ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240646

Efficient Model-Stealing Attacks Against Inductive Graph Neural Networks

Marcin Podhajski[a,b,*], **Jan Dubiński**[a,c], **Franziska Boenisch**[d], **Adam Dziedzic**[d], **Agnieszka Pregowska**[b] **and Tomasz P. Michalak**[a,e]

[a]IDEAS NCBR, 00-801 Warsaw, Poland
[b]Institute of Fundamental Technological Research, Polish Academy of Sciences, Pawińskiego 5B, 02-106 Warsaw, Poland
[c]Warsaw University of Technology, Faculty of Electronics and Information Technology, 00-661 Warsaw, Poland
[d]CISPA, 66123 Saarbrücken, Germany
[e]University of Warsaw, Faculty of Mathematics, Informatics and Mechanics, 00-927 Warsaw, Poland

Abstract. Graph Neural Networks (GNNs) are recognized as potent tools for processing real-world data organized in graph structures. Especially inductive GNNs, which allow for the processing of graph-structured data without relying on predefined graph structures, are becoming increasingly important in a wide range of applications. As such these networks become attractive targets for model-stealing attacks where an adversary seeks to replicate the functionality of the targeted network. Significant efforts have been devoted to developing model-stealing attacks that extract models trained on images and texts. However, little attention has been given to stealing GNNs trained on graph data. This paper identifies a new method of performing *unsupervised* model-stealing attacks against inductive GNNs, utilizing graph contrastive learning and spectral graph augmentations to efficiently extract information from the targeted model. The new type of attack is thoroughly evaluated on six datasets and the results show that our approach outperforms the current state-of-the-art by Shen et al. (2021). In particular, our attack surpasses the baseline across all benchmarks, attaining superior fidelity and downstream accuracy of the stolen model while necessitating fewer queries directed toward the target model.[1]

1 Introduction

Graph Neural Networks (GNNs) are specifically designed to operate on graph-structured data, such as molecules, social networks or complex infrastructure [8, 42, 29]. Unlike models developed for images or text, GNNs simultaneously consider both node features and graph structures. GNNs have garnered significant attention for their effectiveness in tasks involving structured data. A major advantage of GNNs is their ability to scale to large graphs while preserving computational efficiency. However, it is also well-documented that Machine Learning (ML) models, including GNNs, are vulnerable to *model stealing attacks* [35, 14, 6], where an adversary with query access to a *target (victim) model* can extract its parameters or functionality, often at a fraction of the cost required to train the model from scratch.

In such scenarios, the adversary sends a series of queries as inputs to the target model's API and obtains the corresponding outputs. A local surrogate model is then trained to mimic the outputs of the target model for the same input data. Consequently, the surrogate model not only compromises the intellectual property of the target model but also potentially enables further attacks, such as extracting private training data [22] or constructing adversarial examples [26].

It is worth stressing that the majority of the current efforts on model stealing attacks concentrate on ML models for images and text data [35, 24, 38, 18]. Despite the growing significance of GNNs, model-stealing attacks against these types of neural networks

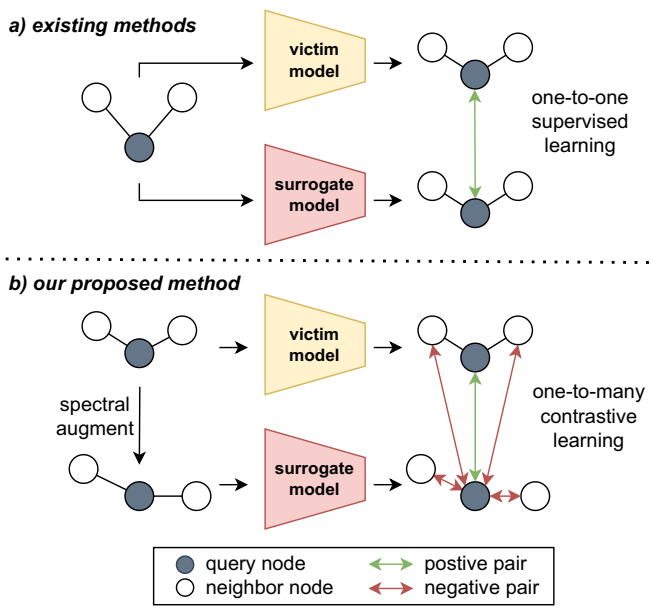

Figure 1. Existing model stealing methods rely on aligning only the predictions for individual graph nodes sent to the victim and surrogate model. The new method identified in this paper greatly increases the amount of information extracted from the victim model outputs. We align the node representation from the surrogate model with the corresponding representation from the victim model while simultaneously distinguishing it from the representation of other nodes. By integrating spectral augmentations to augment the graph inputs of the surrogate models during its training process, we further increase the efficiency of the stealing process per query sent to the victim model.

* Corresponding Author. Email: marcin.podhajski@ideas-ncbr.pl
[1] Code available at https://github.com/m-podhajski/EfficientGNNStealing

are severely underexplored compared to image or text models. To our knowledge, only one study [34] addresses attacks on inductive GNNs, i.e. a type of model that can generalize well to unseen nodes. The authors define the threat model for model stealing attacks on inductive GNNs and describe six attack methods based on the adversary's background knowledge and the responses of the target models. During the attack, the response of the surrogate model is aligned with the response of the target model by minimizing the RMSE loss between them. Although their approach yields promising results, the underlying method of training the surrogate model does not efficiently utilize the information available from the target's responses. This increases the number of queries required to extract the target model's API which results in higher chances of attack detection and mitigation.

Inspired by recent approaches to encoder model stealing [6, 21, 32] we leverage the significant fact that augmented graph inputs should produce similar model outputs. Thus, the GNN model-stealing attack identified in this paper utilizes a contrastive objective at the node level. Specifically, for an input graph, we generate two graph views—one from the target model and one from the surrogate model for an augmented graph input. The surrogate model is trained by maximizing the agreement of node representations in these two views. This contrastive objective considers both positive and negative pairs between the node representations from the target and surrogate model. To augment the graph input to the surrogate model we leverage graph transformation operations derived via graph spectral analysis, enabling contrastive learning to capture useful structural representations. Those transformations include spectral graph cropping and graph frequency components reordering. Thus, the proposed methods that combine contrastive learning loss and effective graph-specific augmentations enable efficient usage of the information from the target model outputs. As a result, our approach fits into the framework established by Shen et al. (2021) [34]. It presents an enhanced surrogate model training methodology that significantly augments the efficiency and performance of the model-stealing process, outperforming the state-of-the-art method with less than half of its query budget.

We evaluate all proposed attacks on three popular inductive GNN models including GraphSAGE [11], Graph Attention Network (GAT) [37], and Graph Isomorphism Network (GIN) [41] with six benchmark datasets, i.e., DBLP [25], Pubmed [31], Citeseer Full [9], Coauthor Physics [33], ACM [39], and Amazon Co-purchase Network for Photos [23]. The experiments demonstrate that our model-stealing attack consistently outperforms the state-of-the-art baseline approach across the vast majority of datasets and models. In the particular case of using SAGE as the surrogate model the increase in accuracy reached 36.8 %pt. for a GIN target trained on Amazon Co-purchase Network for Photos.

Overall, the main contributions of this work are as follows:

- We identify a new unsupervised GNN model stealing attack trained with a contrastive objective at the node level that compels the embedding of corresponding nodes from the surrogate and victim models to align while also distinguishing them from embeddings of other nodes within the surrogate and the target model.
- We adapt spectral graph augmentations to generate multiple graph views for the surrogate model in the contrastive learning process.
- We perform an extensive comparison of the newly identified stealing technique to the state of the art. The new stealing process demonstrates better performance while requiring fewer queries sent to the target model.

