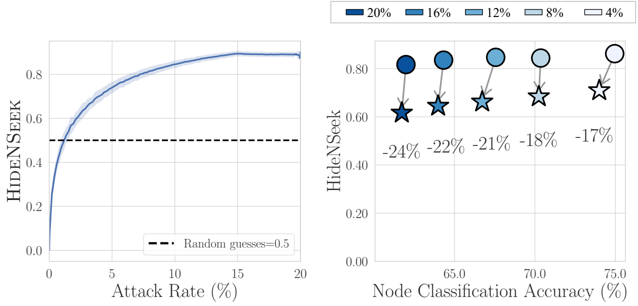
Online Appendix

A ADDITIONAL ANALYSIS REGARDING BYPASSABILITY

In this section, we shall analyze more about the bypassability of HideNSeek to further validate HideNSeek is not easily bypassable. As defined in Property 1, for a measure to be non-bypassable, there should be an *inevitable trade-off between noticeability and the attack performance* for adaptive attacks. Thus, we investigate the trade-off on adaptive attacks defined in Def. 1 by modulating the level of adaptiveness. In addition to the adaptive attacks defined in Def. 1 (which are *filtering-based*), to enrich the empirical evaluation, we further develop adaptive attacks based on *loss-based* methods. Below, we will use the term "filtering-based adaptive attacks" to refer to the adaptive attacks defined in Def. 1 and we will describe *loss-based* adaptive attacks below.

Filtering-based adaptive attacks. We adjust the level of adaptiveness of filtering-based adaptive attacks by adjusting the number Δ_C of attack edge candidates with K = PGD Attack and attack rate $\gamma = 10\%$. Specifically, we set the candidate budget as $\Delta_C = k\Delta$ and adjust the value of k. Recall that the filtering-based adaptive attacks in Fig. 1(b) are designed as follows: (1) we first choose Δ_C candidate attacks using the same attack method used to generate $\hat{G}$; (2) then, among the Δ_C attacks, we greedily choose Δ edges one by one to minimize the measure U. To demonstrate that reducing the noticeability would lead to the loss of attack performance, we compare the result while increasing k from 4 (which is used for the results in Fig. 1b) to 20. For each k value, we run 5 trials and report the mean noticeability (HideNSeek AUROC score) and mean node classification accuracy after the evasion attack. As shown in Fig. 2, we can observe a trade-off between the noticeability (y-axis) and the attack performance (x-axis) as k increases, validating HideNSeek is not easily bypassable.



(a) HideNSeek is sensitive **(b) HideNSeek is less bypassable**

Figure 1: (left) HideNSeek is sensitive: HideNSeek notices attacks immediately, even for adaptive attacks with a low attack rate, while the existing measures fail to do so; (right) HideNSeek is less bypassable: when the adaptive attack is adopted, the noticeability of HideNSeek reduces less, compared to the existing measures. For an attacker to further lower the noticeability w.r.t. HideNSeek, it is inevitable to sacrifice the attack performance (refer to Fig. 2).

Loss-based adaptive attack. Based on the powerful gradient-based attack method, Metattack, we design a *loss-based* adaptive attack out of it. Specifically, we add a regularizing term to the original loss function of Metattack, to make the attack powerful and unnoticeable.

Let us first introduce the details of Metattack. Let $V = [n] = \{1, 2, \ldots, n\}$ denote the node set, and let $f((u, v); G)$ denote an edge scoring function trained on the clean graph G. Metattack learns the attacked adjacency matrix $\hat{A}$ that would hinder the node classification model through meta-learning. Specifically, for every iteration, Metattack picks an edge that is the most influential to the meta-loss, which means that the edge is the most crucial to the performance of the node classification model.

The noticeability of attacks is not considered in the original Metattack does. To enhance Metattack so that it can also generate unnoticeable attacks, we adjust the meta-loss so that each selected edge is not only a crucial but also an unnoticeable one. Specifically, we add a regularization term so that unnoticeable edges are preferred, as follows:

$$\mathcal{L}_U = \sum_{u,v \in V} (1 - 2A_{uv}) \cdot f((u, v); G) \cdot \Delta A_{uv},$$

where ΔA_{uv} is Metattack's current change (i.e., attack) on A_{uv}. Then, let the original loss of Metattack be $\mathcal{L}_M$, we adjust the loss as follows:

$$\mathcal{L} = \mathcal{L}_M + \lambda \mathcal{L}_U,$$

where λ is the regularization coefficient. To demonstrate that reducing the noticeability would impair attack performance, we compare the result while increasing the λ from 0 to 100. For each λ value, we run 5 trials and report the mean noticeability (HideNSeek AUROC score) and mean node classification accuracy after the poisoning attack. As shown in Fig. 2, the attack performance decreases as the weight λ of the unnoticeability regularization term increases.

