step 2. And we observe Cosine similarity yields the best attack performance among these functions. Hence we only report the results of using Cosine similarity. The training and testing data for the white-box attacks are generated in a similar way as to the black-box attacks, ensuring balance and a 7:3 split ratio between A^{train} and A^{test}.

Table 5 displays the attack accuracy of 3-SIA and 4-SIA. Notably, both attacks showcase substantial effectiveness, with attack accuracy ranging from [0.36, 0.44] for 3-SIA and [0.18, 0.26] for

4-SIA. These values are significantly higher than the random guess. Furthermore, SIAs demonstrate lower attack accuracy compared to SMIAs across all settings, as SIAs have to explicitly identify the subgraph structure itself, rather than inferring the inclusion of a structure only.

6 A POSSIBLE DEFENSE

To enhance the privacy of GNN models, a potential solution is to introduce noise into the models. Existing defense mechanisms against the link-level MIAs against GNNs typically inject noise into GNN parameters [1], or embeddings [43, 60], or posteriors [20]. However, our empirical evaluation (will be present in Section 6.3) demonstrates that these methods suffer from significant accuracy degradation in node classification tasks.

To address this challenge, we propose a novel noise mechanism to defend against SMIA. At a high level, our approach introduces noise into the final node embeddings generated by the target model. However, unlike existing methods that add noise across all dimensions of node embeddings [43, 60], our approach selectively adds noise only to dimensions of node embeddings that are considered least significant for the downstream task (node classification). By introducing noise selectively, our approach strikes a balance between maintaining target model accuracy and ensuring privacy protection against SMIAs.

In the following, we present the details of the defense. We begin with the measurement of feature importance (Section 6.1), followed by the discussion of our defense mechanism (Section 6.2). The results of the defense mechanism will be presented in Section 6.3.

6.1 SHAP-based Feature Importance Measurement

To measure the importance of embedding dimensions, we utilize Interpretable Machine Learning (IML) methods [2, 5, 12, 16, 30, 32] which can offer insights into the relevance of features for a model's performance. In our approach, we specifically employ the *SHapley Additive exPlanations (SHAP)* value-based method [30] to measure the importance of embedding dimensions.

SHAP is a technique used to interpret and explain predictions of individual data samples by quantifying the contribution of each feature to the predictions. The computed feature importance takes the form of *Shapley values* [30], where higher values indicate greater contribution (i.e., importance) to the model's output.

To employ SHAP for computing the importance of individual features in node embeddings, we treat the final node embeddings generated by the target model as the underlying features for the downstream task (i.e., node classification). We calculate the Shapley value for each dimension of the node embeddings and identify the dimensions with the top-ℓ lowest Shapley values as the *least important dimensions of the node embeddings*. In our empirical study, we experiment with different values of ℓ (in the format of perturbation ratio) and analyze their impact on defense performance. More details can be found in Section 6.3.

6.2 Details of Defense

Intuitively, while injecting noise into node embeddings (and thus the posterior probabilities) can deter SMIA's attack capabilities, random noise injection may lead to significant accuracy loss in GNN models. Therefore, the primary aim of our defense mechanism is to provide effective protection against SMIA while preserving the accuracy of the target model. To achieve this objective, we propose a perturbation-based strategy that introduces noise to the node embeddings, thereby altering the posterior probabilities and subsequently the attack features derived from these probabilities. However, indiscriminately adding noise to all dimensions of the node embeddings could result in a substantial loss in model accuracy. To mitigate this, we suggest adding noise only to the least important dimensions of the node embeddings. Even with noise introduced in these less important dimensions, node embeddings and their posterior probabilities can still undergo alteration. As will be demonstrated in our empirical study, these modifications can change the distribution of attack features between members and non-members, making them less distinguishable. Furthermore, targeting a subset of dimensions for noise injection helps minimize the degradation in model accuracy.

Our defense mechanism follows a three-step process. In Step 1, we compute the importance of each embedding dimension using SHAP (Section 6.1). We assume that the defender, possibly the model owner, has knowledge of the downstream task for conducting SHAP-based evaluation of dimension importance. Step 2 involves ranking the embedding dimensions in ascending order based on their SHAP values and selecting the top-ℓ dimensions from this ranking. Here, $\ell = \lfloor d \times r \rceil$, where d is the number of dimensions in the embedding and $r \in (0, 1]$ is the *perturbation ratio*. Increasing r perturbs more dimensions, resulting in a higher accuracy loss for the target model. In Step 3, we add noise to the dimensions chosen by Step 2. Specifically, we follow [43, 61] and introduce noise s that follows the Laplace distribution:

$$s = \frac{1}{2b} e^{-\frac{x-\mu}{b}}. \tag{3}$$

Here, b represents the noise scale, and μ is the location parameter of the Laplace distribution. A larger b signifies a more potent noise scale, leading to increased defense against SMIAs.

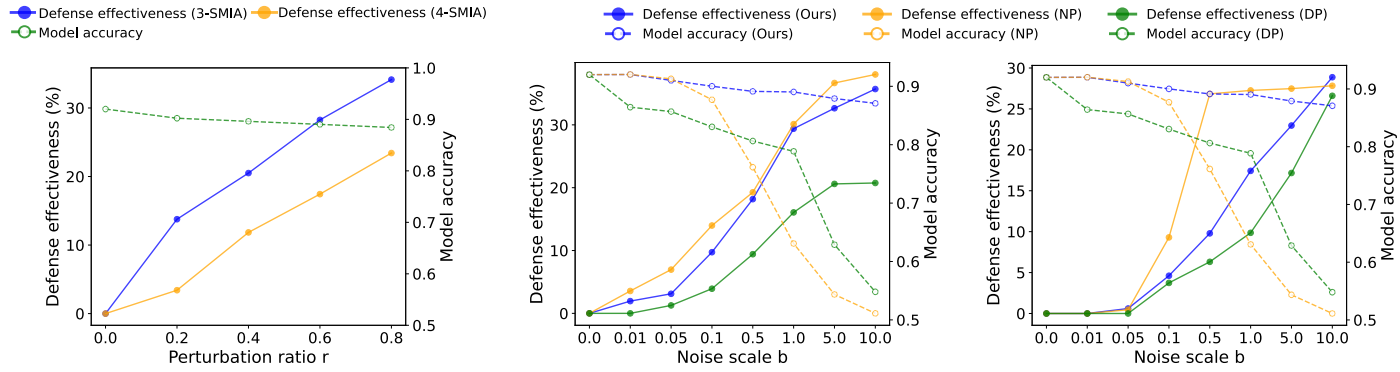
Unfortunately, our method cannot offer a formal privacy guarantee like Differential Privacy (DP) [10], as it does not utilize the concept of adjacency between datasets to determine the amounts of noise. Thus instead of theoretical analysis, we will empirically assess the defense effectiveness of our method. Our aim is to illustrate that our approach can provide comparable level of defense power to DP while minimizing the loss in model accuracy.

An alternative perturbation scheme is to add Gaussian noise instead of Laplace noise on the embeddings [39]. We assess both the defense effectiveness and model accuracy of the two noise schemes and found that both schemes have similar defense performance. The detailed results can be found in Appendix G.1. Therefore, we mainly employ Laplace noise in our paper.

6.3 Performance of Defense

In this section, we provide an assessment of the performance of our defense mechanism. We use the same GNN models and datasets as those for the attack evaluation (Section 5).

Setup. We configure the perturbation ratio $r = \{0.2, 0.4, 0.6, 0.8\}$ and noise scale $b = \{0.01, 0.05, 0.1, 0.5, 1, 5, 10\}$ in our experiments. Larger values of r and b indicate stronger privacy protection.



(a) Various perturbation ratio r (b=1) **(b) Various noise scale b for 3-SMIA (r=0.6)** **(c) Various noise scale b for 4-SMIA (r=0.6)**

Figure 7: Defense performance (GAT, Citeseer dataset). The solid and dotted lines in (b) and (c) denote the defense effectiveness and node classification AUC respectively.

Dataset	3-SMIA									4-SMIA								
	GCN			SAGE			GAT			GCN			SAGE			GAT		
	Ours	NP	DP	Ours	NP	DP	Ours	NP	DP	Ours	NP	DP	Ours	NP	DP	Ours	NP	DP
Lastfm	0.21	0.16	0.15	0.18	0.15	0.14	0.23	0.19	0.17	0.18	0.15	0.13	0.17	0.13	0.13	0.21	0.19	0.17
Google+	0.22	0.19	0.12	0.24	0.22	0.17	0.21	0.17	0.12	0.21	0.2	0.13	0.23	0.21	0.18	0.22	0.17	0.14
Citeseer	0.28	0.24	0.18	0.23	0.22	0.18	0.26	0.2	0.15	0.26	0.22	0.19	0.24	0.19	0.16	0.25	0.2	0.16

Table 6: Trade-off between defense effectiveness and the accuracy of the target model measured as the AUC of the defense-utility ROC curve. The best trade-off for each GNN model and each dataset is marked with olive color.