2 Preliminaries

Table 1. Notations used in the presented paper. Lowercase letters denote scalars, bold lowercase letters denote vectors and bold uppercase letters denote matrices.

Notation	Description
$\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{X}, \mathbf{C})$	graph
$v, u \in V$	node
$n = \lvert\mathbf{V}\rvert$	number of nodes
$d \in \mathbb{N}$	dimension of a node feature vector
$b \in \mathbb{N}$	dimension of a node embedding vector
$\mathbf{A} \in \{0,1\}^{n\times n}$	adjacency matrix
$\mathbf{A}' \in \langle -1,1\rangle^{n\times n}$	augmented adjacency matrix
$\mathbf{X} \in \mathbb{R}^{n\times d}$	feature matrix
$\mathbf{H} \in \mathbb{R}^{n\times b}$	embedding matrix
$\mathbf{h}_v^i \in \mathbb{R}^b$	node embedding at layer i
$\mathcal{N}(v)$	neighborhood of v
$\mathcal{M}_T/\mathcal{M}_S$	target/surrogate GNN model

Graph Neural Networks are a class of neural networks that use the graph structure $\mathbf{A}$ as well as node features $\mathbf{X}$ as the input to the model. GNNs learn an embedding vector of a node h_v or the entire graph H. These embeddings can be then used in a range of important tasks, including node classification [11, 16], link prediction [27, 10, 36] or graphs and subgraph classification [1, 19].

Most of the modern GNNs follow a neighborhood aggregation strategy. We start with $\mathbf{h}_v^0 = \mathbf{X}_v$ and after k iterations of aggregation, the k-th layer of a GNN is

$$\mathbf{h}_v^k = AGGREGATE(\mathbf{h}_v^{k-1}, MSG(\mathbf{h}_v^{k-1}, \mathbf{h}_u^{k-1}),$$

$u \in \mathcal{N}(v)$. In effect, the node embedding at the k-th layer captures the information within its k-hop neighborhood.

The training of GNNs occurs in two distinct settings: the transductive setting and the inductive setting. In the transductive setting the data consists of a fixed graph and a subset of nodes is assigned labels and utilized during the training process, while another subset remains unlabeled. The primary objective is to leverage the labeled nodes and predict the labels for the previously unlabeled nodes within the same graph. However, the transductive models do not generalize to newly introduced nodes.

In the inductive setting, the GNN is designed to generalize its learning to accommodate new, unseen nodes or graphs beyond those encountered during training. The first model capable of inductive learning was GraphSAGE introduced by [11]. Other inductive models include Graph Attention Network (GAT) proposed by [37], and Graph Isomorphism Network (GIN) by [41].

3 Threat Model

This section outlines the threat model, detailing the attack setting, adversary's goals, and capabilities. The proposed approach is based on the threat model widely presented in [34].

Attack Setting. We investigate a challenging *black-box* scenario where the adversary is entirely unaware of the target GNN model's specifics, including its parameters and architecture, and cannot interfere with the training process or the training graph $\mathbf{G}_O$. Our study focuses on the responses to three distinct types of node-level queries. In this context, the target model is an inductive GNN that utilizes the l-hop subgraph $\mathbf{G}_v^l$ of a node v as input to generate outputs for that node. These outputs could be in the form of predicted posterior probabilities, node embedding vectors, or a 2-dimensional t-SNE projection.

Adversary's Goal. According to the classification established by [14], adversaries typically have two primary objectives: theft or reconnaissance. A *theft adversary* aims to create a surrogate model $\mathcal{M}_S$ that replicates the performance of the target model $\mathcal{M}_T$ for the specific task at hand [35, 26]. On the other hand, a *reconnaissance adversary* seeks to develop a surrogate model $\mathcal{M}_S$ that approximates $\mathcal{M}_T$ across all possible inputs. Achieving such a high level of fidelity allows the adversary to use the surrogate model for subsequent attacks, such as generating adversarial examples, without needing to interact directly with the target model $\mathcal{M}_T$ [26].

Adversary's Capabilities. Firstly, we consider a scenario where the adversary interacts with a target model $\mathcal{M}_T$ via a public API [35, 24, 12, 13]. The adversary submits a query graph $\mathbf{G}_Q$ and receives responses $\mathbf{R}$, which could be a node embedding matrix ($\mathbf{H}$), a predicted probability matrix ($\mathbf{\Theta}$), or a t-SNE projection matrix ($\mathbf{\Upsilon}$). These responses represent typical outputs from such APIs, varying in the level of information they provide for tasks like graph visualization, transfer learning, and fine-tuning GNNs.

Additionally, We assume the query graph $\mathbf{G}_Q$, with node features $\mathbf{X}_Q$ and graph structure $\mathbf{A}_Q$, is drawn from the same distribution as the training graph $\mathbf{G}_O$ used to train $\mathcal{M}_T$. This implies that $\mathbf{G}_Q$ and $\mathbf{G}_O$ are sampled from the same dataset, consistent with studies on neural network attacks using public datasets [14, 12], particularly in graph-intensive domains like social networks and molecular structures. As discussed in Section 4.2, this assumption can be adjusted.

4 Attack Framework

This section aims to discuss attack scenarios that can be launched by the adversary depending on different levels of knowledge and describe the general framework of GNN model-stealing. We adopt both, the attack structure and taxonomy proposed by Shen and co-authors [34].

4.1 Attack Taxonomy

To conduct a model stealing attack, the adversary relies on two essential components: the query graph $\mathbf{G}_Q = (\mathbf{A}_Q, \mathbf{X}_Q, \mathbf{C}_Q)$ and the corresponding response $\mathbf{R}$. The query graph consists of

- $\mathbf{A}_Q$: the adjacency matrix, which represents the graph structure,
- $\mathbf{X}_Q$: the node feature matrix, detailing the attributes of each node,
- $\mathbf{C}_Q$: the labels or class information associated with nodes or edges.

The response $\mathbf{R}$ from the target model can be in one of several forms, such as

- $\mathbf{H}$: node embeddings, which are vectors representing each node in the graph,
- $\mathbf{\Theta}$: predicted posterior probabilities, indicating the model's confidence in various outcomes,
- $\mathbf{\Upsilon}$: a t-SNE projection matrix, which visualizes node embeddings in a 2-dimensional space.

The adversary may not have complete structural information about the query graph $\mathbf{G}_Q$, lacking the adjacency matrix $\mathbf{A}_Q$, which leads to six possible attack scenarios grouped into two types. In a Type I attack, the adversary queries the target model using graphs from the same distribution as those used in training. This is common in domains like drug interaction predictions, fraud detection, and recommendation systems, where abundant graph data is available.

Conversely, a Type II attack involves compromising the model without access to the specific graph structure used during training. Here, the adversary must reconstruct or infer missing information, such as in social networks where user connections are private, and only public profile information is accessible [43].

4.2 Learning the Missing Graph Structure

In Type I attacks, the adversary trains a surrogate model $\mathcal{M}_S$ using responses from the target model to the query graph $\mathbf{G}_Q$. In contrast, Type II attacks require inferring the missing graph structure $\mathbf{A}_Q$. Following [34], we use the IDGL framework by Chen et al. [3] to infer $\mathbf{G}_Q$. This involves minimizing a joint loss function that combines task-specific prediction loss with graph regularization. The adversary starts with a k-nearest neighbors (kNN) graph based on multi-head weighted cosine similarity and iteratively improves it using IDGL, optimizing the joint loss function without interacting directly with the target model.

