On Stealing Graph Neural Network Models

Marcin Podhajski[1,2*], Jan Dubiński[3,4], Franziska Boenisch[5], Adam Dziedzic[5], Agnieszka Pręgowska[1], Tomasz Paweł Michalak[6,7]

[1]Institute of Fundamental Technological Research, Polish Academy of Sciences
[2]IDEAS NCBR
[3]Warsaw University of Technology
[4]NASK National Research Institute
[5]CISPA Helmholtz Center for Information Security
[6]University of Warsaw
[7]IDEAS Research Institute

Abstract

Current graph neural network (GNN) model-stealing methods rely heavily on queries to the victim model, assuming no hard query limits. However, in reality, the number of allowed queries can be severely limited. In this paper, we demonstrate how an adversary can extract a GNN with very limited interactions with the model. Our approach first enables the adversary to obtain the model backbone without making direct queries to the victim model and then to strategically utilize a fixed query limit to extract the most informative data. The experiments on eight real-world datasets demonstrate the effectiveness of the attack, even under a very restricted query limit and under defense against model extraction in place. Our findings underscore the need for robust defenses against GNN model extraction threats.

Code — https://github.com/m-podhajski/OnStealingGNNs
Extended version — https://arxiv.org/pdf/2511.07170

1 Introduction

Graph Neural Networks have recently become the center of attention in many dynamically developing Artificial Intelligence-based technologies. They have been successfully applied to various tasks involving graph-structured data: node classification, link prediction, graph classification, and recommendation systems (Sharma, Singh, and Ratna 2024). Unfortunately, like all types of neural networks, GNNs are vulnerable to various security threats, including adversarial attacks (Ma, Ding, and Mei 2020), data poisoning (Nguyen Thanh et al. 2023), model inversion (Zhang et al. 2022), and privacy breaches (Guan et al. 2024; Olatunji, Nejdl, and Khosla 2021).

Particularly, GNN models are vulnerable to *model-stealing attacks* (Tramèr et al. 2016; Jagielski et al. 2020; Dziedzic et al. 2022a). In such attacks, an adversary with query access to a target (victim) model can replicate its functionality by training a local surrogate model on query-response pairs. A typical defense against such an attack is to limit the number of queries. However, previous studies on GNN model stealing (Podhajski et al. 2024; Shen et al. 2022; Zhuang et al. 2024; Wu et al. 2021) generally assume access to a relatively large number of queries, focusing on maximizing performance as the budget increases. This overlooks the practical reality that in many applications, adversaries must operate under severe query restrictions.

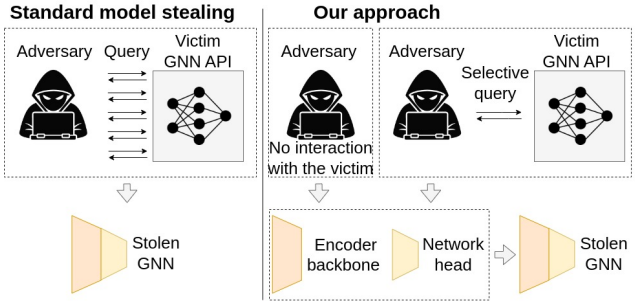

Figure 1: Standard GNN model stealing vs. our approach. Conventional model stealing methods extract the entire GNN through extensive querying of the victim model API. Our method divides this process into stages, focusing on maximizing the stealing outcome within a restricted query limit. First, we show that the adversary can obtain the encoder backbone locally, without any interaction with the victim API. Then, the adversary performs query selection using the representations from the extracted encoder and extracts the network head via selective querying. This enables effective model stealing under strict query budgets, demonstrating that the GNN model stealing threat is significantly more severe than previously assumed.

Our method challenges this conventional approach (summarized visually in Figure 1) and shows that a GNN model can be effectively extracted even when the adversary faces strict constraints on the number of queries allowed to the victim model. It first enables the adversary to recover the model backbone without querying the victim directly, then strategically uses a fixed query budget to extract the most informative data for effective model stealing.

Collectively, our results demonstrate that model-stealing attacks on GNNs are both more effective and resource-

*Corresponding author: marcin.podhajski@ideas-ncbr.pl

efficient than previously understood, highlighting the critical severity of these attacks in both inductive and transductive scenarios. For instance, targeting a SAGE (Hamilton, Ying, and Leskovec 2017) model trained on the Physics dataset, we achieve 91% accuracy with only 100 queries to the victim model —– compared to approximately 5,000 queries, $\sim 15\times$ higher computational cost, and additional victim output (such as embeddings) needed by the current state-of-the-art method to reach similar accuracy.

In summary, our contributions are as follows:

- We identify a threat that has been previously overlooked in research: the ability to steal a GNN model even under a significant limit on access to the victim.
- We investigate this threat by showing that an adversary with access to query data, but no direct access to the model, can locally obtain a high-quality encoder at a low computational cost.
- We further demonstrate that an adversary with restricted access to the victim model can strategically select queries to train the model head, resulting in a stolen model with improved accuracy and fidelity.

2 Background

We first introduce the basic notions considered in this work and move to the motivation behind our method.

2.1 Graph Neural Networks

Graph Neural Networks are a class of neural architectures that use the graph structure $\mathbf{A}$ and node features $\mathbf{X}$ as inputs. They are utilized across tasks such as node classification (Kipf and Welling 2017), link prediction (Perozzi, Al-Rfou, and Skiena 2014; Grover and Leskovec 2016), and graph/subgraph classification (Alsentzer et al. 2020).