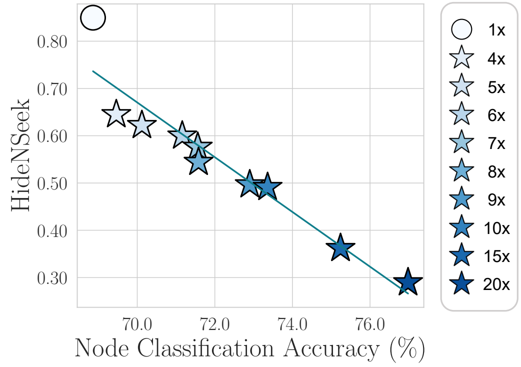


Figure 2: The trade-off between noticeability and attack performance on filtering-based adaptive attacks. Circle markers (○) represent original attacks, while star markers (☆) represent adaptive attacks. The attack performance has to be sacrificed to lower the noticeability, i.e., HideNSeek is not easily bypassable.

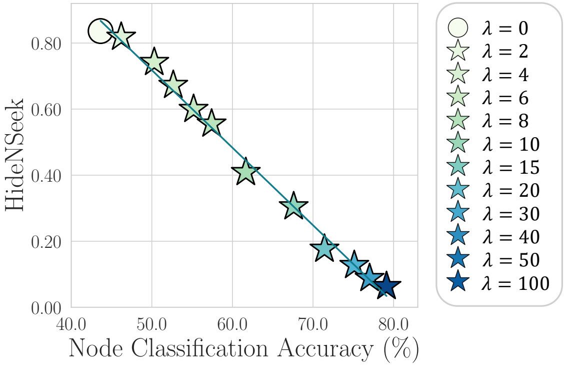


Figure 3: The trade-off between noticeability and attack performance on loss-based adaptive attacks. Circle markers (○) represent original attacks, while star markers (☆) represent loss-based adaptive attacks. The attack performance has to be sacrificed to lower the noticeability even for stronger adaptive attack, i.e., HideNSeek is not easily bypassable.

B COMPLEXITY ANALYSIS

In this section, we prove the time and space complexity of the noticeability measures (DegreeKS, ClsCoefKS, DegreeLR, and HomophKS).

B.1 Time Complexity

DegreeKS. The process of calculating DegreeKS involves calculating the degree of all nodes in both G and $\hat{G}$ and performing the KS test between them. The time complexity of computing the degree in G and $\hat{G}$ is $O(m)$ and $O(\hat{m})$, respectively. The time complexity of the KS test is dominated by sorting the degrees of all nodes, which is $O(n \log n)$. Therefore, the total time complexity is $O(n \log n + m + \hat{m})$.

ClsCoefKS. The process of calculating ClsCoefKS involves calculating the clustering coefficient of all nodes in both G and $\hat{G}$ and performing the KS test between them. The time complexity of computing the clustering coefficient in G and $\hat{G}$ is $O(m^{1.48})$ and $O(\hat{m}^{1.48})$ with the AYZ-algorithm [1]. Like DegreeKS, the time complexity of KS test is $O(n \log n)$, then the total time complexity is $O(nlogn + m^{1.48} + \hat{m}^{1.48})$.

DegreeLR. The process of calculating DegreeLR involves calculating the degree of all nodes in both G and $\hat{G}$ and performing the LR test on them. The time complexity of computing the degree in G and $\hat{G}$ is $O(m)$ and $O(\hat{m})$, respectively. The time complexity of the LR test is dominated by estimating the scaling parameter of the power-law distribution [13], which is $O(n)$. Therefore, the total time complexity is $O(n + m + \hat{m})$.