4.3 Training the Surrogate Model

When the adversary has access to the graph structure $\mathbf{A}_Q$, their goal is to train a surrogate model $\mathcal{M}_S$. This model is trained using the l-hop subgraphs of all nodes from the query graph $\mathbf{G}_Q$ as input and the corresponding responses $\mathbf{R}$ from the target model as supervised signals. As noted by [34], nodes that are close or connected in $\mathbf{G}_Q$ should also be close in the t-SNE projection matrix $\mathbf{\Upsilon}$ or the node embedding matrix $\mathbf{H}$. Thus, the target model's responses can uniformly be treated as embedding vectors. The primary goal is to minimize the loss ($\mathcal{L}_R$) between the surrogate model's output embeddings $\hat{\mathbf{H}}_Q$ and the target model's responses $\mathbf{R}$. It's important to note that the surrogate model's output cannot be directly used for node classification, as it is trained to mimic the target model's responses, not to perform classification. An additional step is required to enable node classification. After training and freezing the surrogate model, an MLP classification head ($\mathcal{O}$) is optimized, using the surrogate model's output ($\hat{\mathbf{H}}_Q$) as input and $\mathbf{C}_Q$ as supervision to minimize prediction error. This two-step process ensures the surrogate model's outputs are effectively utilized for node classification.

5 Proposed Model Stealing Attack

This section comprises a description of the training of the surrogate model in model-stealing attacks on GNNs. The proposed method is in line with the framework shown in Section 4.

5.1 Contrastive GNN stealing

As the number of queries sent by the attacker to the victim's model increases, increase the risk of detection and the cost of the attack. Thus, the efficiency of model stealing is measured by the number of queries sent to the victim model, whose outputs serve as targets for training the surrogate model. However, the adversary faces no constraints on the number of samples sent to the surrogate model. For that reason, we argue that simply matching the outputs of the victim and surrogate model proves inefficient within the context of model stealing. Building upon this observation, we propose a novel approach grounded in contrastive graph learning [48, 46]. We utilize a contrastive objective function that forces the embedding of corresponding nodes in the surrogate and victim models to converge

while simultaneously distinguishing them from the embedding of other nodes within both models.

Starting with the given graph $\mathbf{G} = (\mathbf{X}, \mathbf{A})$, we initiate an augmentation process, resulting in the graph $\mathbf{G}' = (\mathbf{X}', \mathbf{A}')$. The augmentation is applied to both graph structure and feature matrix resulting in more robust representations that are invariant to various transformations and perturbations. Subsequently, the surrogate model $\mathcal{M}_S$ generates an embedding $\mathbf{H}_S$ based on the augmented graph $\mathbf{G}'$. The dimensional of the embedding $\mathbf{H}_S$ matches that of the response $\mathbf{R}$ obtained from the target model $\mathcal{M}_T$. Utilizing a non-linear projection head $g()$ [2, 48], we map both the target response $\mathbf{R}$ and the surrogate embedding $\mathbf{H}_S$ into a shared representation space, where a contrastive loss is applied:

$$\mathcal{J} = -\frac{1}{2n}\sum_{i=1}^{n}[\ell_i(t,s) + \ell_i(s,t)], \tag{1}$$

$$\ell_i(x,y) = \log \frac{e^{c(x_i,y_i)/\tau}}{\sum_{k=1}^{n} e^{c(x_i,y_k)/\tau} + \sum_{k=1,k\neq i}^{n} e^{c(y_i,y_k)/\tau}}. \tag{2}$$

where $t = g(\mathbf{R})$ and $s = g(\mathbf{H}_S)$ represent the matrices in the representation space from the target and surrogate models, respectively, and x_i denotes embedding vector corresponding to node v_i in that space. The function c stands for cosine similarity, and τ is a temperature parameter.

The embeddings from different models x_i and y_i are a positive pair, drawing closer together through the minimization of $e^{c(x_i,y_i)/\tau}$.

To maintain a distinction between a negative pair of embeddings x_i and y_i, $i \neq j$ from target and surrogate models, the term $\sum_{k=1}^{n} e^{c(x_i,y_k)/\tau}$ is employed.

Similarly, embeddings y_i and y_j from the same model form a negative pair for $i \neq j$, which are pushed apart by $\sum_{k=1,k\neq i}^{n} e^{c(y_i,y_k)/\tau}$, which consequently distinguishes embeddings of different nodes in the surrogate embedding space.

This contrastive objective serves two functions: it enforces the alignment of corresponding node embeddings across surrogate and target models, while simultaneously ensuring their differentiation from embeddings of different nodes within the surrogate model.

Additionally, the augmentations introduce corruption to the original graph within the input of the surrogate model, which generates novel node contexts to contrast with.

5.2 Spectral Graph Augmentations

One of our key observations is that the existing GNN stealing methods overlook the capability of GNN models to capture essential node information while filtering out irrelevant noise. This results in ineffective utilization of the information received from the victim model. We address it in our approach. In essence, we leverage the basic but essential fact that the victim model produces similar outputs for a given node and its augmented version. Thus, we train the surrogate model to align its predictions with the output of the victim model for both original and augmented nodes.

Specifically, we consider the graph structure of the GNN training data to develop an optimal augmentation strategy. During surrogate model training, we incorporate spectral graph augmentations initially introduced by [20]. The augmented graph is constructed by extracting information with varying frequencies from the original graph.

Let $\mathbf{D} = \text{diag}(d_1, \ldots, d_n)$ represent the node degree matrix, where $d_i = \sum_{j\in V} \mathbf{A}_{ij}$ is the degree of node $i \in V$. We define $\mathcal{L} = \mathbf{D} - \mathbf{A}$ as the unnormalized graph Laplacian of $\mathbf{G}$, and

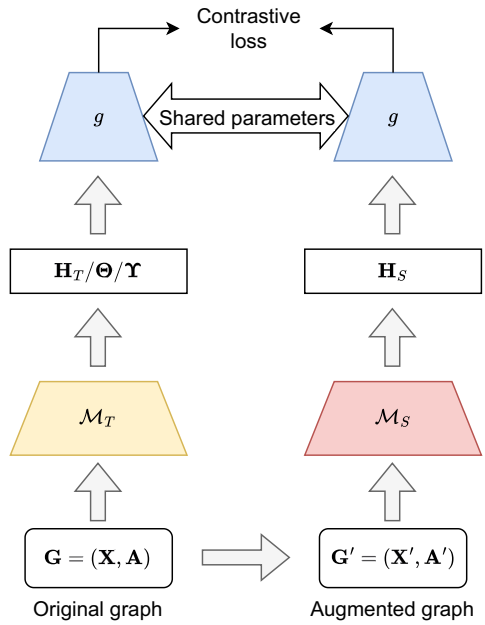

Figure 2. Overview of our model stealing attack against inductive GNNs.

$\hat{\mathcal{L}} = I_n - \hat{A} = D^{-\frac{1}{2}}(D - A)D^{-\frac{1}{2}}$ as the symmetric normalized graph Laplacian. Since $\hat{L}$ is symmetric normalized, its eigendecomposition is $U\Lambda U^\top$, where $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_N)$ and $U = [u_1^\top, \ldots, u_N^\top] \in \mathbb{R}^{N\times N}$ are the eigenvalues and eigenvectors of $\hat{L}$, respectively. Assuming $0 = \lambda_1 \leq \ldots \leq \lambda_N < 2$ (in which we approximately estimate $\lambda_N \approx 2$ [17]) , we call $\{\lambda_1, \ldots, \lambda_{\lfloor N/2 \rfloor}\}$ the amplitudes of low-frequency components and $\{\lambda_{\lfloor N/2 \rfloor+1}, \ldots, \lambda_N\}$ the amplitudes of high-frequency components. The graph spectrum, denoted as $\phi(\lambda)$, is defined by these amplitudes of different frequency components, indicating which parts of the frequency are enhanced or weakened.