Metrics. We quantify the *defense effectiveness* as the percentage reduction in attack AUC achieved by the defense mechanism:

$$\text{Defense effectiveness} = \frac{AUC_{\text{before}} - AUC_{\text{after}}}{AUC_{\text{before}}} \tag{4}$$

Here, AUC_{before} and AUC_{after} represent the attack AUC before and after applying the defense, respectively. A higher reduction in attack AUC indicates stronger defense.

For the target model, we assess its accuracy using the AUC of node classification (Section 5.1). A higher AUC implies better accuracy for the target model.

Baselines. For comparative analysis, we compare our defense mechanism with the following two baselines:

- **Baseline-1: Differential privacy (DP).** We adopt DP-SGD, a state-of-the-art differentially private deep learning method [1], as our first baseline. To ensure a fair comparison between DP-SGD and our defense mechanism, we introduce Laplace noise to the gradients during each iteration, where the noise follows the same distribution as our defense mechanism (Eqn. (3)).
- **Baseline-2: Noisy posteriors (NP).** Since attack features stem from the target model's posteriors, we consider the defense method [20, 60] that introduces noise to the posterior probabilities as our second baseline. Specifically, for any given node and its associated posterior probability vector output by the target model, we incorporate Laplace noise into each posterior in the vector, using the same distribution as ours (Eqn. (3)) for the noise distribution.

Defense effectiveness. Figure 7 (a) (the y-axis at left) presents the attack AUC after the deployment of our defense mechanism with various perturbation ratios when GAT is used as the target model. The performance results for the other GNN models can be found in Appendix G.2. We did not compare our method with the two baselines as they have a fixed perturbation ratio of 1 (i.e., perturbation on all dimensions). We observe that our defense can effectively reduce the attack AUC. Remarkably, even with the perturbation ratio as small as 0.2 (i.e., 20% of embedding dimensions are perturbed), the accuracy of 3-SMIA is reduced by 14.3% (Figure 7 (a)). The defense power becomes stronger as the perturbation ratio r increases.

Why adding noise only to the less important dimensions of node embeddings still enable effective privacy protection? To answer this question, we visualize the distribution of attack features for each class before and after applying our defense method by using the t-SNE projection. The results are presented in Appendix G.3. We notice a significant increase in the indistinguishability of the three classes after introducing noise to the embedding. This observation highlights that, despite being added solely to the least important dimensions of the embedding, the noise has led to alterations in the posterior probabilities and their corresponding similarities. Consequently, it has effectively modified the distribution of attack features across all classes, thereby reducing the accuracy of the attack.

Next, we compare the performance of our defense method with the two baselines under various noise scales (b), and present the results in Figure 7 (b) & (c) (y-axis at right). First, the defense effectiveness of our method increases with a higher noise scale value. As the noise scale becomes as large as 10, our approach exhibits comparable defense effectiveness to NP. This observation aligns with our expectations, as the NP defense technique directly introduces noise

to the posteriors, while our method adds noise selectively to a subset of dimensions of the embeddings. Second, our approach showcases the capability to surpass the DP baseline in terms of defense effectiveness, as illustrated in Figure 7 (b). This can be attributed to the fact that, although Laplace noise is integrated into the gradient, the DP baseline continues to employ the unaltered adjacency matrix during node embedding computations. Consequently, GNN models still encapsulate some underlying structural information within node representations and the subsequent posteriors. This explains why our method can outperform the DP baseline, despite solely introducing noise to a subset of embedding dimensions.

Model accuracy. Figure 7 (a) (the y-axis at right) presents the model accuracy (node classification AUC) under various perturbation ratios. We observe an insignificant amounts of model accuracy loss (up to 0.04%) even when the perturbation ratio increases to 0.8. Furthermore, Figure 7 (b) & (c) (the y-axis at right) compare the model accuracy of our defense solution and the two baselines when the noise scale b increases. Our method outperforms both baselines in terms of model accuracy when b exceeds 0.1.

defense-utility trade-off. To assess the trade-off between defense effectiveness and target model accuracy, we establish a range of settings that utilizes a consistent perturbation ratio ($r = 0.4$) and varying noise scale values $b = \{0.01, 0.05, 0.1, 0.5, 1, 5, 10\}$. Equivalent noise scales are employed for the other two baseline methods. Subsequently, we gauge defense effectiveness and target model AUC across each configuration. These collected outcomes are then synthesized into a *defense-utility ROC curve*, whereby individual points symbolize paired values of defense effectiveness and target model accuracy. We quantify the *defense-utility trade-off* as the Area Under the Curve (AUC) of the defense-utility ROC curve. A higher AUC indicates a more favorable trade-off between defense effectiveness and target model accuracy. The defense-utility ROC curve can be found in Appendix G.4.

Table 6 presents the results of the defense-utility trade-off for both 3-SMIA and 4-SMIA. Our defense mechanism consistently manifests a superior trade-off in comparison to the baseline methods across all settings. This is expected as our method solely introduces noise to the least influential embedding dimensions.

Additionally, we assess the trade-off between defense effectiveness and utility using TPR@1%FPR as the attack accuracy metric. We construct a defense-utility ROC curve, where each point represents paired values of defense effectiveness and target model accuracy. The trade-off between defense and utility is quantified as the Area Under the Curve (AUC) of the defense-utility ROC curve. The results of the defense-utility ROC curve and the defense-utility trade-off are presented in Appendix G.4. We observe that our defense exhibits a superior trade-off compared to the baseline methods across most the settings.

7 RELATED WORK

Membership inference attacks against machine learning models. Recent studies [7, 46] have shown that ML models are vulnerable to various types of privacy attacks. Among these attacks, the *membership inference attack* (MIA) aims to infer whether a data record was used to train a target model or not. It was first studied in the context of genomics privacy [23, 41] and mobility privacy [38]. Shokri et al. [44] was the first work to apply MIA against machine learning. Since then, many endeavors have investigated MIAs in scope and depth. For example, Yeom et al. [55] analyzed the connection between MIAs and model overfitting. Salem et al. [40] proposed a generic attack that relaxes some assumptions of MIAs. Ye et al. [54] formally expressed MIAs under a comprehensive hypothesis testing framework. A line of work designed new MIAs for various ML models such as federated learning [34], generative models [19], language models [45, 47], and contrastive models [28]). And Meanwhile, a large body of work proposes different membership inference defenses to counter the threat of MIAs. We recommend the audience a number of surveys [3, 24, 27] for further reading.

Membership inference attacks against GNNs. A line of work has investigated the vulnerability of GNNs against MIAs [9, 20, 22, 36, 52]. The existing MIAs on GNNs can be categorized into three types: *node-level MIAs*, *link-level MIAs*, and *subgraph-level MIAs*. The node-level MIAs [22, 36] aim to infer the existence of particular nodes in the training graph, The subgraph-level MIA (SIA) [60] aims to infer if a subgraph is included in the target graph. However, as SIA utilizes the graph embedding (i.e., the embedding of the whole graph) not posteriors of individual nodes to derive attack features, it cannot be directly applied to our setting. To the best of our knowledge, ours is the first to investigate MIAs in the context of inference of cliques and multi-hop paths.

8 CONCLUSION

In this paper, we investigated the privacy vulnerabilities of GNN models through the lens of *Structure Membership Inference Attacks* (SMIAs). We introduced novel black-box SMIAs capable of predicting the presence of k-cliques and k-hop paths within a designated set of target nodes in the training graph. Our empirical study demonstrates the effectiveness of SMIAs against three representative GNN models. Moreover, we designed a new defense mechanism to counteract GNN susceptibility to SMIAs. Through empirical analysis, we showcased that our defense approach provides strong defense against SMIA while maintaining target model accuracy.

Looking ahead, we will explore SMIAs' effectiveness across diverse adversary knowledge settings, including scenarios with white-box access to the target model and availability of node features. We will also extend the scope of SMIAs to the inference of other graph structures, such as strongly connected components and stars. These extensions will contribute to a comprehensive understanding of GNN privacy and open up new avenues for safeguarding graph data against potential threats.

ACKNOWLEDGEMENT

We thank the anonymous reviewers for their feedback. This project was supported by the National Science Foundation (#CNS-2029038; #CNS-2135988). Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agency.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 308–318, 2016.

[2] André Altmann, Laura Tološi, Oliver Sander, and Thomas Lengauer. Permutation importance: a corrected feature importance measure. *Bioinformatics*, 26(10):1340–1347, 2010.

[3] Yang Bai, Ting Chen, and Mingyu Fan. A survey on membership inference attacks against machine learning. *management*, 6:14, 2021.

[4] Jörn Birkmann. Measuring vulnerability to promote disaster-resilient societies: Conceptual frameworks and definitions. *Measuring vulnerability to natural hazards: Towards disaster resilient societies*, 1(9):3–7, 2006.

[5] L Breiman. Random forests machine learning. 45: 5–32. *View Article PubMed/NCBI Google Scholar*, 2001.

[6] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *Proceedings of the Conference on Symposium on Security and Privacy*, pages 1897–1914, 2022.

[7] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium*, pages 267–284, 2019.