HomophKS. The process of calculating HomophKS includes calculating the homophily score of all nodes in both G and $\hat{G}$ and performing the KS test on them. The homophily score of each node is determined by calculating the cosine similarity between the node feature and the aggregated node feature generated by averaging the node features of its neighbors [4]. The time complexity to compute the aggregated node feature is $O(kd_{avg})$, where d_{avg} is the average degree of a graph. The time complexity of calculating cosine similarity between the node feature and the aggregated node feature is $O(k)$. Hence, the time complexity of calculating the homophily score for each node is $O(k + kd_{avg})$. Consequently, the time complexity of calculating the homophily score for all nodes is $O(n(k+kd_{avg}))$, where d_{avg} for G and $\hat{G}$ is represented respectively by m/n and $\hat{m}/n$. Therefore, the time complexity of computing the homophily score for all nodes in G and $\hat{G}$ is $O(k(n + m))$ and $O(k(n + \hat{m}))$, respectively. As mentioned previously in DegreeKS, the KS test has a time complexity of $O(n \log n)$. Therefore, the total time complexity is $O(n \log n + k(n + m + \hat{m}))$.

B.2 Space Complexity

DegreeKS. The process of calculating DegreeKS involves calculating the degree of all nodes in both G and $\hat{G}$ and performing the KS test between them. The space complexity of storing G and $\hat{G}$ is $O(n + m)$ and $O(n + \hat{m})$, respectively. The space complexity of computing the degree in G and $\hat{G}$ is $O(n)$. The space complexity of the KS test is dominated by storing the degrees of all nodes to perform sorting, which is $O(n)$. Therefore, the total space complexity is $O(n + m + \hat{m})$.

CluCoefKS. The process of calculating ClsCoefKS involves calculating the clustering coefficient of all nodes in both G and $\hat{G}$ and performing the KS test between them. The space complexity of storing G and $\hat{G}$ is $O(n + m)$ and $O(n + \hat{m})$, respectively. The space complexity of computing the clustering coefficient in G and $\hat{G}$ is $O(m^{1.05})$ and $O(\hat{m}^{1.05})$ with the AYZ-algorithm and the Strassen algorithm [1, 3]. The space complexity of the KS test is $O(n)$. Therefore, the total space complexity is $O(n + m^{1.05} + \hat{m}^{1.05})$.

DegreeLR. The process of calculating DegreeLR involves calculating the degree of all nodes in both G and $\hat{G}$ and performing the LR test on them. The space complexity of storing G and $\hat{G}$ is $O(n + m)$ and $O(n + \hat{m})$, respectively. The space complexity of computing the degree in G and $\hat{G}$ is $O(n)$. The space complexity of the LR test is dominated by storing the degrees of all nodes to estimate the scaling parameter of the power-law distribution [13], which is $O(n)$. Therefore, the total space complexity is $O(n + m + \hat{m})$.

HomophKS. The process of calculating HomophKS includes calculating the homophily score of all nodes in both G and $\hat{G}$ and performing the KS test on them. The space complexity of storing G and $\hat{G}$ is $O(n + m)$ and $O(n + \hat{m})$, respectively. The space complexity of computing the homophily score of G and $\hat{G}$ is dominated by storing node features and aggregated node features, which is $O(nk)$. The space complexity of the KS test is $O(n)$. Therefore, the total time complexity is $O(nk + m + \hat{m})$.

B.3 Details of LEO

LEO consists of three components, M_G, M_S and M_P. We use GCN model [8] for the vanilla GNN module M_G. For the structure learning-based GNN module M_S, we use SLAPS [6]. The number of layers in GCN and SLAPS is equal to two. Lastly, for the node-proximity module M_P, we use cosine similarity for homophilic graphs (Cora, Citeseer, Cora-ML, and LastFMAsia) and SVD for the heterophilic graphs (Chameleon and Squirrel).

Below, we shall provide more details for the "Training" entry in Sec 4.2. We search the learning rate within the range of $\{5e - 3, 1e - 2, 5e - 2\}$, the hidden dimension within the range of $\{32, 64, 128\}$, the ratio k in filtered positive samples within the range of $\{0, 10, 20, 30, 40, 50\}$, and the weights of the additional cross-entropy loss terms within the range of $\{0.1, 1, 10\}$. The model is trained for 50 epochs. Additionally, we fine-tune the following parameters in SLAPS: the number of pre-train epochs for structure learning within the range of $\{20, 40, 60\}$, and the number of degrees in the generated structure within the range of $\{10, 15, 20\}$.

B.4 Details of Competitors in Sec. 5.2

In this section, we provide descriptions of the competitors used in Sec. 5.2 and introduce their detailed parameter settings.