For two random augmentations V_1 and V_2, their graph spectra are $\phi_{V_1}(\lambda)$ and $\phi_{V_2}(\lambda)$. Then, for all $\lambda_m \in [1, 2]$ and $\lambda_n \in [0, 1]$, V_1 and V_2 constitute an effective pair of graph augmentations if the following condition is met:

$$Optim_{V_1,V_2} : |\phi_{V_1}(\lambda_m) - \phi_{V_2}(\lambda_m)| > |\phi_{V_1}(\lambda_n) - \phi_{V_2}(\lambda_n)|. \tag{3}$$

Such a pair of augmentations is termed an *optimal contrastive pair* as in [20]. On a high level, ensuring that the difference in high-frequency components is greater than in low-frequency components helps the model to learn more robust and discriminative features.

Following this rule, we transform the adjacency matrix A to a new augmentation A', where A and A' form an optimal contrastive pair $Optim_{A,A'}$ for all $\lambda_m \in [1, 2]$ and $\lambda_n \in [0, 1]$. The new graph, with its adjacency matrix A', undergoes further augmentations such as feature masking, edge dropping, and edge perturbation. The result serves as the input for training the surrogate model. This form of augmentation improves the learning process of the surrogate model by boosting the effects of contrastive graph learning.

6 Experimental Evaluation

In this section, we conduct a thorough experimental analysis of a model stealing attack on inductive GNNs and show that our approach outperforms the state-of-the-art algorithm by Shen et al. [34]. In what follows, we introduce the experimental setup, evaluate Type I and Type II attacks from theft and reconnaissance perspectives, and

investigate the impact of different query budgets on attack performance. To our knowledge, ours and Shen et al. [34] are the only existing model stealing methods applicable to inductive GNN.

Datasets. We evaluate performance of our attack on 6 benchmark datasets for evaluating GNN performance [16, 11, 41]: DBLP [25], Pubmed [31], Citeseer Full (referred to as Citeseer) [9], Coauthor Physics (referred to as Coauthor) [33], ACM [39], and Amazon Copurchase Network for Photos (referred to as Amazon) [23]. We use these datasets to evaluate our attack's efficacy across different graph characteristics such as size, node features, and classes. Each dataset is split into three segments: the first 20% of randomly selected nodes for training the target model $\mathcal{M}_T$, the next 30% as the query graph $\mathbf{G}_Q$, and the final 50% as testing data for both $\mathcal{M}_T$ and $\mathcal{M}_S$. Further, we show that our attacks remain effective even with a reduced number of nodes in the query graph (see Section 6.3). Importantly, in the official implementation of [34] the testing data is sampled from the whole dataset, creating an overlap between the first two and the third data split. We conduct all experiments, including the state-of-the-art baseline by Shen et al. [34] on disjoint data sets.

Target and Surrogate Models. Following [34] we use GIN, GAT, and GraphSAGE as our target and surrogate models' architectures for evaluating our attack. We outline the model and training details for reproducibility purposes in Supplement [28] A and B.

Metrics. Following [14] taxonomy, we employ three metrics — accuracy, F1 score and fidelity — to assess our attack's performance. For the theft adversary, we gauge accuracy (correct predictions divided by total predictions) and F1 score (the harmonic mean of precision and recall), a widely-used metric in GNN node classification evaluation [16, 11, 37]. As the reconnaissance adversary aims to mimic the target model's behavior, we utilize fidelity (predictions agreed upon by both $\mathcal{M}_S$ and $\mathcal{M}_T$) as our evaluation metric [15, 14].

6.1 Evaluation of Type I Attacks

The overview of the accuracy and F1 score of the original node classification tasks for the target models are shown in Table 2. All analyzed GNN models perform well across all datasets, underscoring the effectiveness of jointly considering node features and graph structure for classification. Subsequently, the accuracy and fidelity results for Type I attacks are presented in Table 3. We focus on the results when the adversary targets the GIN model with the stealing attack. Similar performance patterns with GAT and Sage as target models are detailed in Supplement [28] due to space constraints.

Accuracy and F1 score. In our analysis of the target GIN model, a noteworthy surge in accuracy and F1 score is observed across all six datasets when utilizing various surrogate models for the prediction task. Particularly remarkable is the ACM dataset, where accuracy and F1 register a substantial increase of approximately 15 %pt. across all cases. This trend extends to the projection task, showcasing enhanced accuracy for most datasets, including DBLP, Pubmed, Citeseer, and Coauthor. Furthermore, our method exhibits a significant improvement in embedding accuracy across all datasets and surrogate models. Remarkably, our approach outperforms the baseline method for all tasks, datasets, and surrogate models, with the most pronounced benefits evident in the DBLP dataset.

Table 2. The performance of the original classification tasks on all 6 datasets using 3 different GNN structures.

Dataset	GIN		GAT		SAGE	
	Accuracy	F1	Accuracy	F1	Accuracy	F1
DBLP	0.772	0.713	0.759	0.693	0.781	0.715
Pubmed	0.844	0.839	0.821	0.816	0.856	0.852
Citeseer	0.834	0.836	0.836	0.838	0.838	0.839
Coauthor	0.912	0.907	0.941	0.923	0.943	0.924
ACM	0.890	0.890	0.897	0.896	0.905	0.905
Amazon	0.819	0.754	0.920	0.912	0.902	0.897

Fidelity. Examining the fidelity metrics for the target GIN model, we observe a consistent boost in prediction fidelity across all datasets and surrogate models, except for Pubmed, where our results closely align with those produced by the baseline. The most substantial increase in fidelity for the prediction task is witnessed in the ACM dataset, reaching nearly 20 %pt. with the GAT surrogate model. Additionally, there is an observable enhancement in projection fidelity for the majority of dataset and surrogate model configurations. In the embedding task, our method achieves increased fidelity across all datasets, except for the Pubmed dataset. Notably, for the DBLP and Coauthor datasets, our method consistently outperforms the baseline in terms of fidelity for all tasks, datasets, and surrogate models.

Stability. We conduct three runs for each combination, employing different graph partition seeds to evaluate how accuracy and fidelity values deviate from the average (indicated by standard deviation). The consistently low standard deviation values, as evident in Table 3, signify minimal fluctuation. This observation implies that the adversary can reliably steal from target models with statistically stable accuracy and fidelity.

6.2 Evaluation of Type II Attacks

Recall that for Type II attacks, the adversary must first construct an adjacency matrix $\mathbf{A}_Q$ for the query graph $\mathbf{G}_Q$ before querying the target model $\mathcal{M}_T$ and executing the model stealing attack. The outcomes of Type II attacks are outlined in Table 4. We present results for GIN as the target model; however, similar patterns for targeting GAT and GraphSAGE can be found in Supplement [28].

Accuracy and F1 score. Under the GIN target model, our approach consistently enhances prediction accuracy and F1 score across the majority of datasets and surrogate models. Noteworthy, the accuracy observed with the SAGE surrogate model on the Amazon dataset increases by 18.7 %pt. This trend persists in the context of projection accuracy, where our method outperforms competing configurations. Similarly, improvements in embedding accuracy and F1 score are evident, particularly on the Amazon dataset with the SAGE surrogate model, showcasing a remarkable 35.8 %pt. accuracy lead over the baseline.

Fidelity. In terms of accuracy fidelity, our method either surpasses or closely aligns with the baseline across the majority of configurations, with the most substantial enhancement observed on the Pubmed dataset. This trend extends to projection fidelity, where our approach exhibits notable advantage, particularly pronounced on the Pubmed dataset. In the realm of embedding fidelity, our method consistently matches or outperforms the baseline. Notably, with the SAGE surrogate model on the Amazon dataset, we achieve an increase in embedding fidelity from 46.3% to 81.1%.

Stability. Table 4 contains consistently low standard deviation values, highlighting the adversary's ability to steal target models with stable accuracy and fidelity in Type II attacks.