[8] Marco d'Errico, Francesca Grazioli, and Rebecca Pietrelli. Cross-country evidence of the relationship between resilience and the subjective perception of well-being and social inclusion: Evidence from the regions of matam (senegal) and the triangle of hope (mauritania). *Journal of International Development*, 30(8):1339–1368, 2018.

[9] Vasisht Duddu, Antoine Boutet, and Virat Shejwalkar. Quantifying privacy leakage in graph embedding. In *Proceedings of 17th EAI International Conference on Mobile and Ubiquitous Systems*, pages 76–85, 2020.

[10] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[11] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation. In *The world wide web conference*, pages 417–426, 2019.

[12] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[13] Chen Gao, Xiang Wang, Xiangnan He, and Yong Li. Graph neural networks for recommender system. In *Proceedings of the International Conference on Web Search and Data Mining*, pages 1623–1625, 2022.

[14] Chen Gao, Yu Zheng, Nian Li, Yinfeng Li, Yingrong Qin, Jinghua Piao, Yuhan Quan, Jianxin Chang, Depeng Jin, Xiangnan He, et al. A survey of graph neural networks for recommender systems: Challenges, methods, and directions. *ACM Transactions on Recommender Systems*, 1(1):1–51, 2023.

[15] Javier O Garcia, Arian Ashourvan, Sarah Muldoon, Jean M Vettel, and Danielle S Bassett. Applications of community detection techniques to brain graphs: Algorithmic considerations and implications for neural function. *Proceedings of the IEEE*, 106(5):846–867, 2018.

[16] I Guyou, J Weston, S Barnhill, and V Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.

[17] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[18] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the Conference on Neural Information Processing Systems*, 2018.

[19] Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. Logan: Membership inference attacks against generative models. *In Proceedings of the Privacy Enhancing Technologies (PoPETs)*, 2017.

[20] Xinlei He, Jinyuan Jia, Michael Backes, Neil Zhenqiang Gong, and Yang Zhang. Stealing links from graph neural networks. In *Proceedings of the 30th USENIX Security Symposium*, pages 2669–2686, 2021.

[21] Xinlei He, Jinyuan Jia, Michael Backes, Neil Zhenqiang Gong, and Yang Zhang. Stealing links from graph neural networks. In *USENIX Security Symposium*, pages 2669–2686, 2021.

[22] Xinlei He, Rui Wen, Yixin Wu, Michael Backes, Yun Shen, and Yang Zhang. Node-level membership inference attacks against graph neural networks. *arXiv preprint arXiv:2102.05429*, 2021.

[23] Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V Pearson, Dietrich A Stephan, Stanley F Nelson, and David W Craig. Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS Genet*, 4(8):e1000167, 2008.

[24] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S Yu, and Xuyun Zhang. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys (CSUR)*, 54(11s):1–37, 2022.

[25] Shuting Jin, Xiangxiang Zeng, Feng Xia, Wei Huang, and Xiangrong Liu. Application of deep learning methods in biological networks. *Briefings in bioinformatics*, 22(2):1902–1917, 2021.

[26] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2017.

[27] Bo Liu, Ming Ding, Sina Shaham, Wenny Rahayu, Farhad Farokhi, and Zihuai Lin. When machine learning meets privacy: A survey and outlook. *ACM Computing Surveys (CSUR)*, 54(2):1–36, 2021.

[28] Hongbin Liu, Jinyuan Jia, Wenjie Qu, and Neil Zhenqiang Gong. Encodermi: Membership inference against pre-trained encoders in contrastive learning. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pages 2081–2095, 2021.

[29] Yixin Liu, Ming Jin, Shirui Pan, Chuan Zhou, Yu Zheng, Feng Xia, and Philip Yu. Graph self-supervised learning: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[30] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[31] Shengjie Min, Zhan Gao, Jing Peng, Liang Wang, Ke Qin, and Bo Fang. Stgsn—a spatial–temporal graph neural network framework for time-evolving social networks. *Knowledge-Based Systems*, 214:106746, 2021.

[32] Saumitra Mishra, Bob L Sturm, and Simon Dixon. Local interpretable model-agnostic explanations for music content analysis. In *ISMIR*, volume 53, pages 537–543, 2017.

[33] Milad Nasr, Reza Shokri, and Amir Houmansadr. Machine learning with membership privacy using adversarial regularization. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 634–646, 2018.

[34] Milad Nasr, Reza Shokri, and Amir 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)*, pages 739–753, 2019.

[35] Iyiola E Olatunji, Anmar Hizber, Oliver Sihlovec, and Megha Khosla. Does black-box attribute inference attacks on graph neural networks constitute privacy risk? *arXiv preprint arXiv:2306.00578*, 2023.

[36] Iyiola E Olatunji, Wolfgang Nejdl, and Megha Khosla. Membership inference attack on graph neural networks. In *Proceedings of the 3rd IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, pages 11–20, 2021.

[37] Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *Proceedings of the international conference on World Wide Web*, pages 201–210, 2007.

[38] Apostolos Pyrgelis, Carmela Troncoso, and Emiliano De Cristofaro. Knock knock, who's there? membership inference on aggregate location data. In *Network and Distributed Systems Security (NDSS) Symposium*, 2018.

[39] Sina Sajadmanesh, Ali Shahin Shamsabadi, Aurélien Bellet, and Daniel Gatica-Perez. Gap: Differentially private graph neural networks with aggregation perturbation. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 3223–3240, 2023.

[40] Ahmed Salem, Yang Zhang, Mathias Humbert, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246*, 2018.

[41] Sriram Sankararaman, Guillaume Obozinski, Michael I Jordan, and Eran Halperin. Genomic privacy and limits of individual detection in a pool. *Nature genetics*, 41(9):965–967, 2009.

[42] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[43] Yun Shen, Yufei Han, Zhikun Zhang, Min Chen, Ting Yu, Michael Backes, Yang Zhang, and Gianluca Stringhini. Finding mnemon: Reviving memories of node embeddings. *arXiv preprint arXiv:2204.06963*, 2022.

[44] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE symposium on security and privacy (SP)*, pages 3–18, 2017.

[45] Congzheng Song and Ananth Raghunathan. Information leakage in embedding models. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 377–390, 2020.

[46] Congzheng Song, Thomas Ristenpart, and Vitaly Shmatikov. Machine learning models that remember too much. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 587–601, 2017.

[47] Congzheng Song and Vitaly Shmatikov. Auditing data provenance in text-generation models. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 196–206, 2019.

[48] Anshuman Suri and David Evans. Formalizing and estimating distribution inference risks. *arXiv preprint arXiv:2109.06024*, 2021.

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

# node sets	5,000	10,000	50,000	100,000
Time cost of sorting	0.0034s	0.0082s	0.032s	0.053s
Time cost of training a attack classifier	5.41s	17.97s	32.82s	103.8s

Table 7: Comparison of time cost (in seconds) between similarity sorting and attack classifier training.

[50] Meng Wang, Chaokun Wang, Jeffrey Xu Yu, and Jun Zhang. Community detection in social networks: an in-depth benchmarking study with a procedure-oriented framework. *Proceedings of the VLDB Endowment*, 8(10):998–1009, 2015.

[51] Xiuling Wang and Wendy Hui Wang. Group property inference attacks against graph neural networks. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security*, pages 2871–2884, 2022.

[52] Fan Wu, Yunhui Long, Ce Zhang, and Bo Li. Linkteller: Recovering private edges from graph neural networks via influence analysis. In *Proceedings of IEEE Symposium on Security and Privacy (S&P)*, 2022.

[53] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[54] Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. Enhanced membership inference attacks against machine learning models. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pages 3093–3106, 2022.

[55] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *IEEE 31st computer security foundations symposium (CSF)*, pages 268–282, 2018.

[56] Daokun Zhang, Jie Yin, Xingquan Zhu, and Chengqi Zhang. Network representation learning: A survey. *IEEE transactions on Big Data*, 6(1):3–28, 2018.

[57] Shijie Zhang, Hongzhi Yin, Tong Chen, Zi Huang, Lizhen Cui, and Xiangliang Zhang. Graph embedding for recommendation against attribute inference attacks. In *Proceedings of the Web Conference*, pages 3002–3014, 2021.

[58] Si Zhang, Dawei Zhou, Mehmet Yigit Yildirim, Scott Alcorn, Jingrui He, Hasan Davulcu, and Hanghang Tong. Hidden: hierarchical dense subgraph detection with application to financial fraud detection. In *Proceedings of the SIAM international conference on data mining*, pages 570–578, 2017.

[59] Wanrong Zhang, Shruti Tople, and Olga Ohrimenko. Leakage of dataset properties in {Multi-Party} machine learning. In *Proceedings of the USENIX Security Symposium*, pages 2687–2704, 2021.

[60] Zhikun Zhang, Min Chen, Michael Backes, Yun Shen, and Yang Zhang. Inference attacks against graph neural networks. In *Proceedings of the USENIX Security Symposium*, pages 1–18, 2022.

[61] Zhikun Zhang, Min Chen, Michael Backes, Yun Shen, and Yang Zhang. Inference attacks against graph neural networks. In *Proceedings of the 31th USENIX Security Symposium*, pages 1–18, 2022.