Our work addresses model-stealing vulnerabilities in GNNs, specifically for node-level tasks. For such tasks, GNNs are typically trained and evaluated in two scenarios crucial to understanding these vulnerabilities: transductive and inductive settings. In the transductive paradigm, the data consists of a fixed graph with a portion of nodes labeled for training, while another portion remains unlabeled. The objective is to apply these labeled nodes to predict labels for the previously unlabeled ones within the same graph. However, transductive models are limited in their ability to generalize to novel nodes. Conversely, in the inductive setting, GNNs are designed to expand their learning to accommodate unseen nodes or graphs that were not part of the training data.

2.2 Model Stealing Attacks

Model stealing attacks aim to replicate the functionality of a victim model f_v trained on a private dataset $\mathbf{G}_V$. An attacker with black-box access to the victim model first selects a query graph $\mathbf{G}_Q$ from their available data. For each query node v_i from $\mathbf{G}_Q$, the attacker obtains the corresponding output $y_i = f_v(v_i)$ from the victim model. The attacker then constructs a surrogate training dataset, $\{(v_i, y_i)\}$, which is then used to train a surrogate model f_s that mimics the behavior of f_v. Model extraction attacks have been demonstrated to be successful against various types of models, including classifiers (Jagielski et al. 2020; Tramèr et al. 2016) and encoders (Dziedzic et al. 2022a; Sha et al. 2023).

	Trans.	Ind.	Data Acc.	Vict. Q. Acc.	Vict. Output
Zhuang et al.	✓	✓	None	Unlimited	Only Pred.
Wu et al.	✓	✗	Limited	Unlimited	Only Pred.
Podhajski et al.	✗	✓	Limited	Unlimited	Emb. & Pred.
Shen et al.	✗	✓	Limited	Unlimited	Emb. & Pred.
Ours	✓	✓	Limited	Limited	Only Pred.

Table 1: Existing stealing attacks on node-level GNNs.

Relation to existing work. An adversary attempting to steal a model faces two primary challenges: limited data access or restricted access to the model itself. Existing approaches to GNN model stealing predominantly address scenarios with limited or no data access, but none explore the problem of severely restricted model access. Specifically, Shen et al. and Podhajski et al. concentrate on stealing the encoder in the inductive setting, assuming unlimited access to the model but limited access to data. Both of these methods leverage the victim model's output (e.g., embeddings) to train the encoder and then train Multi-Layer Perceptrons (MLPs) on top of the frozen encoder using available class labels. This requirement for intermediate embeddings contrasts with our work, which assumes a more restrictive and practical threat model where adversaries only receive the final class predictions. Similarly, Wu et al. focuses on limited data access in the transductive setting, while also assuming unrestricted access to the victim. Another contribution is the data-free model extraction attack framework proposed by Zhuang et al.. This framework enables GNN model extraction without requiring access to actual node features or graph structures. However, it assumes unlimited access to the victim model and relies on an extensive number of queries.

A comprehensive comparison of various GNN model-stealing methods, including the assumptions regarding data and model access, is presented in Table 1, highlighting the diverse methodologies and their limitations.

2.3 Self-Supervised Learning in GNNs

Self-supervised learning (SSL) for GNNs trains models on data without explicit labels, focusing on node or graph representations. Representative methods include Deep Graph Infomax (DGI) by Veličković et al. (2019), Latent Graph Prediction (LaGraph) by Xie, Xu, and Ji (2022), and Bootstrapped Graph Latents (BGRL) by Thakoor et al. (2022).

In the context of model-stealing attacks, the capability of learning useful feature representations from unlabeled data may be crucial when the adversary has access to surplus data, *i.e.*, additional data points beyond the victim model query limit. Thus, SSL techniques can leverage this surplus data to refine and enhance the stolen model's performance. This means that SSL can be a powerful tool for overcoming constraints imposed on traditional model-stealing methods. SSL allows adversaries to bypass limitations imposed by direct querying of the victim model. By training on unlabeled

Setting	Dataset	Random	SSL-Trained	Gain	Method
Inductive	Reddit	93.3	94.0	0.7	DGI
	PPI	62.6	63.8	1.2	DGI
	WikiCS	78.9	79.9	1.0	BGRL
	Computer	86.5	90.3	3.8	BGRL
	Photo	92.0	93.1	1.1	BGRL
	CS	91.6	93.3	1.7	BGRL
	Physics	93.7	95.7	2.0	BGRL
Transductive	Cora	69.3	82.3	13.0	DGI
	Citeseer	61.9	71.8	9.9	DGI
	Pubmed	69.6	76.8	7.2	DGI

Table 2: Comparison of test accuracies using a randomly initialized GCN encoder with a trained MLP head vs. a fully SSL-trained GCN encoder in both inductive and transductive settings, as reported by DGI (Veličković et al. 2019) and BGRL (Thakoor et al. 2022).

data and utilizing self-supervised objectives, SSL can produce feature representations that are robust and informative, even without extensive interaction with the victim model. This is especially beneficial when direct access to the victim model is limited or restricted.