6.3 Query Budget

We explore the efficacy of our attack under varying query budgets, represented by different sizes of the query graph $\mathbf{G}_Q$. We compare the query efficiency of our approach with that of [34], focusing on the Citeseer dataset due to space constraints, though similar trends are

Table 3. The accuracy and fidelity scores of **Type I** attacks on all of the datasets with **GIN** as the target model.

Dataset	Task		Surrogate GIN			Surrogate GAT			Surrogate SAGE		
			Accuracy	Fidelity	F1 score	Accuracy	Fidelity	F1 score	Accuracy	Fidelity	F1 score
DBLP	Prediction	Shen et al.	0.738±0.001	0.805±0.000	0.675±0.002	0.715±0.003	0.782±0.004	0.620±0.002	0.735±0.001	0.799±0.002	0.679±0.001
		ours	**0.774±0.002**	**0.826±0.002**	**0.718±0.002**	**0.776±0.001**	**0.847±0.002**	**0.717±0.002**	**0.781±0.001**	**0.834±0.001**	**0.723±0.002**
	Projection	Shen et al.	0.705±0.007	0.751±0.000	0.569±0.004	0.695±0.016	0.757±0.020	0.549±0.011	0.704±0.031	0.755±0.038	**0.588±0.022**
		ours	**0.740±0.001**	**0.783±0.000**	**0.607±0.004**	**0.721±0.004**	**0.772±0.003**	**0.593±0.001**	**0.746±0.002**	**0.788±0.001**	0.579±0.004
	Embedding	Shen et al.	0.705±0.002	0.759±0.000	0.626±0.001	0.683±0.006	0.723±0.013	0.621±0.001	0.710±0.002	0.762±0.001	0.645±0.004
		ours	**0.770±0.000**	**0.806±0.004**	**0.804±0.001**	**0.774±0.004**	**0.827±0.005**	**0.836±0.002**	**0.755±0.009**	**0.796±0.005**	**0.787±0.002**
Pubmed	Prediction	Shen et al.	0.846±0.001	**0.912±0.001**	0.843±0.004	0.825±0.002	**0.906±0.001**	0.819±0.005	0.846±0.001	**0.914±0.000**	0.840±0.005
		ours	**0.856±0.001**	0.892±0.001	**0.847±0.004**	**0.845±0.001**	0.897±0.001	**0.835±0.004**	**0.859±0.002**	0.896±0.002	**0.852±0.001**
	Projection	Shen et al.	0.816±0.026	0.872±0.029	0.795±0.005	0.797±0.037	0.864±0.052	0.808±0.005	0.793±0.016	0.848±0.020	0.833±0.005
		ours	**0.848±0.001**	**0.883±0.000**	**0.841±0.009**	**0.840±0.002**	**0.896±0.001**	**0.824±0.004**	**0.855±0.002**	**0.889±0.002**	**0.840±0.002**
	Embedding	Shen et al.	0.853±0.002	**0.910±0.001**	0.847±0.002	0.834±0.001	**0.902±0.005**	0.829±0.002	0.846±0.001	**0.911±0.003**	0.839±0.002
		ours	**0.863±0.002**	0.888±0.003	**0.857±0.004**	**0.853±0.001**	0.898±0.003	**0.850±0.009**	**0.862±0.003**	0.879±0.005	**0.862±0.005**
Citeseer	Prediction	Shen et al.	0.773±0.001	0.818±0.001	0.777±0.002	0.725±0.007	0.766±0.009	0.741±0.001	0.762±0.002	0.794±0.003	0.768±0.001
		ours	**0.820±0.003**	**0.843±0.001**	**0.820±0.003**	**0.827±0.003**	**0.861±0.003**	**0.827±0.005**	**0.814±0.003**	**0.843±0.002**	**0.814±0.009**
	Projection	Shen et al.	0.668±0.050	0.688±0.051	0.681±0.001	0.704±0.012	0.733±0.015	0.754±0.005	**0.664±0.043**	**0.695±0.053**	0.599±0.002
		ours	**0.751±0.003**	**0.763±0.002**	**0.740±0.001**	**0.766±0.005**	**0.779±0.004**	**0.760±0.009**	0.631±0.006	0.648±0.008	**0.612±0.001**
	Embedding	Shen et al.	0.736±0.005	0.772±0.004	0.748±0.004	0.730±0.004	0.771±0.007	0.782±0.001	0.753±0.003	0.797±0.002	0.745±0.001
		ours	**0.810±0.003**	**0.827±0.001**	**0.815±0.001**	**0.828±0.000**	**0.852±0.000**	**0.837±0.002**	**0.813±0.004**	**0.818±0.011**	**0.810±0.001**
Amazon	Prediction	Shen et al.	0.869±0.002	0.808±0.000	0.821±0.001	0.759±0.021	0.722±0.028	0.671±0.010	0.842±0.012	0.801±0.016	0.789±0.002
		ours	**0.915±0.003**	**0.839±0.002**	**0.877±0.003**	**0.855±0.006**	**0.825±0.016**	**0.796±0.003**	**0.881±0.009**	**0.816±0.005**	**0.856±0.001**
	Projection	Shen et al.	0.609±0.055	**0.633±0.100**	0.512±0.012	0.606±0.052	**0.600±0.081**	0.559±0.001	**0.635±0.021**	**0.646±0.011**	0.540±0.010
		ours	**0.635±0.002**	0.616±0.007	**0.673±0.002**	**0.619±0.041**	0.591±0.042	**0.564±0.009**	0.524±0.015	0.538±0.014	**0.613±0.002**
	Embedding	Shen et al.	0.891±0.005	0.820±0.000	0.866±0.004	0.777±0.032	0.737±0.024	0.781±0.005	0.877±0.004	0.809±0.003	0.822±0.001
		ours	**0.919±0.000**	**0.828±0.005**	**0.887±0.001**	**0.901±0.009**	**0.825±0.010**	**0.834±0.001**	**0.893±0.007**	**0.823±0.002**	**0.861±0.004**
Coauthor	Prediction	Shen et al.	0.913±0.001	0.930±0.000	0.888±0.004	0.864±0.009	0.888±0.010	0.844±0.001	0.901±0.006	0.918±0.007	0.885±0.002
		ours	**0.942±0.002**	**0.950±0.002**	**0.926±0.009**	**0.921±0.002**	**0.942±0.003**	**0.876±0.003**	**0.942±0.000**	**0.957±0.001**	**0.923±0.009**
	Projection	Shen et al.	0.790±0.047	0.797±0.000	0.760±0.005	0.832±0.018	0.854±0.018	0.725±0.001	0.773±0.068	0.784±0.070	0.655±0.001
		ours	**0.920±0.004**	**0.926±0.004**	**0.888±0.002**	**0.881±0.013**	**0.896±0.012**	**0.866±0.001**	**0.824±0.008**	**0.836±0.008**	**0.675±0.009**
	Embedding	Shen et al.	0.887±0.004	0.902±0.000	0.852±0.004	0.878±0.000	0.891±0.002	0.869±0.002	0.910±0.005	0.922±0.005	0.882±0.004
		ours	**0.943±0.003**	**0.948±0.005**	**0.926±0.001**	**0.934±0.001**	**0.944±0.002**	**0.911±0.004**	**0.941±0.001**	**0.941±0.002**	**0.915±0.005**
ACM	Prediction	Shen et al.	0.733±0.007	0.759±0.007	0.736±0.004	0.725±0.040	0.744±0.051	0.741±0.004	0.750±0.008	0.793±0.007	0.738±0.004
		ours	**0.895±0.005**	**0.913±0.006**	**0.896±0.001**	**0.888±0.008**	**0.934±0.008**	**0.889±0.003**	**0.895±0.004**	**0.928±0.004**	**0.906±0.005**
	Projection	Shen et al.	**0.794±0.009**	**0.810±0.016**	0.712±0.004	**0.780±0.028**	0.791±0.036	0.737±0.005	0.801±0.039	**0.831±0.042**	0.847±0.004
		ours	0.770±0.011	0.798±0.008	**0.804±0.004**	0.757±0.004	**0.793±0.003**	**0.738±0.001**	**0.816±0.016**	0.823±0.020	**0.853±0.002**
	Embedding	Shen et al.	0.811±0.010	**0.836±0.008**	0.789±0.001	0.817±0.010	0.840±0.011	0.800±0.001	0.835±0.003	0.865±0.003	0.834±0.004
		ours	**0.842±0.007**	0.822±0.015	**0.882±0.003**	**0.857±0.009**	**0.870±0.009**	**0.845±0.001**	**0.882±0.009**	**0.894±0.011**	**0.874±0.003**