[62] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020.

APPENDIX

A PSEUDO CODE OF OUR ATTACK

Algorithm 1 shows the pseudo code of our attack.

B TIME COST OF SIMILARITY SORTING FOR ATTACK FEATURE GENERATION

Recall that our attack feature derivation necessitates sorting probability similarity values, raising concerns about potential computational overhead. To investigate this, we measured the time required for sorting probability similarities during attack feature derivation and compared it with the time needed for attack classifier training. The results are presented in Table 7, revealing that the sorting time is minimal compared to attack classifier training. In other words, sorting probability similarity values does not impose significant computational overhead on the training of the attack classifier.

Algorithm 1: Generating attack training data A^{train}

Input: A set of shadow graphs $G^S = \{G_1^S, G_2^S, ...\}$, the target node set V_{att}, the number of node sets in each class N.

Output: Attack training data A^{train}.

A^{train}={};
S = {};
Initialize the membership label y=0;
for *each shadow graph* G_i^S *in* G^S **do**
  Train a shadow model Φ_i^S on G_i^S;
  Randomly sample $N > 1$ k-cliques and add to S;
  Randomly sample $N > 1$ (k-1)-hop paths and add to S;
  Randomly sample $N > 1$ node sets that are neither k-cliques nor (k-1)-hop paths and add to S;
  for *each k-node set* V_s *in* S **do**
    $\mathbf{x}$ =<> //$\mathbf{x}$: the feature vector of V_s;
    for *each similarity metric* f_s **do**
      Initialize $\vec{v}$ as an empty vector;
      for $\forall v_i, v_j \in V_s$ **do**
        Add $f_s(\Phi_i^S(v_i), \Phi_i^S(v_j))$ to $\vec{v}$;
      end
      Sort all values in $\vec{v}$ in descending order ;
      $\mathbf{x} = \mathbf{x}||\vec{v}$;
    end
    Assign the membership label y of V_s: y=1 if V_s is a clique, y=2 if V_s is a (k-1)-hop path, and y=0 otherwise;
    Add the sample $(\mathbf{x}, y)$ to A^{train};
  end
end
return Attack training set A^{train}.

C DETAILS OF DATASETS

We conduct our experiments on three datasets, each possessing its own unique characteristics:

- **Google+** dataset [5]: The graph consists of 4,417 nodes and 119,582 edges in total. Each node is associated with a set of features including gender, institution, job title, last name, place, and university.
- **Lastfm** dataset [6]: The Lastfm dataset represents a social network of users of the Lastfm music platform. The edges in the dataset indicate the mutual follower relationships between users.
- **Citeseer** dataset: The Citeseer dataset is a citation network where the nodes represent scientific documents, and the edges represent citation links between documents. It consists of 3,312 nodes and 4,732 citation links.

[5] https://snap.stanford.edu/data/ego-Gplus.html
[6] http://snap.stanford.edu/data/feather-lastfm-social.html

Attack Settings	3-SMIA			4-SMIA		
	MLP	RF	SVM	MLP	RF	SVM
Lastfm	0.62	0.61	0.58	0.59	0.59	0.55
Google+	0.52	0.5	0.5	0.53	0.5	0.5
Citeseer	0.75	0.75	0.73	0.54	0.53	0.52

Table 8: Impact of type of attack classifiers on attack accuracy (GCN model). The best attack accuracy for each dataset and each attack classifier is highlighted with olive color.

Sim. metric			3-SMIA			4-SMIA		
C	D	E	GCN	SAGE	GAT	GCN	SAGE	GAT
✓			0.56	0.48	0.56	0.53	0.43	0.47
	✓		0.61	0.55	0.48	0.58	0.4	0.37
		✓	0.61	0.57	0.53	0.59	0.52	0.46
✓	✓		0.6	0.54	0.56	0.58	0.43	0.46
	✓	✓	0.6	0.57	0.55	0.59	0.52	0.47
✓		✓	0.61	0.57	0.51	0.59	0.52	0.45
✓	✓	✓	0.62	0.58	0.56	0.59	0.53	0.48

Table 9: Balanced attack accuracy (BA) for various combinations of similarity metrics. "C", "D", "E" represent Cosine similarity, Dot Product, and Euclidean distance respectively. The best attack performance for each GNN model is highlighted in olive color. (Lastfm dataset).

D ATTACK PERFORMANCE UNDER VARIOUS ATTACK SETTINGS

D.1 Varying Type of Attack Classifier

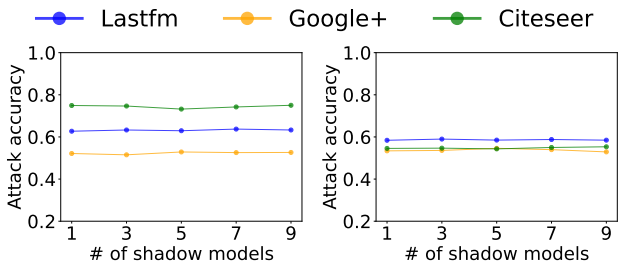
To evaluate the impact of the type of attack classifiers on the performance attacks, we evaluate the attack performance by three types of attack classifiers, namely, *Multi-layer Perceptron (MLP)*, *Random Forest (RF)*, and *Support Vector Machine (SVM)*. Table 8 presents the attack accuracy of the three attack classifiers when GCN model is used as the target model. All the three classifiers exhibit their effectiveness, with their attack accuracy exceeding the random guess (0.5). In particular, MLP outperforms RF and SVM consistently. Therefore, in our experiments, we mainly use MLP as the attack classifier.

D.2 Varying Similarity Metric(s)

To measure the impact of various similarity metrics on attack performance, we consider all the seven combinations of the three similarity metrics (Cosine similarity, Dot product similarity, and Euclidean distance), and measure the balanced accuracy of 3-SMIA and 4-SMIA under each combination. As reported in Table 9, the attack leveraging the concatenation of all three similarity metrics consistently outperforms or achieves the same attack performance as those employing either a single metric or the combination of any two metrics. Therefore, we adopt the concatenation of Cosine similarity, Dot Product, and Euclidean distance to generate the attack features.

D.3 Varying the Number of Shadow Models

Figure 8 presents the attack accuracy (BA) for various number of shadow models. The main observation is that the number of shadow models does not impact the attack accuracy, which remains stable when the number of shadow models varies. Therefore, we use only one shadow model in our experiments.



(a) 3-SMIA **(b) 4-SMIA**

Figure 8: Impact of the number of shadow models on attack accuracy (BA), GCN model.

Dataset	GCN	SAGE	GAT
Google+	0.93	0.93	0.95
Lastfm	0.75	0.76	0.73
Citeseer	0.94	0.95	0.92

Table 10: Node classification AUC performance of the GNN models.

E TARGET MODELS: SETUP AND PERFORMANCE

We assess the GNN models' performance through the measurement of node classification AUC. The results are presented in Table 10. It is evident that across all three datasets, the AUC values for the three GNN models are significantly superior to random classification, spanning the range of [0.73, 0.95]. This observation demonstrates the commendable performance of the target GNN models.

F ADDITIONAL RESULTS OF ATTACK PERFORMANCE

F.1 Non-transfer Setting

In this part of empirical evaluation, we measure both AUC and TPR@1%FPR of 3-SMIA and 4-SMIA. We also include the results of Baseline-2 for comparison. We did not consider Baseline-1 and Baseline-3 because their AUC and TPR@1%FPR are not measurable as they do not use a threshold.

AUC performance of attack classifier. Table 11 shows the AUC performance of 3-SMIA and 4-SMIA under Setting 1 (non-transfer setting). We have the following observations. First, the AUC values of our attacks across all settings consistently surpass the random guess threshold of 0.5. Specifically, the attack AUC of 3-SMIA ranges from 0.67 to 0.89, while the attack AUC of 4-SMIA ranges from 0.69 to 0.8. Second, our attacks outperform Baseline-2 in terms of attack AUC in all the settings.

TPR@1%FPR performance of attack classifier. Table 12 shows the TPR@1%FPR performance of 3-SMIA and 4-SMIA. The TPR@1%FPR of 3-SMIA ranges from 0.08 to 0.41, and the attack TPR@1%FPR of 4-SMIA ranges from 0.08 to 0.21. Furthermore, both

Dataset	3-SMIA						4-SMIA					
	GCN		SAGE		GAT		GCN		SAGE		GAT	
	Ours	Baseline-2	Ours	Baseline-2	Ours	Baseline-2	Ours	Baseline-2	Ours	Baseline-2	Ours	Baseline-2
Lastfm	0.80	0.79	0.71	0.69	0.73	0.72	0.79	0.73	0.72	0.71	0.69	0.67
Google+	0.70	0.67	0.67	0.65	0.71	0.70	0.74	0.72	0.71	0.70	0.70	0.68
Citeseer	0.89	0.84	0.83	0.82	0.84	0.81	0.80	0.74	0.76	0.72	0.71	0.69

Table 11: Attack AUC of 3-SMIA and 4-SMIA under Setting 1 (non-transfer setting). We did not consider Baseline-1 and Baseline-3 because their AUC is not measurable as they do not use a threshold. The best attack AUC for each dataset and each GNN model is highlighted with olive color.