2.4 Motivation

Existing works on inductive GNN model stealing, such as those by Shen et al. and Podhajski et al., rely on stealing the victim's encoder using rich responses like query embeddings. However, a key observation from self-supervised learning provides a more efficient, query-free path for the adversary. In the inductive graph learning setting, Veličković et al. (2019) showed that a randomly initialized graph convolutional network (GCN) can already extract informative features and serve as a strong baseline. This phenomenon is linked to the Weisfeiler-Lehman graph isomorphism test (Weisfeiler and Lehman 1968), as discussed by Kipf and Welling (2017); Hamilton, Ying, and Leskovec (2017). As shown by recent works, a randomly initialized GNN encoder, when paired with a trained MLP head, can achieve performance comparable to that of a fully trained model. For instance, DGI (Veličković et al. 2019) reports test accuracies of 93.3% (random encoder) vs. 94.0% (SSL-trained) on Reddit, and 62.6% vs. 63.8% on PPI. Similarly, BGRL (Thakoor et al. 2022) observes minimal performance gaps across datasets such as WikiCS, Computer, Photo, CS, and Physics. These results are summarized in Table 2. Our experimental findings align with these insights. T-SNE visualizations (in the extended version) suggest that even randomly initialized encoders produce structured embeddings in the inductive setting.

In contrast, the transductive setting sees greater benefits from self-supervised learning. As shown in Table 2, training a GNN encoder with SSL yields notable performance gains. Since the full graph structure (including test node connectivity) is available during training, the encoder can learn representations that better align with the downstream task – unlike in the inductive setting, where such structural information is unavailable (Veličković et al. 2019; Thakoor et al. 2022). These improvements are further supported by the visualizations in the extended version. Notably, transductive graphs are typically small, keeping SSL training costs low and making it a viable option even with limited computational resources.

In the context of model stealing, these observations suggest that in both settings, in practice, the encoder can be obtained without interacting with the victim model—either by using a randomly initialized backbone (inductive) or by training an encoder locally via SSL (transductive). We confirm this hypothesis empirically in Section 4.1. Moreover, this approach benefits from low computational requirements. Randomly initialized encoders in the inductive setting and lightweight SSL training in the transductive case enable adversaries to operate under limited resources. This highlights the practicality and severity of model-stealing threats under realistic constraints.

With the encoder part of the model fixed, the central challenge becomes selecting informative queries to obtain the model head. We hypothesize that naive random query sampling is insufficient to expose the victim model's decision boundaries. We address this by leveraging the encoder's representations to guide query selection. Specifically, we apply clustering to the node embeddings and choose representative nodes near cluster centroids. This ensures coverage across the input space and increases the likelihood that each query contributes new information, as confirmed in Section 4.2.

3 Method

In this section, we outline our proposed approach, which is designed to extract GNN models even under restrictive query limits to the victim model. The scheme of our proposed approach is illustrated in Figure 2. First, we acquire the encoder backbone, which serves as a feature extractor, independently of the victim model. This step involves constructing a pre-trained encoder network to generate meaningful embeddings of the input data without requiring any interaction with the victim model. The encoder backbone is crucial as it provides a robust foundation for representation learning, capturing rich semantic features from the data. Next, we focus on selecting the optimal queries to interact with the victim model. This involves leveraging the embeddings generated by the encoder to identify a set of inputs that are most informative or representative of the data distribution. Finally, we extract the model head by training an MLP on top of the encoder output. The training process utilizes class-label responses to the selected queries. The result is a reconstructed model head that, when combined with the encoder backbone, approximates the victim model.

3.1 Threat Model

To introduce the necessary research methodology, we describe the threat model, outlining the attack setting as well as the adversary's goal and capabilities.

Attack Setting. Our research operates in a challenging *black-box* scenario where the adversary has no knowledge of the target GNN model's parameters, architecture, or the training graph $\mathbf{G}_V$. Our investigation focuses on GNNs that

produce node-level query responses, taking node v as input and providing the corresponding class label. We consider a query limit q_n representing the maximum number of node predictions the adversary can obtain from the victim model.

Adversary's Goal. Referring to the taxonomy defined by (Jagielski et al. 2020), adversaries' goals fall into two categories, *i.e.*, *theft* and *reconnaissance*. The *theft adversary* aims to construct a surrogate model f_s matching f_v on the target task (Tramèr et al. 2016; Papernot et al. 2017), violating the intellectual property in the victim model. In contrast, the *reconnaissance adversary* seeks a surrogate model f_s mirroring f_v across all inputs. This high-fidelity match serves as a tool for subsequent attacks, such as crafting adversarial examples without direct queries to f_v (Papernot et al. 2017).

Adversary's Capabilities. First, we assume that the adversary has access to a graph $\mathbf{G}_D$ representing their own available (unlabeled) dataset. Next, we assume that the adversary queries a target model f_v hidden behind a publicly accessible API (Tramèr et al. 2016; Orekondy, Schiele, and Fritz 2019; He et al. 2021a,b), receiving responses $\mathbf{R}$ based on an input query graph $\mathbf{G}_Q$, which is a subgraph of $\mathbf{G}_D$. The response $\mathbf{R}$ has a size of at most q_n. We consider only class labels as responses, reflecting the most common API outputs encountered in real-world scenarios. Finally, we assume that the graph $\mathbf{G}_D = (\mathbf{X}_D, \mathbf{A}_D)$ is drawn from the same distribution as the graph $\mathbf{G}_V$, which is used for training the target model f_v (which is a standard assumption in the literature (Shen et al. 2022; Podhajski et al. 2024; Wu et al. 2021; Tramèr et al. 2016; Jagielski et al. 2020)). In practice, we consider $\mathbf{G}_D$ and $\mathbf{G}_V$ to come from the same distribution if they are sampled from the same dataset.