observed in other datasets. Accuracy and fidelity outcomes for both Type I and Type II attacks on the embedding task (see Figure 3) are presented for three surrogate models: GIN, GAT, and SAGE. Plots for prediction and projection tasks are provided in Supplement [28] E. We also compare our results to the accuracy and fidelity scores achieved by the best-performing surrogate model trained on 30% of dataset nodes using the method of [34]. Our method achieves the same or higher level of accuracy and fidelity for prediction and embedding responses using only 50% of the queries required in the baseline method. For the projection task, we can make a similar observation for the GAT surrogate model. Learning from the t-SNE projection requires using a number of queries closer to 30% to match the performance best surrogate model of the baseline when GIN or SAGE is selected as the surrogate model.

7 Discussion

Limitations. Firstly, the proposed model stealing attack is limited to scenarios where the target model provides node-level results. However, our method can be extended to graph-level and link prediction tasks by adjusting the augmentations (e.g., using node instead of link augmentations for link prediction). Secondly, the learning process of the missing graph structure was based on the approach presented in the paper [34]. Developing a novel technique for reconstructing graph structure, specifically designed for training the surrogate model with contrastive loss, constitutes a direction for further research.

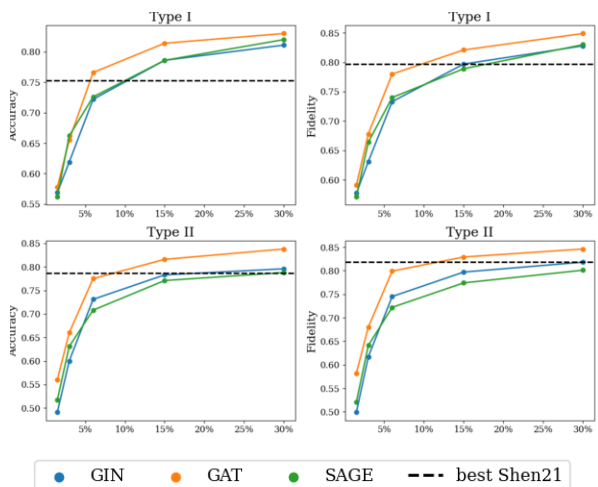

Figure 3. The average accuracies and fidelities on the Citeseer dataset and GIN target with embedding response using different percentages of nodes in the stealing process. We compare with the highest accuracy and fidelity achieved by [34] best-performing surrogate model using 30% of nodes.

Defense. Several active countermeasures against model stealing attacks have been proposed in the literature [5, 7, 35]. The applied formulation of the defense mechanism follows the approach by Shen and co-authors [34]. To address diverse query responses, we inject random Gaussian noise into node embeddings and t-SNE projections

Table 4. The accuracy and fidelity scores of **Type II** attacks on all of the datasets with **GIN** as the target model.

Dataset	Task		Surrogate GIN			Surrogate GAT			Surrogate SAGE		
			Accuracy	Fidelity	F1 score	Accuracy	Fidelity	F1 score	Accuracy	Fidelity	F1 score
DBLP	Prediction	Shen et al.	0.731±0.000	**0.822±0.000**	0.565±0.004	0.687±0.003	0.761±0.004	0.453±0.003	0.732±0.001	**0.817±0.000**	0.439±0.003
		ours	**0.747±0.001**	0.813±0.001	**0.628±0.004**	**0.746±0.005**	**0.815±0.003**	**0.682±0.005**	**0.741±0.005**	0.804±0.005	**0.648±0.003**
	Projection	Shen et al.	0.681±0.001	0.752±0.000	0.366±0.001	0.680±0.004	0.752±0.005	**0.518±0.003**	0.677±0.000	0.751±0.001	0.362±0.003
		ours	**0.706±0.003**	**0.767±0.003**	**0.574±0.001**	**0.697±0.007**	**0.753±0.005**	0.505±0.001	**0.693±0.006**	**0.756±0.011**	**0.579±0.002**
	Embedding	Shen et al.	0.720±0.006	**0.796±0.000**	0.540±0.005	0.682±0.003	0.751±0.001	0.540±0.004	**0.741±0.002**	**0.818±0.003**	0.605±0.005
		ours	**0.742±0.002**	0.785±0.002	**0.650±0.005**	**0.749±0.005**	**0.791±0.002**	**0.677±0.004**	0.704±0.011	0.744±0.009	**0.613±0.001**
Pubmed	Prediction	Shen et al.	0.786±0.001	0.831±0.000	0.738±0.001	0.731±0.005	0.785±0.003	0.704±0.001	0.785±0.001	0.820±0.000	0.691±0.003
		ours	**0.853±0.001**	**0.889±0.001**	**0.847±0.004**	**0.834±0.004**	**0.887±0.005**	**0.822±0.003**	**0.848±0.001**	**0.884±0.001**	**0.846±0.004**
	Projection	Shen et al.	0.814±0.002	0.874±0.000	0.716±0.001	0.798±0.001	0.871±0.001	0.801±0.001	0.822±0.000	**0.878±0.000**	0.687±0.003
		ours	**0.846±0.000**	**0.884±0.002**	**0.836±0.003**	**0.830±0.001**	**0.883±0.002**	**0.805±0.009**	**0.829±0.004**	0.866±0.005	**0.821±0.002**
	Embedding	Shen et al.	0.831±0.002	**0.887±0.000**	0.774±0.002	0.821±0.002	**0.889±0.006**	0.819±0.003	0.818±0.000	0.864±0.001	0.676±0.001
		ours	**0.855±0.001**	0.879±0.001	**0.857±0.001**	**0.849±0.002**	0.883±0.002	**0.839±0.004**	**0.850±0.002**	**0.868±0.003**	**0.842±0.001**
Citeseer	Prediction	Shen et al.	0.797±0.001	0.854±0.000	0.810±0.004	0.764±0.001	0.815±0.001	0.820±0.009	0.797±0.001	0.853±0.001	0.798±0.001
		ours	**0.813±0.002**	0.854±0.004	**0.822±0.001**	**0.840±0.005**	**0.887±0.004**	**0.836±0.001**	**0.813±0.003**	**0.856±0.005**	**0.813±0.003**
	Projection	Shen et al.	0.660±0.006	0.696±0.000	0.670±0.003	0.709±0.005	0.754±0.003	0.667±0.001	0.651±0.010	0.680±0.008	**0.675±0.004**
		ours	**0.727±0.005**	**0.747±0.002**	**0.714±0.009**	**0.757±0.002**	**0.784±0.001**	**0.731±0.001**	**0.667±0.048**	**0.691±0.051**	0.583±0.005
	Embedding	Shen et al.	0.780±0.002	0.817±0.000	0.791±0.009	0.770±0.002	0.796±0.004	0.760±0.001	0.787±0.003	**0.818±0.001**	0.803±0.001
		ours	**0.802±0.006**	**0.825±0.006**	**0.800±0.002**	**0.838±0.004**	**0.851±0.009**	**0.844±0.001**	**0.791±0.005**	0.800±0.005	**0.808±0.009**
Amazon	Prediction	Shen et al.	0.881±0.002	0.828±0.000	0.833±0.002	**0.876±0.002**	**0.839±0.005**	0.575±0.001	0.642±0.006	0.586±0.007	0.680±0.009
		ours	**0.911±0.005**	**0.833±0.002**	**0.900±0.002**	0.874±0.024	0.813±0.034	**0.882±0.003**	**0.829±0.008**	**0.775±0.014**	**0.774±0.001**
	Projection	Shen et al.	0.532±0.004	0.479±0.000	0.654±0.009	**0.480±0.116**	**0.455±0.122**	0.301±0.005	**0.500±0.009**	**0.461±0.005**	**0.688±0.001**
		ours	**0.661±0.028**	**0.602±0.029**	**0.673±0.001**	0.462±0.028	0.418±0.025	**0.492±0.002**	0.473±0.048	0.424±0.047	0.491±0.004
	Embedding	Shen et al.	0.744±0.005	0.683±0.000	0.847±0.001	0.863±0.002	0.842±0.001	0.619±0.004	0.525±0.006	0.473±0.004	0.725±0.002
		ours	**0.914±0.003**	**0.838±0.006**	**0.883±0.009**	**0.901±0.004**	**0.851±0.006**	**0.880±0.005**	**0.893±0.008**	**0.811±0.003**	**0.835±0.009**
Coauthor	Prediction	Shen et al.	0.942±0.000	**0.958±0.000**	0.910±0.005	**0.889±0.008**	**0.921±0.008**	0.910±0.009	0.944±0.000	**0.961±0.000**	0.931±0.005
		ours	**0.946±0.002**	0.952±0.001	**0.920±0.001**	0.888±0.002	0.917±0.002	0.910±0.001	0.944±0.000	0.953±0.001	**0.933±0.003**
	Projection	Shen et al.	0.848±0.002	0.856±0.000	0.876±0.005	**0.850±0.018**	**0.866±0.018**	**0.861±0.001**	0.813±0.001	0.824±0.001	0.822±0.002
		ours	**0.933±0.001**	**0.940±0.000**	**0.911±0.009**	0.734±0.008	0.750±0.009	0.850±0.002	**0.873±0.018**	**0.884±0.015**	**0.890±0.001**
	Embedding	Shen et al.	**0.946±0.001**	**0.959±0.000**	0.916±0.004	**0.923±0.001**	**0.944±0.001**	**0.930±0.004**	0.922±0.002	0.936±0.004	0.949±0.001
		ours	0.941±0.001	0.946±0.001	0.916±0.004	0.898±0.005	0.925±0.005	0.920±0.001	**0.936±0.004**	**0.941±0.003**	**0.960±0.003**
ACM	Prediction	Shen et al.	0.905±0.000	**0.945±0.000**	0.901±0.009	0.892±0.000	**0.959±0.000**	0.876±0.002	**0.897±0.000**	**0.943±0.000**	0.900±0.001
		ours	**0.907±0.005**	0.913±0.003	**0.904±0.002**	**0.900±0.002**	0.926±0.002	**0.894±0.001**	0.892±0.011	0.911±0.016	**0.910±0.002**
	Prediction	Shen et al.	0.835±0.000	0.858±0.000	0.884±0.002	0.872±0.000	0.920±0.000	0.859±0.003	0.876±0.000	0.904±0.000	0.879±0.001
		ours	**0.907±0.005**	**0.913±0.003**	**0.904±0.002**	**0.900±0.002**	**0.926±0.002**	**0.894±0.001**	**0.892±0.011**	**0.911±0.016**	**0.910±0.002**
	Embedding	Shen et al.	**0.906±0.000**	**0.935±0.000**	**0.910±0.005**	0.830±0.000	0.871±0.000	0.875±0.002	0.861±0.000	0.882±0.000	0.876±0.004
		ours	0.885±0.007	0.902±0.008	0.880±0.001	**0.882±0.009**	**0.914±0.008**	**0.879±0.002**	**0.868±0.006**	**0.887±0.001**	**0.884±0.001**