Dataset	3-SMIA						4-SMIA					
	GCN		SAGE		GAT		GCN		SAGE		GAT	
	Ours	Baseline-2	Ours	Baseline-2	Ours	Baseline-2	Ours	Baseline-2	Ours	Baseline-2	Ours	Baseline-2
Lastfm	0.22	0.21	0.25	0.22	0.17	0.17	0.21	0.14	0.13	0.13	0.12	0.10
Google+	0.12	0.11	0.08	0.07	0.19	0.18	0.12	0.09	0.12	0.11	0.13	0.12
Citeseer	0.40	0.30	0.24	0.23	0.41	0.38	0.18	0.11	0.13	0.10	0.08	0.05

Table 12: TPR@1%FPR of 3-SMIA & 4-SMIA under Setting 1 (Non-transfer setting). We did not consider Baseline-1 and Baseline-3 for comparison because their TPR@1%FPR is not measurable as they do not use a threshold. The best attack AUC for each dataset and each GNN model is highlighted with olive color.

Attack AUC								TPR@1%FPR							
3-SMIA				4-SMIA				3-SMIA				4-SMIA			
Class	GCN	SAGE	GAT	Class	GCN	SAGE	GAT	Class	GCN	SAGE	GAT	Class	GCN	SAGE	GAT
3-clique	0.66	0.63	0.71	4-clique	0.82	0.83	0.79	3-clique	0.12	0.08	0.11	4-clique	0.2	0.18	0.23
2-hop path	0.65	0.62	0.65	3-hop path	0.79	0.68	0.72	2-hop path	0.05	0.05	0.06	3-hop path	0.07	0.11	0.1

Table 13: Attack AUC and TPR@1%FPR of member classes of 3-SMIA and 4-SMIA under Setting 1 (Non-transfer setting).

3-SMIA and 4-SMIA outperform Baseline-2 in all the settings. This demonstrates the effectiveness of our attacks.

Attack AUC and TPR@1%FPR performance of member classes. To better understand how the attack performs over the member classes (i.e., cliques and paths), next, we measure the attack performance for each member class (i.e., 3-clique and 2-hop paths for 3-SMIA, and 4-cliques and 3-hop paths for 4-SMIA). For the evaluation, we only consider one subgraph structure (e.g., the clique) as the member, and the remaining structures (e.g., the paths and other structures that are neither paths and cliques) as the non-member. Table 13 presents the AUC and TPR@1%FPR for each member class. For both attack accuracy metrics, 3-SMIA and 4-SMIA are shown to be more effective to infer cliques than the paths.

F.2 Transfer Setting

Attack Performance of 4-SMIA under Transfer Setting Figure 9 presents the attack AUC of 4-SMIA under both dataset transfer and model transfer settings. First, we observe that 4-SMIA is still effective under the data transfer setting. The attack AUC spans a range of 0.59 to 0.72 (Figure 9 (a) - (c)), which is much higher than the random guess (0.5). 4-SMIA also remains effective under model transfer setting, with the attack AUC ranges between 0.58 and 0.76 (Figure 9 (d) - (f)).

Why do transfer attacks work? To understand why the transfer attacks remains effective under both data transfer and model transfer settings, we plot the distribution of three classes (clique, path, and non-members) by the attack features derived from the shadow model and the target model respectively for both dataset transfer and model transfer settings in Figure 10. We observe that, despite the varying distributions of attack features in both the shadow and target models, the three classes remain discernible from each other. We attribute this phenomenon to the inherent similarity among connected nodes within subgraphs compared to disconnected nodes outside them, owing to the intrinsic message-passing mechanisms of GNN models. These results demonstrate that our attack has acquired and effectively transferred this knowledge across diverse datasets and models.

F.3 Impact of Node Importance on Attack Performance

Figure 11 illustrates the impact of node degree, closeness centrality, betweenness centrality, and node feature similarity on the attack accuracy of 3-SMIA and 4-SMIA targeting the GCN model using the Google+ dataset. We observe comparable trends to those in Figure 6. Specifically, the attack accuracy of clique inference steadily increases for the target nodes of higher importance across all the four types of importance metrics. This trend is likely attributable to the target model's tendency to memorize more significant nodes, influenced by the message passing and neighborhood aggregation mechanisms inherent in GNNs. However, a consistent pattern is

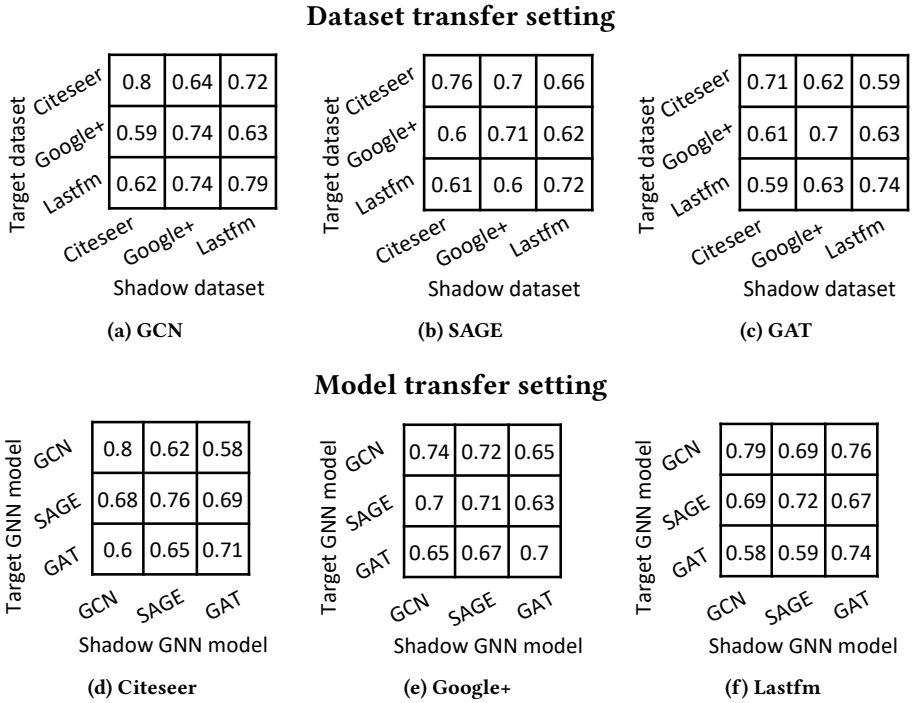
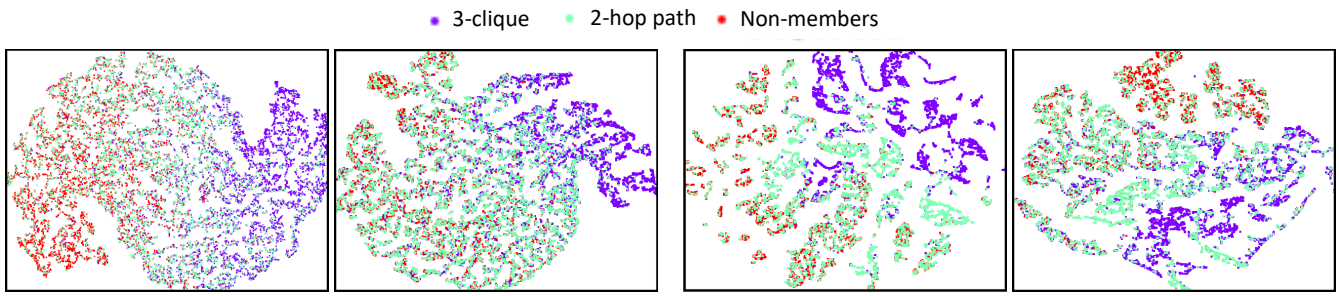


Figure 9: Attack AUC of 4-SMIA under dataset and model transfer setting.