3.2 Obtaining the Encoder

In the initial stage of our method, we deliberately avoid making any queries to the victim model. Instead, we focus on obtaining the encoder part of the model locally, allowing us to thoroughly analyze and understand the data without prematurely utilizing resources on victim model queries. The encoder obtained in this step serves a twofold purpose: 1) we reuse the encoder as a component of the surrogate model, and 2) it allows us to comprehend the data and select the best possible queries in the subsequent step of our method.

In the inductive setting, we start from the observation that various studies on SSL training, including DGI (Veličković et al. 2019), BGRL (Thakoor et al. 2022), and LaGraph (Xie, Xu, and Ji 2022), report that a randomly initialized encoder (particularly using GCN architecture) often yields results comparable to those of an SSL-trained encoder. Our observations align with these findings. We hypothesize that a randomly initialized encoder can often achieve similar performance to an SSL-trained encoder. The results presented in the experimental section positively verify this hypothesis. This leads to a novel paradigm for the inductive setting in which a randomly initialized model can achieve results comparable to those of stolen models without needing direct queries to the victim model. On top of this, using our approach, even without significant computational resources, one can obtain a good encoder for large graph datasets.

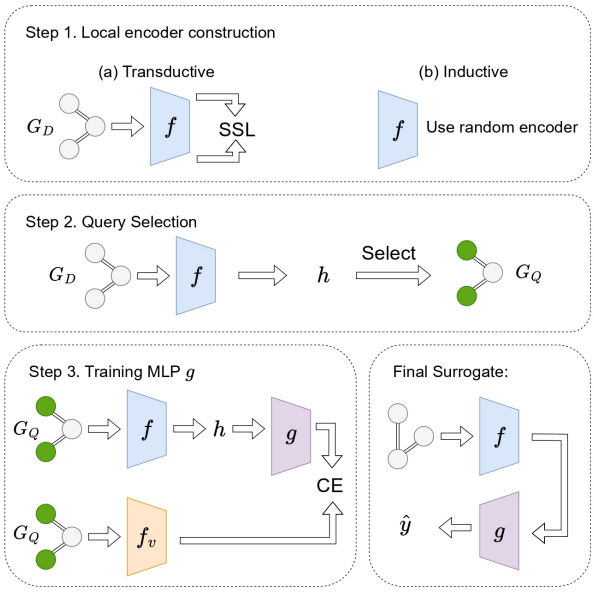

Figure 2: Proposed Approach. We first train an encoder using SSL on the adversary's full data G_D in the transductive setting, or use a random encoder in the inductive setting. Using embeddings from this encoder, we select queries G_Q. Finally, we train an MLP and combine it with the encoder to form the surrogate model.

In the transductive setting, the randomly initialized GCN was observed to have lower performance (Veličković et al. 2019). Therefore, in this setting, we utilize an SSL approach. We note that in the transductive setting, the graph used for training is typically small, enabling us to train the encoder locally via SSL without extensive resource consumption. This is advantageous because it ensures that our initial model training is efficient and cost-effective. Additionally, this procedure ensures that the adversary utilizes all of the data points that it has access to, regardless of the strict limit of queries to the victim.

Our approach is tailored to be compatible with any SSL framework that produces an encoder capable of generating meaningful representations. This flexibility allows us to integrate our approach with a variety of existing SSL techniques, ensuring broad applicability and effectiveness. In particular, in our numerical experiments, as the SSL framework we employ LaGraph (Xie, Xu, and Ji 2022). In the inductive setting, as our encoder, we applied the GCN architecture following the approach proposed by Xie, Xu, and Ji (2022). In the transductive setting, we use the SSL approach, which is effective for a range of backbone GNNs. For our experiments, we utilize both GCN, GIN (Xu et al. 2019), and SAGE architectures.

3.3 Query Selection

In the final step, we aim to select a data subset that maximizes information extraction from the victim model within

the query limit. Intuitively, our goal is to select the most diverse subset possible, ensuring comprehensive coverage of the dataset accessible to the adversary. This strategy of selecting representative points to cover the input space is conceptually analogous to diversity sampling methods in the field of active learning (Settles 2009; Tsitsulin et al. 2023). This selected subset is then used to train an MLP head on top of the encoder, enabling the construction of a high-quality surrogate model.

Graph data $\mathbf{G}_D$ available to the adversary consists of a set of attributes $\mathbf{X}_D \in \mathcal{R}^{n*d}$ and a graph structure $\mathbf{A}_D \in \{0,1\}^{n\times n}$, where n is the number of nodes and d is the number of features. To cover the dataset well, we want to map $\mathbf{X}_D$ and $\mathbf{A}_D$: $f(\mathbf{X}_D, \mathbf{A}_D)$ into a space that will contain information about both parts. For this purpose, we can use an encoder f that maps the attributes and structure of the graph into the $\mathbf{H} \in \mathcal{R}^{n*b}$ embedding space, where b is the embedding size. Note that for the inductive setting, we use a randomly initialized encoder, and for the transductive setting, we use an SSL-trained model. The training set for the SSL-encoder comprises all data available to the adversary, denoted as the set of nodes $\{v_1, ..., v_n\}$.