produced by target models, employing GIN as the surrogate model to assess the countermeasure's efficacy. The surrogate model's accuracy, serving as a metric for attack performance within the defined threat model, is evaluated on the ACM dataset (refer to Figure 4 for the embedding response and Supplement [28] F for prediction and projection responses). Our findings indicate a marginal impact of random Gaussian noise on the surrogate model's accuracy, similar to the results obtained by [34]. Even with a high standard deviation of the noise added, the model stealing attack still results in a useful surrogate model. Moreover, injecting random noise into the target's model responses results in a lowered utility of the model for legitimate users of the model API, as outlined in [6]. Developing robust active defense mechanisms continues to be a focus for future research. However, we note the existence of several methods for protecting the Intellectual Property after the GNN model has already been stolen [30, 40, 44, 47, 45, 4].

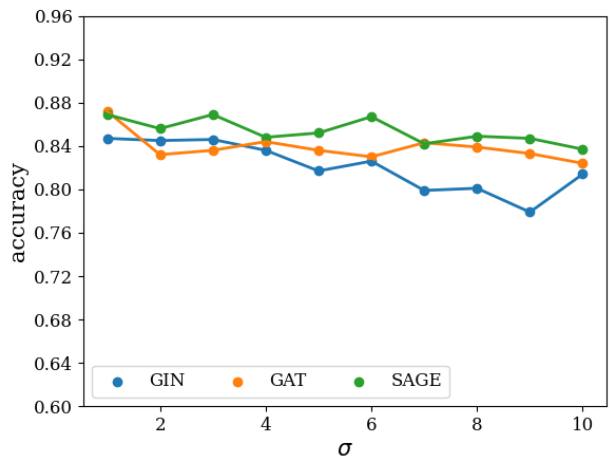

Figure 4. Accuracy scores after adding Gaussian noise with a standard deviation σ to embedding response from target model GIN on the ACM dataset.

8 Conclusions

In this work, we identify a new unsupervised GNN model-stealing attack. Only a single work [34] targeting inductive GNNs existed previously. Our proposed approach combines a contrastive learning objective with graph-specific augmentations, improving the efficient utilization of information derived from target model outputs. Specifically, we applied spectral graph augmentations to create varied graph perspectives for the surrogate model within a contrastive learning framework at the node level. Thus, the considered method seamlessly aligns with the overarching framework proposed by [34], serving as an enhanced approach for training surrogate models. This adaptation significantly enhances the efficiency and overall performance of the model-stealing process. The conducted comparative analysis against the state of the art showed that the proposed stealing technique demonstrates superior performance while requiring fewer queries of the target model.

References

[1] E. Alsentzer, S. Finlayson, M. Li, and M. Zitnik. Subgraph neural networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 8017–8029. Curran Associates, Inc., 2020.

[2] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR, 13–18 Jul 2020.

[3] Y. Chen, L. Wu, and M. J. Zaki. Iterative Deep Graph Learning for Graph Neural Networks: Better and Robust Node Embeddings. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2020.

[4] E. Dai, M. Lin, and S. Wang. Pregip: Watermarking the pretraining of graph neural networks for deep intellectual property protection. *arXiv preprint arXiv:2402.04435*, 2024.

[5] J. Dubiński, S. Pawlak, F. Boenisch, T. Trzcinski, and A. Dziedzic. Bucks for buckets (b4b): Active defenses against stealing encoders. *Advances in Neural Information Processing Systems*, 36, 2024.

[6] A. Dziedzic, N. Dhawan, M. A. Kaleem, J. Guan, and N. Papernot. On the difficulty of defending self-supervised learning against model extraction. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5757–5776. PMLR, 17–23 Jul 2022.