(a) Distribution of three classes by the attack features derived from shadow model (GCN trained on Lastfm dataset)

(b) Distribution of three classes by attack features derived from target model (GCN trained on Google+ dataset)

(c) Distribution of three classes by attack features derived from shadow model (GCN trained on Citeseer dataset)

(d) Distribution of three classes by attack features derived from target model (SAGE trained on Citeseer dataset)

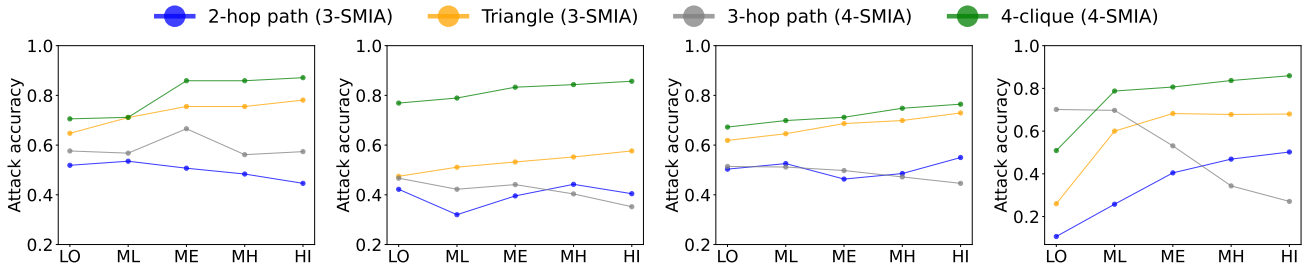
Data transfer setting **Model transfer setting**

Figure 10: Distribution of attack features under both dataset transfer and model transfer settings (3-SMIA).

not observed for k-hop paths. We guess it could be attributed the introduction of additional uncertainty in attack inference due to the presence of disconnected node pairs within V_{att}. In particular, we observe a drop of the attack accuracy on the inference of 3-hop paths. Such drop was minor for node sets that contain nodes of higher degrees, closeness centrality, and betweenness centrality (Figure 11 (a) - (c)). But the drop became substantial for node sets of similar node features (Figure 11 (d)). We guess the reason behind this phenomenon is that node features become less influential on attack accuracy compared to node structure when inferring higher-order connectivity.

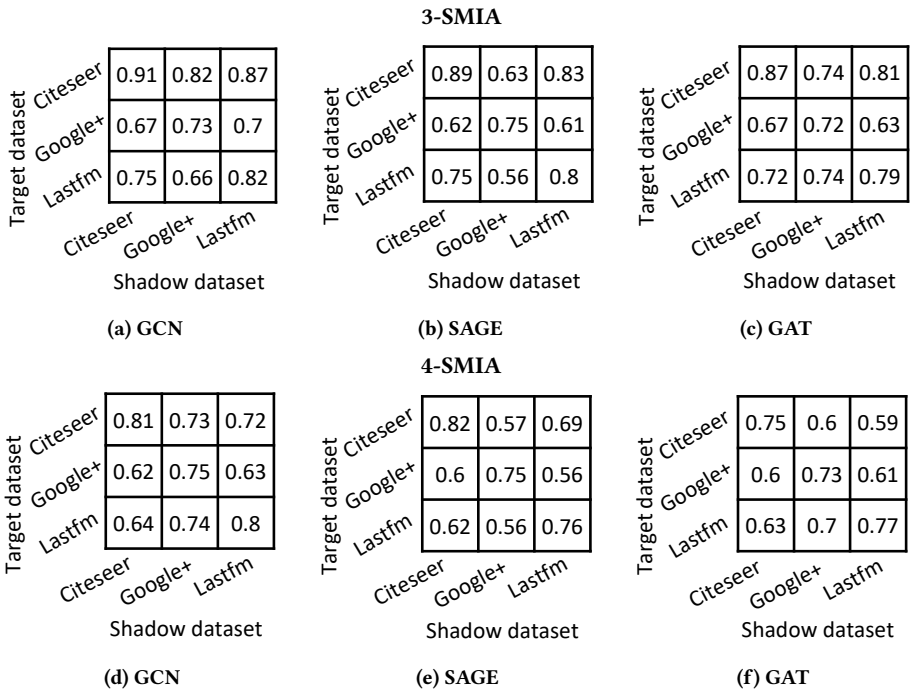
F.4 White-box Attacks under Transfer Setting

Data transfer setting. Figure 12 presents the AUC performance of white-box attacks under the data transfer setting. We observe that the attack AUC ranges from 0.61 to 0.87 for 3-SMIA and 0.6 to 0.74 for 4-SMIA across the three datasets and three GNN models. This indicates the effectiveness of our attack under the white-box data transfer setting. Additionally, similar to the non-transfer setting, the performance of the white-box attack under the data transfer setting generally outperforms the black-box attack in most of the settings. However, there are a few exceptions. For example, the attack AUC of the white-box attacks is lower than that of the black-box attacks



(a) Node degree (b) Node closeness centrality (c) Node betweenness centrality (d) Node feature similarity

Figure 11: Impact of node degree, node closeness, betweenness centrality, and node feature similarity on attack accuracy (BA) of 3-SMIA and 4-SMIA (GCN, Google+ dataset). "LO", "ML", "ME", "MH", "HI" represent the node group of low, medium low, medium, medium high, and high average node degree/closeness/betweenness centrality respectively.



(a) GCN (b) SAGE (c) GAT

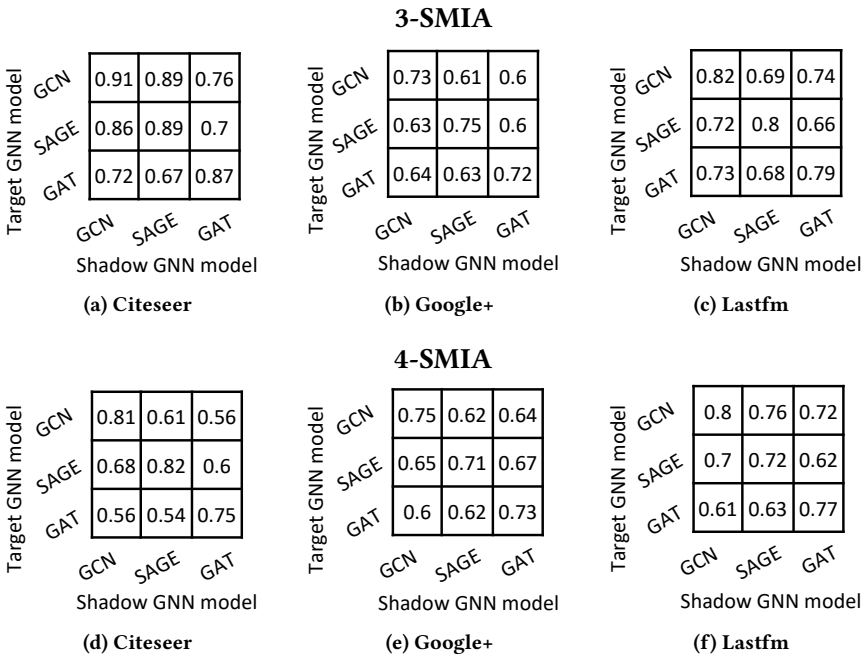
(d) GCN (e) SAGE (f) GAT

Figure 12: White-box attack AUC of 3-SMIA and 4-SMIA under dataset transfer setting.

for both 3-SMIA and 4-SMIA when Google+ is used as the shadow model and the Lastfm dataset is used as the target dataset. This phenomenon implies that, under rare cases, the similarity between node embeddings may not be as effective as the similarity between posterior probabilities for knowledge transfer.

Model transfer setting. Figure 13 presents the AUC performance of white-box attacks under the model transfer setting. We observe that the attack AUC ranges from 0.6 to 0.89 for 3-SMIA and 0.54 to 0.76 for 4-SMIA across the three datasets. This indicates the effectiveness of our white-box attack under the model transfer setting.

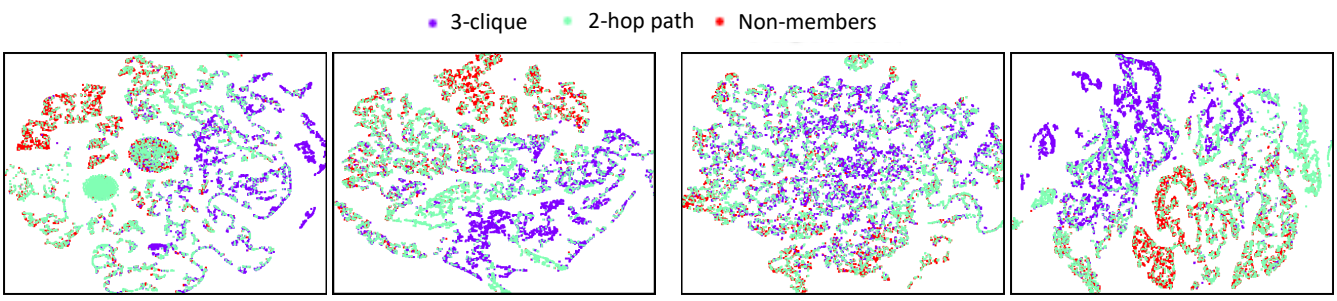
We have also noted that, although uncommon, the AUC of white-box attacks can be lower than that of black-box attacks in specific configurations. An example of this is seen when using GAT as the shadow model and SAGE as the target model on the Citeseer dataset for the 3-SMIA attack. To delve into this phenomenon, we present in Figure 14 the distribution of the three classes (k-cliques, (k-1)-hops, and non-members) by their attack features for both white-box and black-box attacks in the aforementioned setting. Our analysis reveals that under this particular setup, the black-box attack features derived from posterior probabilities exhibit a better ability to distinguish between the three classes compared to the white-box attack features derived from node embeddings. Consequently, this leads to a higher attack performance by the black-box approach compared to the white-box method in this specific scenario.



(a) Citeseer (b) Google+ (c) Lastfm

(d) Citeseer (e) Google+ (f) Lastfm

Figure 13: White-box attack AUC of 3-SMIA and 4-SMIA under model transfer setting.