To choose a subset of nodes that covers the embedding space $\mathbf{H}$, we use the K-means algorithm (Lloyd 1982). We partition the embeddings $\{h_1, ..., h_n\} \subset \mathbf{H}$ of nodes $\{v_1, ..., v_n\}$ into q_n clusters, each containing at least one node. The number of clusters q_n is set equal to the victim model's query limit, corresponding to the number of queries we are allowed to send to the target. Next, we select one node from each cluster (specifically, the one whose embedding is closest to the cluster centroid), forming the set of nodes $\{v'_1, ..., v'_{q_n}\}$ for our query graph $\mathbf{G}_Q$. Finally, we query the victim model f_v and obtain predictions $\{y_1, ..., y_{q_n}\}$.

In the last step, we train an MLP component g to predict a class y based on the representation h using Cross Entropy (CE) loss. The training set of g consists of the chosen embeddings $\{h'_1, ..., h'_{q_n}\} = \{f(v'_1), ..., f(v'_{q_n})\}$ and labels returned by the victim $\{y_1, ..., y_{q_n}\}$. The final surrogate model f_s consists of the encoder f and a head g which makes predictions based on a graph structure $\mathbf{A}$ and attributes $\mathbf{X}$: $\hat{y} = f_s(\mathbf{X}, \mathbf{A}) = g(f(\mathbf{X}, \mathbf{A}))$.

4 Empirical Evaluation

We evaluate our proposed approach across eight benchmark datasets to demonstrate its effectiveness and to highlight the vulnerability of GNNs to model-stealing attacks in both transductive and inductive settings. The experimental setup is described in detail in the extended version.

4.1 Stealing with a Random Encoder and Self-Supervised Learning

We first consider the inductive setting and investigate the efficacy of using a randomly initialized encoder. Specifically, we compare two approaches:

- training a multi-layer perceptron on top of a randomly initialized and frozen encoder (**R-init**), and
- training the entire GNN architecture, including both the encoder and the head, in an end-to-end manner (**E2E**).

The results of this comparison, as presented in Table 3 and the extended version, include the accuracy and fidelity of both approaches, reported along with their standard deviations. These evaluations were conducted using randomly selected queries, without leveraging any query selection algorithm. The results reveal that the performance of the frozen encoder with an MLP is comparable to that of the *E2E* approach across most datasets, as well as when using both GAT and SAGE target models. Additionally, the extended version present charts that confirm these results hold across different query limits. Importantly, the R-init method requires significantly fewer computational resources, as only the MLP layer needs training, whereas the *E2E* approach involves updating the entire network. We used T-SNE to visualize embeddings from the Physics dataset, as shown in the extended version. The 2D plot compares embeddings from a random encoder and a trained encoder. Clusters corresponding to node classes are clearly separated, with the random encoder producing well-defined clusters closely resembling those of the trained encoder.

In addition to the inductive setting, we also explore the transductive setting by evaluating the impact of self-supervised learning on encoder performance. Beyond the *E2E* approach, we consider an SSL-trained encoder paired with an MLP (**SSL**). The results, summarized in Table 4 and the extended version, demonstrate a significant improvement in accuracy and fidelity when using SSL compared to relying solely on the query data set. This improvement is particularly notable in datasets such as Cora and Citeseer and is observed consistently across different target models (GCN and GAT) as well as surrogate architectures (GIN, GCN, and SAGE). The supplementary material confirms this result across different query limits, showing that the performance improvement from SSL becomes more pronounced as the query limit decreases. In addition, the findings suggest that incorporating all available adversarial data in local self-supervised training yields substantial performance gains across all benchmarks. This underscores the utility of SSL in extracting high-quality representations, even in extreme scenarios, enhancing the overall robustness and effectiveness of the model-stealing process.

4.2 Strategic Query Selection

Finally, we evaluate the ultimate step of the method proposed in both the transductive setting (see Table 4 and the extended version) and the inductive setting (see Table 3 and the extended version). For comparison, we use:

- a randomly chosen set of queries (**Random**), and
- a set of queries selected using our method (**Select**).

The results show that selecting queries with K-means based on the encoder embeddings results in higher accuracy and fidelity in both setups across all of the datasets and target models. The results also indicate that in the transductive setting, our method provides a selection of more diverse queries based on the SSL-trained embeddings. For the inductive setting, it is further confirmed that the representations produced by the random encoder are meaningful and

Method	Reddit		CS		Physics		Photo		WikiCS	
	Acc.	**Fid.**	**Acc.**	**Fid.**	**Acc.**	**Fid.**	**Acc.**	**Fid.**	**Acc.**	**Fid.**
Target accuracy	94.8		93.9		96.0		93.0		72.5	
E2E	47.0±4.5	47.0±4.4	73.6±3.9	74.5±3.9	89.9±1.1	91.1±1.3	81.2±0.8	83.3±1.7	61.6±1.3	71.0±2.5
R-init + Random	76.9±4.0	77.0±3.9	74.2±2.2	74.5±2.3	86.0±1.5	87.8±1.5	85.0±1.7	87.8±1.8	63.0±2.0	71.4±2.4
Shen et al. (2022)	77.2*±5.1	77.0*±4.5	77.7*±0.8	78.7*±0.7	90.6*±0.5	91.6*±0.6	84.4*±0.8	86.3*±0.8	64.9*±1.0	**81.0***±0.9
Podhajski et al. (2024)	79.9*±4.1	79.5*±4.4	78.0*±0.5	78.1*±0.4	89.9*±0.2	89.1*±0.3	84.0*±1.0	84.2*±1.2	64.0*±1.1	70.0*±0.5
Zhuang et al. (2024)	13.6±4.1	19.5±3.2	24.8±2.8	27.1±3.9	55.5±5.0	54.9±4.6	24.9±2.8	24.9±3.2	38.6±2.1	40.8±2.0
R-init + Select (ours)	**82.5**±1.2	**82.7**±1.2	**78.4**±2.1	**79.2**±2.2	**91.2**±0.4	**92.7**±0.5	**86.8**±1.0	**89.8**±0.9	**65.5**±1.8	73.6±1.9

Table 3: Inductive setting (target: SAGE, surrogate: GCN (same for all methods), $q_n = 100$). Accuracy (Acc.) and Fidelity (Fid.) are reported as mean ± std. dev. in percentage over 3 runs. Methods marked with * assume access to victim embeddings.