[7] A. Dziedzic, M. A. Kaleem, Y. S. Lu, and N. Papernot. Increasing the cost of model extraction with calibrated proof of work, 2022.

[8] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin. Graph neural networks for social recommendation. In *The World Wide Web Conference*, WWW '19, page 417–426, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450366748.

[9] C. L. Giles, K. D. Bollacker, and S. Lawrence. CiteSeer: An Automatic Citation Indexing System. In *International Conference on Digital Libraries (ICDL)*, pages 89–98. ACM, 1998.

[10] A. Grover and J. Leskovec. node2vec: Scalable Feature Learning for Networks. In *ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pages 855–864. ACM, 2016.

[11] W. L. Hamilton, Z. Ying, and J. Leskovec. Inductive Representation Learning on Large Graphs. In *Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1025–1035. NIPS, 2017.

[12] X. He, J. Jia, M. Backes, N. Z. Gong, and Y. Zhang. Stealing Links from Graph Neural Networks. In *USENIX Security Symposium (USENIX Security)*, pages 2669–2686. USENIX, 2021.

[13] X. He, R. Wen, Y. Wu, M. Backes, Y. Shen, and Y. Zhang. Node-Level Membership Inference Attacks Against Graph Neural Networks. *CoRR abs/2102.05429*, 2021.

[14] M. Jagielski, N. Carlini, D. Berthelot, A. Kurakin, and N. Papernot. High Accuracy and High Fidelity Extraction of Neural Networks. In *USENIX Security Symposium (USENIX Security)*, pages 1345–1362. USENIX, 2020.

[15] M. Juuti, S. Szyller, S. Marchal, and N. Asokan. PRADA: Protecting Against DNN Model Stealing Attacks. In *IEEE European Symposium on Security and Privacy (Euro S&P)*, pages 512–527. IEEE, 2019.

[16] T. N. Kipf and M. Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[17] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[18] K. Krishna, G. S. Tomar, A. P. Parikh, N. Papernot, and M. Iyyer. Thieves on Sesame Street! Model Extraction of BERT-based APIs. In *International Conference on Learning Representations (ICLR)*, 2020.

[19] J. Lee, I. Lee, and J. Kang. Self-Attention Graph Pooling. In *International Conference on Machine Learning (ICML)*, pages 3734–3743. PMLR, 2019.

[20] N. Liu, X. Wang, D. Bo, C. Shi, and J. Pei. Revisiting graph contrastive learning from the perspective of graph spectrum. In *Advances in Neural Information Processing Systems*, volume 35, pages 2972–2983. Curran Associates, Inc., 2022.

[21] Y. Liu, J. Jia, H. Liu, and N. Z. Gong. Stolenencoder: Stealing pre-trained encoders in self-supervised learning, 2022.

[22] Y. Liu, R. Wen, X. He, A. Salem, Z. Zhang, M. Backes, E. D. Cristofaro, M. Fritz, and Y. Zhang. ML-Doctor: Holistic Risk Assessment of Inference Attacks Against Machine Learning Models. In *USENIX Security Symposium (USENIX Security)*. USENIX, 2022.

[23] J. J. McAuley, C. Targett, Q. Shi, and A. van den Hengel. Image-Based Recommendations on Styles and Substitutes. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 43–52. ACM, 2015.

[24] T. Orekondy, B. Schiele, and M. Fritz. Knockoff Nets: Stealing Functionality of Black-Box Models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4954–4963. IEEE, 2019.

[25] S. Pan, J. Wu, X. Zhu, C. Zhang, and Y. Wang. Tri-Party Deep Network Representation. In *International Joint Conferences on Artifical Intelligence (IJCAI)*, pages 1895–1901. IJCAI, 2016.

[26] N. Papernot, P. D. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical Black-Box Attacks Against Machine Learning. In *ACM Asia Conference on Computer and Communications Security (ASIACCS)*, pages 506–519. ACM, 2017.

[27] B. Perozzi, R. Al-Rfou, and S. Skiena. DeepWalk: Online Learning of Social Representations. In *ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pages 701–710. ACM, 2014.

[28] M. Podhajski, J. Dubiński, F. Boenisch, A. Dziedzic, A. Pregowska, and T. Michalak. Efficient model-stealing attacks against inductive graph neural networks, 2024. URL https://arxiv.org/abs/2405.12295.

[29] M. Ringsquandl, H. Sellami, M. Hildebrandt, D. Beyer, S. Henselmeyer, S. Weber, and M. Joblin. Power to the relational inductive bias: Graph neural networks in electrical power grids. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 1538–1547, 2021. ISBN 9781450384469.

[30] V. Sai Pranav Bachina, A. Gangwal, A. Ajay Sharma, and C. Sharma. Genie: Watermarking graph neural networks for link prediction. *arXiv e-prints*, pages arXiv–2406, 2024.

[31] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective Classification in Network Data. *AI Magazine*, 2008.

[32] Z. Sha, X. He, N. Yu, M. Backes, and Y. Zhang. Can't steal? cont-steal! contrastive stealing attacks against image encoders, 2023.

[33] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann. Pitfalls of Graph Neural Network Evaluation. *CoRR abs/1811.05868*, 2018.

[34] Y. Shen, X. He, Y. Han, and Y. Zhang. Model stealing attacks against inductive graph neural networks, 2021.

[35] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart. Stealing Machine Learning Models via Prediction APIs. In *USENIX Security Symposium (USENIX Security)*, pages 601–618. USENIX, 2016.

[36] R. van den Berg, T. N. Kipf, and M. Welling. Graph Convolutional Matrix Completion. *CoRR abs/1706.02263*, 2017.

[37] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph Attention Networks. In *International Conference on Learning Representations (ICLR)*, 2018.

[38] B. Wang and N. Z. Gong. Stealing Hyperparameters in Machine Learning. In *IEEE Symposium on Security and Privacy (S&P)*, pages 36–52. IEEE, 2018.

[39] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu. Heterogeneous Graph Attention Network. In *The Web Conference (WWW)*, pages 2022–2032. ACM, 2019.

[40] J. Xu, S. Koffas, O. Ersoy, and S. Picek. Watermarking graph neural networks based on backdoor attacks. In *IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, pages 1179–1197. IEEE, 2023.

[41] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How Powerful are Graph Neural Networks? In *International Conference on Learning Representations (ICLR)*, 2019.

[42] K. Yang, K. Swanson, W. Jin, C. Coley, P. Eiden, H. Gao, A. Guzman-Perez, T. Hopper, B. Kelley, M. Mathea, A. Palmer, V. Settels, T. Jaakkola, K. Jensen, and R. Barzilay. Analyzing learned molecular representations for property prediction. *Journal of Chemical Information and Modeling*, 59(8):3370–3388, 2019. PMID: 31361484.

[43] Q. Yang, Y. Liu, T. Chen, and Y. Tong. Federated Machine Learning: Concept and Applications. *ACM Transactions on Intelligent Systems and Technology*, 2019.

[44] X. You, Y. Jiang, J. Xu, M. Zhang, and M. Yang. Gnnfingers: A fingerprinting framework for verifying ownerships of graph neural networks. In *Proceedings of the ACM on Web Conference*, pages 652–663, 2024.

[45] X. You, Y. Jiang, J. Xu, M. Zhang, and M. Yang. Gnnguard: A fingerprinting framework for verifying ownerships of graph neural networks. In *The Web Conference*, 2024.

[46] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen. Graph Contrastive Learning with Augmentations. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2020.

[47] L. Zhang, M. Xue, L. Y. Zhang, Y. Zhang, and W. Liu. An imperceptible and owner-unique watermarking method for graph neural networks. In *ACM Turing Award Celebration Conference*, pages 108–113, 2024.

[48] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang. Deep Graph Contrastive Representation Learning. In *ICML Workshop on Graph Representation Learning and Beyond*, 2020.