***Node proximity-based* methods.** In node proximity-based methods, the score of an edge can be considered as the proximity between its two endpoints. In SVD, a rank-100 approximation of the adjacency matrix is used, with the score of an edge determined by the corresponding element in the approximated adjacency matrix. In Cosine Sim., the score of an edge is computed based on the cosine similarity between the two endpoints.

***Logistic regression-based* methods.** Logistic regression-based methods are designed to predict the scores of edges based on graph statistics used in existing noticeability measures, namely node degrees, clustering coefficients, and homophily scores. A logistic regression model is employed to derive the score of each edge, taking the graph statistics values of the two endpoints of each edge as the input. Logistic regression models using node degree, clustering coefficient, and homophily were named Degree Model, Clustering Coeff. Model, and Homophily Model, respectively. These models are trained with ground-truth labels, where real edges are labeled as 1 and attack edges are labeled as 0. For training, edges in the attacked graph are split into training and validation sets in a 3:1 ratio. The model is fine-tuned via a grid search, where the range of learning rates is $\{5e-3, 1e-2, 5e-2\}$. Training is conducted with a batch size of 256 for 50 epochs, and the model with the best validation performance is selected.

***GCN-based* methods.** GCN-based methods employ GCNs to score the edges in the attacked graph. To infer edge scores using GCNs, we augmented GCNs with a bilinear layer followed by sigmoid activation. The score of an edge is computed via this bilinear layer with sigmoid activation with embeddings of nodes belonging to the edge, which is inferred from GCNs. This process is equal to the edge scoring layer in LEO.

To obtain node embeddings, GCN and its variants that enhance robustness against attacks, such as GCNSVD, RGCN, MedianGCN, GNNGuard, and MetaGC are employed. All GCNs are fine-tuned via a grid search, where the range of learning rate is $\{5e-3, 1e-2, 5e-2\}$, and the range of hidden unit numbers is $\{32, 64, 128\}$. Additionally, for GCNSVD, a rank-100 approximation of the adjacency matrix of the attacked graph is generated for use as input in GCN. For RGCN, hyperparameters $(\gamma, \beta_1, \beta_2)$ are set to $(1, 5e-4, 5e-4)$ as reported in the original paper. For GNNGuard and MetaGC, all hyperparameters are set based on the source code released by the author. After training MetaGC, the inferred weight values from its meta-model are used as scores for the edges. For all *GCN-based* methods, training is conducted with a batch size of 256 for 50 epochs, and the model with the best validation performance is selected.

B.5 Details of Competitors in Sec. 6

We provide descriptions of the competitors used in Sec. 6 and introduce their detailed parameter settings. We employ four methods as our competitors: SVD filtering [5], Jaccard filtering [10], GNNGuard filtering [11], and MetaGC filtering [7]. For each method, the edges with low scores are removed. For SVD filtering, the scores of edges are determined by a rank-100 approximation of the adjacency matrix. For Jaccard filtering, the scores of edges are computed based on the Jaccard index between node pairs belonging to the edges. For GNNGuard filtering, the scores of edges are computed in the same manner as LEO. For MetaGC filtering, weights inferred from the meta-model are used as scores for the edges. Note that the number of removed edges is equal to the number of attack edges.

C ADDITIONAL EXPERIMENTAL RESULTS

C.1 Ablation Study

In Table 1, we conduct experiments on LEO and its 7 variants, each consisting of a different combination of the modules in LEO. The experiment settings of the ablation study are the same as Sec. 5.2. In all the settings, LEO performs best in terms of average rank, indicating that each module of LEO positively contributes to performance improvements.

C.2 Additional Experimental Results Supplementing Sec. 5.2

Table 2, Table 3, and Table 4 present the additional results regarding Q1 with attack rates of 5%, 10%, and 15%, respectively. Across all the experimental results, LEO consistently outperforms the competitors in terms of average ranks. These findings align with the results in Sec. 5.2 in the main paper and further support the justification for LEO.