Surrogate	Method	Cora		Citeseer		Pubmed	
		Acc.	**Fid.**	**Acc.**	**Fid.**	**Acc.**	**Fid.**
	Target accuracy	83.3		72.1		80.0	
GIN	*E2E* (Wu et al. 2021)	39.1±9.7	38.8±8.0	37.8±3.1	40.1±5.5	57.1±3.1	66.0±3.0
	Zhuang et al. (2024)	17.7±4.4	21.0±5.1	23.3±2.8	26.0±4.0	35.1±2.2	36.6±4.0
	SSL + Random	52.7±4.7	51.5±6.1	63.3±2.1	56.5±3.5	69.1±4.3	72.5±4.0
	SSL + Select (ours)	57.7±2.9	57.8±2.3	65.6±1.0	71.5±1.2	69.9±3.1	76.0±3.0
SAGE	*E2E* (Wu et al. 2021)	46.2±2.2	35.8±5.1	27.8±4.9	25.9±7.6	64.1±0.9	63.1±6.1
	Zhuang et al. (2024)	11.5±3.0	10.9±3.1	13.9±2.9	12.3±4.6	36.9±2.7	41.0±3.0
	SSL + Random	40.8±7.5	42.9±7.4	44.5±7.1	50.5±6.8	62.8±3.5	65.4±3.0
	SSL + Select (ours)	46.8±5.6	46.5±4.5	45.7±4.5	47.7±7.1	62.8±1.6	69.4±1.2
GCN	*E2E* (Wu et al. 2021)	47.5±3.7	45.7±1.0	37.2±6.1	41.1±7.5	61.0±4.9	67.5±5.0
	Zhuang et al. (2024)	18.1±2.7	21.1±3.9	22.1±3.3	23.1±3.8	33.2±2.9	33.4±3.0
	SSL + Random	56.1±2.7	56.8±3.0	51.3±5.1	57.6±5.5	66.1±7.3	72.7±9.0
	SSL + Select (ours)	**69.9**±1.2	**72.5**±1.3	**66.3**±1.9	**72.4**±2.3	**67.0**±6.0	**80.1**±4.7

Table 4: Transductive setting (target: GCN, surrogates: GIN, SAGE, GCN, $q_n = 10$). Accuracy (Acc.) and Fidelity (Fid.) reported as mean ± std. dev. in percentage over 3 runs.

possible to interpret. By comparing the accuracy of our surrogate models with the performance of the victim models, we observe that the surrogate models achieve comparable results with significantly fewer labeled data. Additionally, figures in the extended version illustrate how performance improves with different query limits, further demonstrating that as the query limit decreases, the effectiveness of our method increases. To justify the use of our approach in the *Select* phase, we compare K-means with several representative selection strategies, including farthest-first, K-center greedy, entropy sampling, coreset herding, and margin sampling (Settles 2009; Scheffer, Decomain, and Wrobel 2001; Welling 2009; Gonzalez 1985). As shown in the extended version, while methods such as coreset herding offer improvements over random selection, K-means consistently delivers the highest accuracy and fidelity across all datasets.

To quantitatively assess the properties of the selected query set, we perform an experiment measuring the fraction of class coverage per query. Based on the average of 100 runs on the CS dataset (see the extended version), we observe that our selection method covers a greater number of classes than random sampling under small budgets, with both approaches converging to full class coverage as the query limit increases. Additionally, the extended version presents a T-SNE projection illustrating the distribution of nodes selected by our query strategy.

We also conduct McNemar's test (McNemar 1947), as presented in the extended version, to compare the stolen model with the original model by evaluating their classification errors on the same dataset. The null hypothesis assumes no significant difference in the classification error rates between the stolen and original models. The results demonstrate that our method produces a stolen model with a higher degree of similarity to the victim model.

4.3 Comparison with Existing Methods

We thoroughly compare our method with existing approaches for stealing inductive and transductive GNNs.

Inductive setting. We compare our method against all existing approaches, *i.e.*, Shen et al. (2022) and Podhajski et al. (2024). We note that these two previous approaches do not take into account scenarios where the adversary's dataset exceeds the query limit. Thus, to explore this scenario, we replicate the experiments described in Shen et al. (2022) and Podhajski et al. (2024) with randomly selected queries from the entire available dataset. Additionally, we compare our

performance with Zhuang et al. (2024), which demonstrates that an adversary can successfully steal a model without access to any training data, though this requires a substantial number of queries: 100 queries for graphs of size 250, totaling 25,000 query nodes. We show that our method performs much better in a scenario with very restricted model access (we evaluate our approach on 10, 25, 50, 100, and 500 nodes). We present our results in Table 3 and the extended version, where we compare all methods using the same surrogate model architecture and different target architectures: SAGE and GAT, with a query limit q_n of 100. We also compare the performance of all methods across different query limits q_n in the extended version. Furthermore, in the extended version, we compare the performance using the architectures originally employed in Shen et al. (2022) and Podhajski et al. (2024). Our method, which does not rely on query embeddings from the victim, consistently outperforms the previous methods in both accuracy and fidelity. This superiority is evident across various baseline surrogate models, including GIN, GAT, and SAGE. Additionally, by utilizing the randomly initialized encoder, we significantly improved the execution times (the extended version). Moreover, even when the same architecture is used, our approach yields significantly better results, demonstrating its effectiveness.