(a) GAT (b) SAGE (c) GAT (d) SAGE

Black-box attack White-box attack

Figure 14: Distribution of attack features for both white-box and black-box attacks (3-SMIA, Citeseer dataset).

G ADDITIONAL RESULTS OF DEFENSE PERFORMANCE

G.1 Defense Performance under Gaussian noise

We evaluate the defense performance of adding Gaussian noise to node embeddings. The Gaussian noise s follows the density distribution of

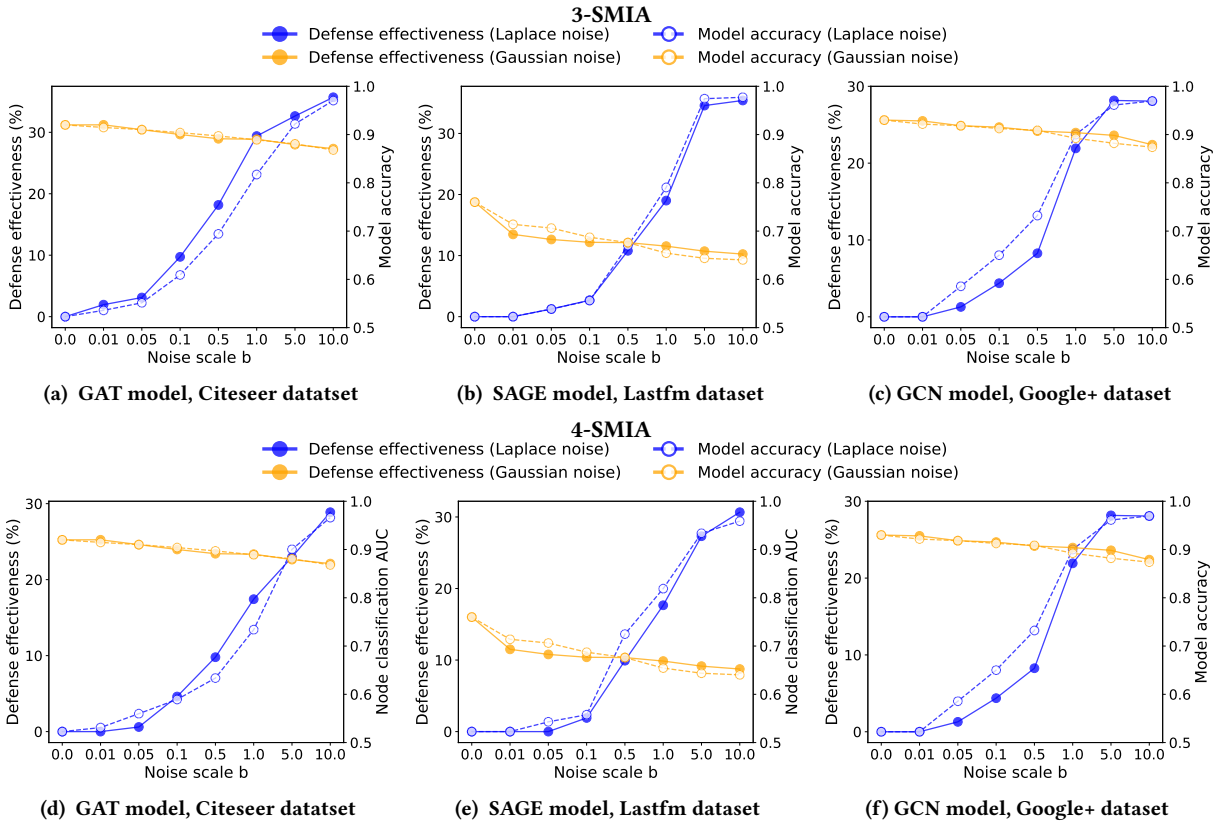
$$s = \frac{1}{\sigma\sqrt{2\pi}} \exp^{\frac{1}{2}(\frac{x-\mu}{\sigma})^2}, \tag{5}$$

with the expectation of μ and the standard deviation of σ. In our experiments, we set $\mu = 0$, and $\sigma = \{0, 0.01, 0.05, 0.1, 0.5, 1, 5, 10\}$, which is the same as the noise scale of Laplace noise.

Figure 15 presents both defense performance and model accuracy when Gaussian noise or Laplace noise is employed for defense. We have observed that employing Gaussian noise for defense yields similar performance to using Laplace noise in terms of both defense effectiveness and model accuracy. Consequently, we primarily focus on presenting the performance results of the defense mechanism utilizing Laplace noise in the main body of the paper.

G.2 Defense Performance Results for More Settings

Figure 16 (a) evaluates the impact of the perturbation ratio r on SAGE model and Citeseer dataset. The trends observed align closely with those in Figure 7 (a). First, our perturbation techniques effectively curtail the attack AUC. For instance, even with a minimal perturbation ratio of 0.2 (perturbing 20% of embedding dimensions), the attack AUC of 3-SMIA can be reduced from 0.83 to 0.70 (15.7% reduction). Second, an increase in the perturbation ratio (r) notably enhances defense efficacy. As r escalates, the defense power experiences substantial improvement. Notably, when the perturbation ratio reaches 0.8 for both attacks, the attack AUC can be diminished to a level close to the random guess threshold of 0.5.



(a) GAT model, Citeseer datatset **(b) SAGE model, Lastfm dataset** **(c) GCN model, Google+ dataset**

(d) GAT model, Citeseer datatset **(e) SAGE model, Lastfm dataset** **(f) GCN model, Google+ dataset**

Figure 15: Gaussian noise vs. Laplace noise in terms of defense performance and model accuracy.

Furthermore, we observe a discrepancy in the defense performance of our method across GAT, GCN and SAGE. Specifically, our method demonstrates a stronger defense capability for GAT (Figure 7) and GCN compared to SAGE. We attribute this discrepancy to the differing aggregation functions employed by these models. Specifically, both GAT and GCN utilize the entire neighborhood for aggregation during training, whereas SAGE adopts a sampling-based approach resulting in partial neighborhood aggregation. Consequently, our method introduces less perturbations for GAT and GCN than for SAGE, thereby rendering it less effective in the former case.

Figure 16 (b) and Figure 16 (c) evaluate the impact of the noise scale b on 3-SMIA and 4-SMIA, alongside comparison with two baseline methods. Firstly, our method's defense effectiveness rises proportionally with increasing noise scale values. Our approach's performance becomes comparable to NP's efficacy when the noise scale reaches 10 for both 3-SMIA and 4-SMIA. This correlation aligns with our expectations, given NP's direct introduction of noise into posteriors, contrasted with our method's selective noise addition to specific embedding dimensions. Additionally, our approach achieves similar performance to DP when the noise scale expands to 10 for both attacks.

Figure 16 (d) evaluates the impact of the perturbation ratio r on GCN model and Citeseer dataset. The trends observed align closely with those in Figure 7 (a). First, our perturbation techniques effectively restrain the attack AUC. A compelling instance is evident when considering a modest perturbation ratio of 0.2, corresponding to the perturbation of just 20% of embedding dimensions, the attack AUC of 3-SMIA undergoes a reduction of 12.4% (from 0.89 to 0.78). Second, an increase in the perturbation ratio (r) notably enhances defense efficacy. As r escalates, the defense power experiences substantial improvement. Notably, when the perturbation ratio reaches 0.8 for both attacks, the attack AUC can be diminished to a level close to the random guess threshold of 0.5.

Figure 16 (e) & (f) evaluate the impact of the noise scale b on 3-SMIA and 4-SMIA and two baselines. Our observations are similar to those from Figure 7 (b) & (c); thus we omit the detailed discussion for simplicity.

Model accuracy. Figure 16 (a) & (d) (with the y-axis on the right) illustrate the model accuracy (node classification AUC) across various perturbation ratios. We observe that there is a negligible amount of model accuracy loss (up to 0.05%) even as the perturbation ratio increases to 0.8. Additionally, Figure 16 (b) & (c) & (e) & (f)(the y-axis at right) compare the model accuracy of our defense solution and the two baselines when the noise scale b increases. Our method consistently outperforms both baselines in terms of model accuracy when b exceeds 0.05.