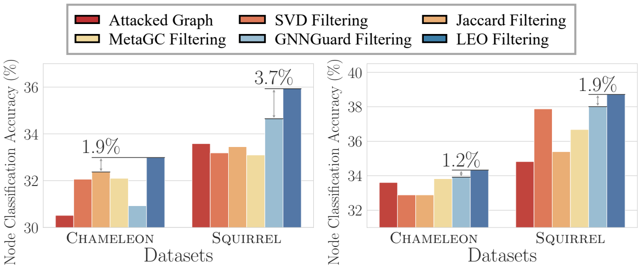
C.3 Additional Experimental Results Supplementing Sec. 5.3

Additional experimental results regarding Q2 & Q3 on Rand. Attack, DICE, Structack, and Metattack with an attack rate of 10% are presented in Table 5, Table 6, Table 7, and Table 8, respectively. In Table 5, 6, and 7, results on the Squirrel and Chameleon datasets are excluded, because Rand. Attack, DICE, and Structack do not support attacks to heterophilic graphs. In Table 8, the results on LastFMAsia are excluded because the computation on this dataset runs out of memory. Note that Metattack targets at a poisoning attack scenario, i.e., the target model is supposed to be retrained on attacked graphs. Thus, we evaluate the target model by retraining a GCN model for each of the attacked graphs.

Across all the experimental results, HideNSeek consistently achieves lower bypassable rates compared to the existing measures. In addition, HideNSeek effectively detects attack edges even when the attack rate is low.

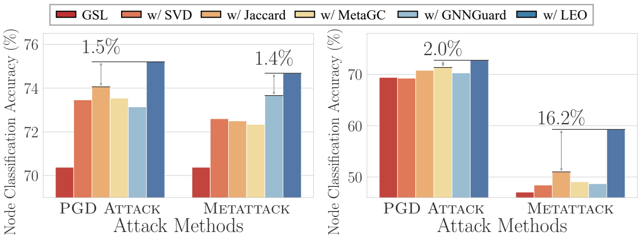
C.4 Additional Experimental Results Supplementing Sec. 6

Heterophilic graphs. Below, we shall show that LEO can improve the performance of GNNs on heterophilic graphs. In heterophilic graphs, we tend to have edges that connect nodes from different classes. Therefore, using a vanilla GNN that aggregates neighbors' embeddings through message passing is not suitable for heterophilic



(a) PGD Attack **(b) Metattack**

Figure 4: LEO consistently improves the node classification accuracy on heterophilic graphs.



(a) SLAPS **(b) SE-GSL**

Figure 5: LEO consistently improves the node classification accuracy of graph structure learning (GSL) methods, SLAPS, and SE-GSL.

graphs. Several GNNs specialized for heterophilic graphs have been proposed. According to Platonov et al. [9], FAGCN [2] achieves the best performance on the Chameleon and Squirrel datasets. Hence, we use FAGCN in our experiments.

We perform the same experiments described in Sec. 6 on FAGCN using attacked graphs with 10% PGD Attack and Metattack applied to Cora.[1] As shown in Fig. 4, LEO consistently improves the performance of FAGCN on heterophilic graphs. Furthermore, LEO achieves the best performance compared to its competitors. Specifically, LEO improves the node classification accuracy up to 3.7% compared to the second-best method (see Fig. 4a). This indicates that LEO can effectively distinguish suspicious edges even in heterophilic graphs.

Graph structure learning. LEO also improves the performance of graph structure learning (GSL) methods. GSL methods optimize the graph structure while optimizing specific downstream tasks (e.g., node classification). For the experiments on GSL, we use SLAPS [6] and SE-GSL [12], which are representative GSL methods. SLAPS is a method that infers graph structures from node features in a graph-agnostic manner and trains graph structures simultaneously with the specific downstream task. SE-GSL is a method that enhances graph structures using one-dimensional structural entropy maximization and constructs an encoding tree to capture hierarchical information.

In this experiment, we incorporate LEO into both GSL methods, SLAPS and SE-GSL. For SLAPS, the adjacency matrix to be learned is initially based on the cosine similarity of node features before training. To integrate LEO and its competitors with SLAPS, we replaced the cosine similarity matrix with a filtered adjacency matrix provided by LEO. For SE-GSL, the graph is enhanced using structural entropy and encoding tree theory. To apply LEO with SE-GSL, we use the filtered adjacency matrix instead of the enhanced graph. We use PGD Attack and Metattack with a 10% attack rate on the Cora dataset. For competitors (SVD, Jaccard, and MetaGC), we also adopt the same procedure.

As shown in Fig. 5, both GSL methods achieve the best performance when equipped with LEO, compared to when equipped with the competitors (SVD, Jaccard, and MetaGC). Specifically, LEO improves the node classification accuracy of GSL methods up to 16.2% compared to the second-best method (see Fig. 5b).