Transductive setting. For the transductive setting, there is no previous work in the literature to which we can directly compare. However, we believe that in the work by Wu et al. (2021) we can find a setting to which a comparison of our *E2E* is reasonable. In particular, unlike our work, Wu et al. (2021) focuses on the scenarios where the adversary lacks some types of data, e.g., the graph structure or node attributes. In contrast, our focus is on scenarios where the adversary is not limited in terms of data type but in terms of the number of queries that can be made. To create a relevant comparison, we modified the approach by Wu et al. (2021) by training their model using randomly selected queries under the assumption that it has access to the entire data available to the adversary (with no restrictions on data types). Similarly, we compare our results to Zhuang et al. (2024), which generates graph queries in a restricted scenario, where the query limit is reduced to just 5, 10, 20, or 50 nodes. The results presented in Table 4 and the extended version show a significant improvement in both accuracy and fidelity when our method is applied. Specifically, when the surrogate encoder is trained with all available data and the query selection is optimized using the K-means clustering algorithm, the performance of the stolen model is greatly enhanced. This demonstrates the advantage of our approach in transductive settings, where strategic data utilization and query selection can significantly boost the attack's effectiveness. We further validate the effectiveness of our method across different surrogate architectures (GIN, SAGE, and GCN) and observe that our method consistently outperforms all baselines regardless of the surrogate used. Notably, the GCN surrogate yields the highest performance overall, surpassing all other methods and architectural configurations. Additionally, the extended version shows how the performance improves with different query limits, further emphasizing the impact of our method as the query limit decreases.

5 Defense

In this paper, we demonstrate that our method can successfully perform a model extraction even under very restrictive conditions, where the victim model returns only the predicted class and there are a limited number of queries that the adversary can make. Although numerous defenses against model extraction have been proposed in the literature (Tramèr et al. 2016; Dziedzic et al. 2022b,a; Jiang et al. 2024), they typically assume a less restrictive setup. For example, if the victim model returns the embeddings, a possible defense is to inject a random Gaussian noise into the node embeddings (the setup studied by Podhajski et al. (2024); Shen et al. (2022); Dubiński et al. (2023)). Similarly, when returning class probabilities, some methods (Jiang et al. 2024; Orekondy, Schiele, and Fritz 2020) perturb the output distribution. In contrast, in our hard-label setting, a possible defense (Tramèr et al. 2016) is based on changing the class prediction (for example, with some probability p). However, such defenses come with the cost of reducing the accuracy of the model.

We evaluate all stealing methods under a defense that flips predictions with a probability of $p = 10\%$ in both inductive and transductive settings (see the extended version). Our proposed method consistently achieves the highest performance, even in the presence of defense measures. These results highlight the difficulty of defending against our extraction attack, even under strong defense mechanisms that come at the cost of the utility of the model.

6 Conclusions

In this paper, we studied and challenged traditional approaches to stealing GNN models within the frameworks of both inductive and transductive settings.

We examined existing GNN model theft methods in the **inductive setting**, which typically involve the extraction of an encoder and training of an MLP using class labels. Our analysis reveals that models initialized randomly can yield results comparable to those trained with additional responses (e.g., embeddings) from the victim. This approach demonstrates the potential for effective model extraction, even with low computational resources and restricted victim access.

We extend our analysis to the **transductive setting**, showing that when the adversary's data surpasses the query limit, the excess can be used via SSL to boost the stolen model's performance. Our results demonstrate that even with limited queries, leveraging extra data significantly improves the stolen model's accuracy and fidelity.

Additionally, in **both settings**, we studied how the adversary can take advantage of having more data than the query limit. We demonstrated that by strategically selecting optimal nodes for querying, it is possible to significantly enhance the accuracy and fidelity of the stolen model.

We believe that our research offers novel insights into GNN model-stealing techniques across both inductive and transductive frameworks. The results obtained not only contribute to a deeper understanding of these attacks but also highlight the urgent need for improved security measures against such adversarial strategies.

Acknowledgments

This research was supported by the Polish National Science Centre (NCN) within grant no. 2023/51/I/ST6/02854. This work was also supported by the German Research Foundation (DFG) within the framework of the Weave Programme under the project titled "Protecting Creativity: On the Way to Safe Generative Models" with number 545047250. We also gratefully acknowledge support from the Initiative and Networking Fund of the Helmholtz Association in the framework of the Helmholtz AI project call under the name "PAFMIM", funding number ZT-I-PF-5-227. Responsibility for the content of this publication lies with the authors.

References

Alsentzer, E.; Finlayson, S.; Li, M.; and Zitnik, M. 2020. Subgraph Neural Networks. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 8017–8029. Curran Associates, Inc.

Dubiński, J.; Pawlak, S.; Boenisch, F.; Trzciński, T.; and Dziedzic, A. 2023. Bucks for Buckets (B4B): Active Defenses Against Stealing Encoders. arXiv:2310.08571.