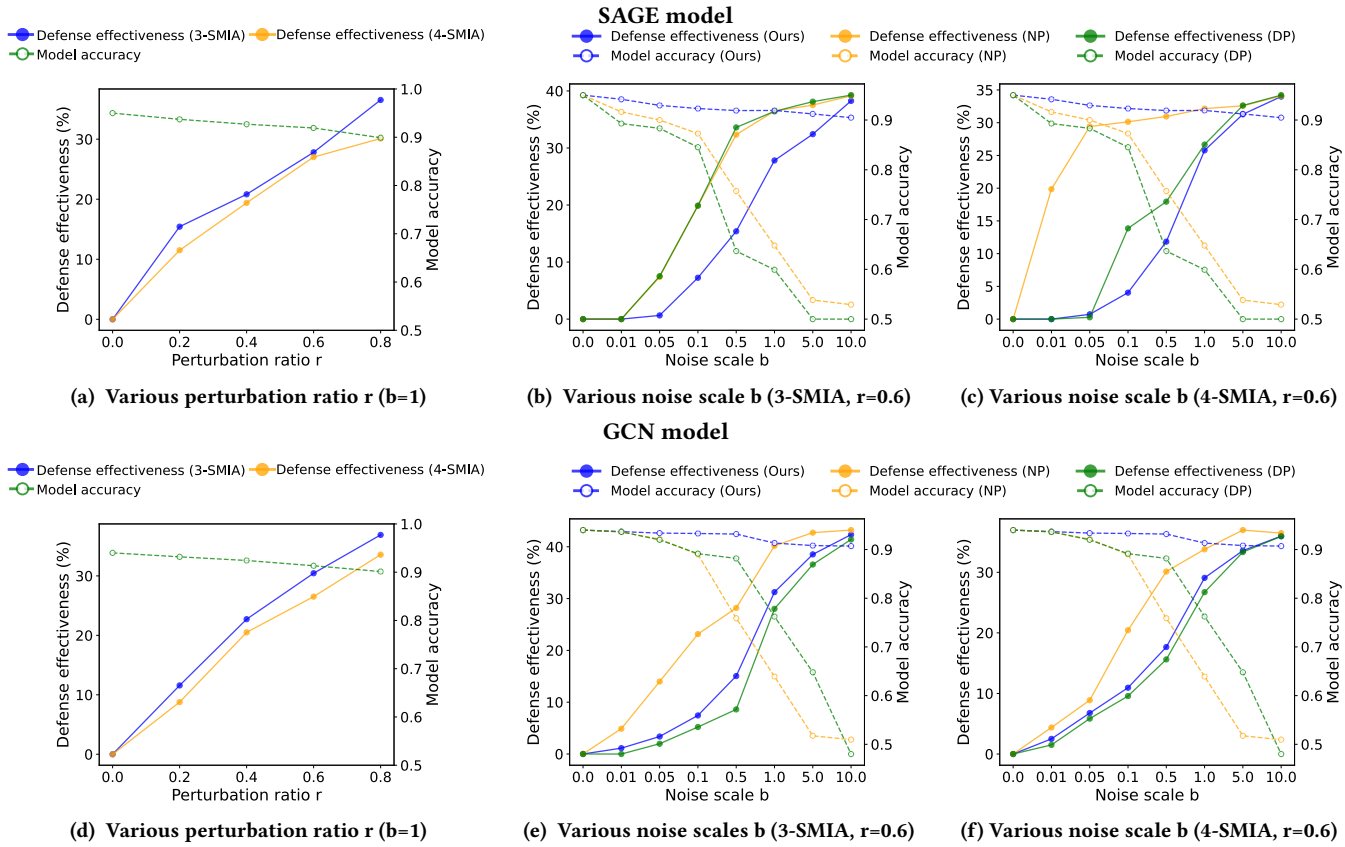


Figure 16: Defense performance when GCN and SAGE models are the target models (Citeseer dataset). The solid and dotted lines in (b) and (c) denote the defense effectiveness and node classification AUC respectively.

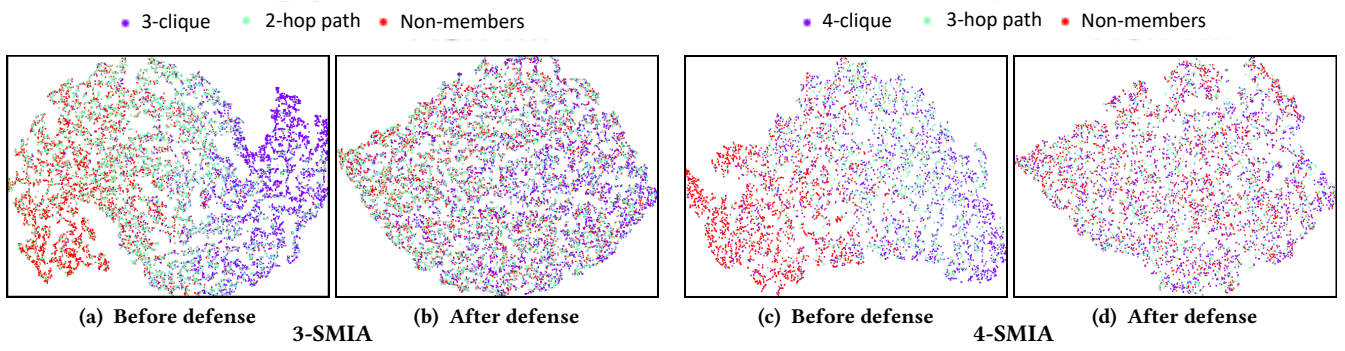


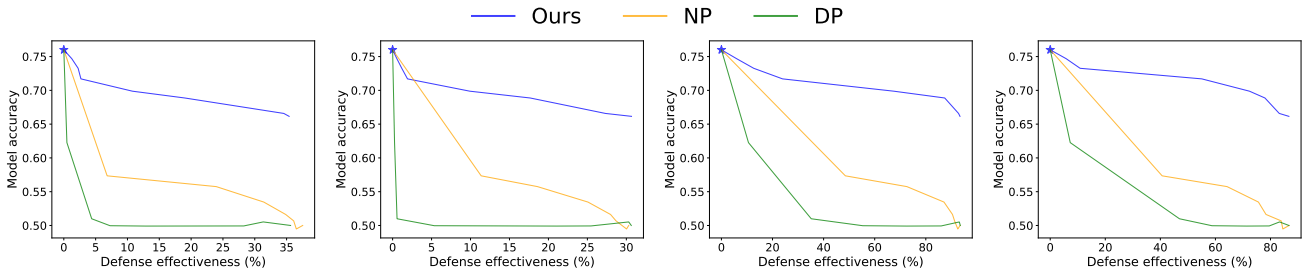
Figure 17: Visualization of the distribution of attack features of different member classes before and after applying our defense (GCN model, Lastfm dataset).

G.3 Visualization of Attack Features' Distribution before and after Defense

Figure 17 illustrates the distribution of the attack features of all the classes before and after applying our defense method. We notice a significant increase in the indistinguishability of the three classes after introducing noise to the embedding. This observation highlights that, despite being added solely to the least important dimensions of the embedding, the noise has led to alterations in the posterior probabilities and their corresponding similarities. Consequently, it has effectively modified the distribution of attack features across all classes, thereby reducing the accuracy of the attack.

G.4 Defense-utility Trade-off

First, to illustrate the trade-off between defense and utility, we plot the defense-utility ROC curve of our defense and the two



(a) 3-SMIA, Attack AUC (b) 4-SMIA, Attack AUC (c) 3-SMIA, TPR@1%FPR (d) 4-SMIA, -TPR@1%FPR

Figure 18: The defense-utility ROC curve (SAGE model, Lastfm dataset). The ⋆ symbol indicates no defense.

Dataset	3-SMIA									4-SMIA								
	GCN			SAGE			GAT			GCN			SAGE			GAT		
	Ours	NP	DP	Ours	NP	DP	Ours	NP	DP	Ours	NP	DP	Ours	NP	DP	Ours	NP	DP
Lastfm	0.59	0.59	0.5	0.6	0.52	0.5	0.45	0.45	0.42	0.63	0.57	0.5	0.61	0.6	0.51	0.51	0.53	0.32
Google+	0.48	0.42	0.42	0.7	0.56	0.41	0.65	0.67	0.58	0.51	0.5	0.45	0.65	0.64	0.58	0.73	0.61	0.63
Citeseer	0.87	0.73	0.67	0.83	0.72	0.67	0.85	0.81	0.75	0.92	0.85	0.81	0.86	0.85	0.74	0.91	0.8	0.72

Table 14: Defense-utility trade-off score of the three defense methods. The reduction in TPR@1%FPR is used to measure the defense effectiveness. The best trade-off for each GNN model and each dataset is marked with olive color.

baseline methods in Figure 18. We establish a set of experimental configurations that vary the noise scale values b across {0.01, 0.05, 0.1, 0.5, 1, 5, 10} with a constant perturbation ratio (r = 0.4). Equivalent noise scales are applied to the other two baseline methods. Following this, we evaluate the defense effectiveness and target model AUC for each configuration. These results are consolidated into a defense-utility ROC curve, capturing the interplay between defense effectiveness and target model accuracy across different settings. We consider two different metrics of defense effectiveness, namely the percentage of attack AUC and TPR@1%FPR that is reduced after the defense, respectively. A higher reduction indicates a stronger defense. We observe that our defense method outperforms the two baselines in the trade-off between defense performance and target model performance. For example, in Figure 18 (a), when the defense effectiveness is 30% (the attack accuracy is reduced to 0.56), the model performance of our attack is significantly higher than NP and DP.

Next, to quantify the defense-utility trade-off, we measure the trade-off score as the Area Under the Curve (AUC) of the defense-utility ROC curve. Table 14 shows the defense-utility trade-off score when the defense effectiveness is measured as the reduction in TPR@1%FPR. Our defense mechanism surpasses the two baseline methods in terms of the defense-utility trade-off in most of the settings.