Dziedzic, A.; Dhawan, N.; Kaleem, M. A.; Guan, J.; and Papernot, N. 2022a. 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*, 5757–5776. PMLR.

Dziedzic, A.; Kaleem, M. A.; Lu, Y. S.; and Papernot, N. 2022b. Increasing the Cost of Model Extraction with Calibrated Proof of Work. arXiv:2201.09243.

Gonzalez, T. F. 1985. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38: 293–306.

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

Guan, F.; Zhu, T.; Zhou, W.; and Choo, K. R. 2024. Graph neural networks: a survey on the links between privacy and security. *Artificial Intelligence Review*, 57(2): 40.

Hamilton, W. L.; Ying, R.; and Leskovec, J. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1025–1035.

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

He, X.; Wen, R.; Wu, Y.; Backes, M.; Shen, Y.; and Zhang, Y. 2021b. Node-Level Membership Inference Attacks Against Graph Neural Networks. arXiv:2102.05429.

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

Jiang, W.; Li, H.; Xu, G.; Zhang, T.; and Lu, R. 2024. A Comprehensive Defense Framework Against Model Extraction Attacks. *IEEE Transactions on Dependable and Secure Computing*, 21(2): 685–700.

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

Lloyd, S. P. 1982. Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, 28: 129–136.

Ma, J.; Ding, S.; and Mei, Q. 2020. Towards more practical adversarial attacks on graph neural networks. *Advances in neural information processing systems*, 33: 4756–4766.

McNemar, Q. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2): 153–157.

Nguyen Thanh, T.; Quach, N. D. K.; Nguyen, T. T.; Huynh, T. T.; Vu, V. H.; Nguyen, P. L.; Jo, J.; and Nguyen, Q. V. H. 2023. Poisoning GNN-based recommender systems with generative surrogate-based attacks. *ACM Transactions on Information Systems*, 41(3): 1–24.

Olatunji, I. E.; Nejdl, W.; and Khosla, M. 2021. Membership inference attack on graph neural networks. In *2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, 11–20. IEEE.

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

Orekondy, T.; Schiele, B.; and Fritz, M. 2020. Prediction Poisoning: Towards Defenses Against DNN Model Stealing Attacks. In *ICLR*.

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

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

Podhajski, M.; Dubiński, J.; Boenisch, F.; Dziedzic, A.; Pregowska, A.; and Michalak, T. P. 2024. Efficient Model-Stealing Attacks Against Inductive Graph Neural Networks. In *The 27th European Conference on Artificial Intelligence (ECAI 2024)*, Frontiers in Artificial Intelligence and Applications. IOS Press.

Scheffer, T.; Decomain, C.; and Wrobel, S. 2001. Active hidden Markov models for information extraction. In *Advances in Intelligent Data Analysis*, 309–318. Springer.

Settles, B. 2009. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin-Madison, Madison, WI, USA.

Sha, Z.; He, X.; Yu, N.; Backes, M.; and Zhang, Y. 2023. Can't Steal? Cont-Steal! Contrastive Stealing Attacks Against Image Encoders. In *2023 IEEE/CVF Conference*

on Computer Vision and Pattern Recognition (CVPR). IEEE Computer Society.

Sharma, A.; Singh, S.; and Ratna, S. 2024. Graph Neural Network Operators: a Review. *Multimedia Tools and Applications*, 83(8): 23413–23436.

Shen, Y.; He, X.; Han, Y.; and Zhang, Y. 2022. Model Stealing Attacks Against Inductive Graph Neural Networks. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE.

Thakoor, S.; Tallec, C.; Azar, M. G.; Azabou, M.; Dyer, E. L.; Munos, R.; Veličković, P.; and Valko, M. 2022. Large-scale representation learning on graphs via bootstrapping. *International Conference on Learning Representations (ICLR)*.

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

Tsitsulin, A.; Perozzi, B.; Esfandiari, H.; Kazemi, M.; Bateni, M. H.; Mirrokni, V.; and Ramachandran, D. 2023. Tackling Provably Hard Representative Selection via Graph Neural Networks. *Transactions on Machine Learning Research*.

Veličković, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2019. Deep Graph Infomax. In *International Conference on Learning Representations*.

Weisfeiler, B.; and Lehman, A. A. 1968. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsia*, 2(9): 12–16.

Welling, M. 2009. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*.

Wu, B.; Yang, X.; Pan, S.; and Yuan, X. 2021. Model Extraction Attacks on Graph Neural Networks: Taxonomy and Realization. arXiv:2010.12751.

Xie, Y.; Xu, Z.; and Ji, S. 2022. Self-Supervised Representation Learning via Latent Graph Prediction. In Chaudhuri, K.; Jegelka, S.; Song, L.; Szepesvari, C.; Niu, G.; and Sabato, S., eds., *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 24460–24477. PMLR.

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

Zhang, Z.; Liu, Q.; Huang, Z.; Wang, H.; Lee, C.-K.; and Chen, E. 2022. Model inversion attacks against graph neural networks. *IEEE Transactions on Knowledge and Data Engineering*.

Zhuang, Y.; Shi, C.; Zhang, M.; Chen, J.; Lyu, L.; Zhou, P.; and Sun, L. 2024. Unveiling the Secrets without Data: Can Graph Neural Networks Be Exploited through Data-Free Model Extraction Attacks? In *33rd USENIX Security Symposium (USENIX Security 24)*, 5251–5268. Philadelphia, PA: USENIX Association. ISBN 978-1-939133-44-1.