Notes on Tables 1 to 8

Note that Tables 1 to 8 are available in the following pages.

[1] Attack methods, except for gradient-based methods, are designed under the assumption of graph homophily, making them less effective on heterophilic graphs.

REFERENCES

[1] Noga Alon, Raphael Yuster, and Uri Zwick. 1997. Finding and counting given length cycles. *Algorithmica* 17, 3 (1997), 209–223.

[2] Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. 2021. Beyond low-frequency information in graph convolutional networks. In *AAAI Conference on Artificial Intelligence*.

[3] Coen Boot et al. 2016. *Algorithms for determining the clustering coefficient in large graphs*. B.S. thesis.

[4] Yongqiang Chen, Han Yang, Yonggang Zhang, MA KAILI, Tongliang Liu, Bo Han, and James Cheng. 2021. Understanding and Improving Graph Injection Attack by Promoting Unnoticeability. In *International Conference on Learning Representations*.

[5] Negin Entezari, Saba A Al-Sayouri, Amirali Darvishzadeh, and Evangelos E Papalexakis. 2020. All you need is low (rank) defending against adversarial attacks on graphs. In *International Conference on Web Search and Data Mining*.

[6] Bahare Fatemi, Layla El Asri, and Seyed Mehran Kazemi. 2021. SLAPS: Self-supervision improves structure learning for graph neural networks. In *Conference on Neural Information Processing Systems*.

[7] Hyeonsoo Jo, Fanchen Bu, and Kijung Shin. 2023. Robust Graph Clustering via Meta Weighting for Noisy Graphs. In *International Conference on Information and Knowledge Management*.

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

[9] Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. 2022. A critical look at the evaluation of GNNs under heterophily: Are we really making progress?. In *International Conference on Learning Representations*.

[10] Huijun Wu, Chen Wang, Yuriy Tyshetskiy, Andrew Docherty, Kai Lu, and Liming Zhu. 2019. Adversarial examples for graph data: deep insights into attack and defense. In *International Joint Conference on Artificial Intelligence*.

[11] Xiang Zhang and Marinka Zitnik. 2020. Gnnguard: Defending graph neural networks against adversarial attacks. *NeurIPS*.

[12] Dongcheng Zou, Hao Peng, Xiang Huang, Renyu Yang, Jianxin Li, Jia Wu, Chunyang Liu, and Philip S Yu. 2023. Se-gsl: A general and effective graph structure learning framework through structural entropy optimization. In *Proceedings of the ACM on Web Conference*.

[13] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. 2018. Adversarial attacks on neural networks for graph data. In *International Conference on Knowledge Discovery and Data Mining*.

Table 1: Ablation study: LEO ranks first in all attack methods w.r.t average rank (AR) compared with its variants. O.O.T: out of time (> 3 hours). O.O.M: out of (GPU) memory. N.A: not available (specifically, the maximum number of attacks allowed by Structack is exceeded). For each setting (each column), the best and second-best results are in red and blue, respectively.

	Modules			Datasets						AR
	M_G	M_S	M_P	Cora	Citeseer	Cora-ML	LastFMAsia	Chameleon	Squirrel	
Rand. Attack	✓	✗	✗	0.790±0.006	0.720±0.123	0.826±0.019	0.911±0.003	0.939±0.009	0.941±0.002	5.3
	✗	✓	✗	0.846±0.011	0.860±0.004	0.827±0.020	0.894±0.003	0.840±0.018	0.819±0.012	5.2
	✗	✗	✓	0.802±0.007	0.893±0.006	0.795±0.005	0.835±0.002	0.937±0.002	0.953±0.001	5.5
	✓	✓	✗	0.837±0.009	0.852±0.003	0.814±0.027	0.926±0.001	0.894±0.067	0.799±0.143	5.5
	✓	✗	✓	0.849±0.006	0.910±0.005	0.895±0.004	0.928±0.002	0.974±0.001	0.976±0.001	2.0
	✗	✓	✓	0.857±0.007	0.930±0.003	0.877±0.008	0.883±0.003	0.974±0.002	0.976±0.001	2.3
	✓	✓	✓	0.894±0.011	0.926±0.006	0.914±0.004	0.931±0.000	0.973±0.003	0.976±0.001	**1.5**
DICE	✓	✗	✗	0.779±0.021	0.675±0.163	0.835±0.019	0.917±0.005	0.944±0.009	0.938±0.002	5.2
	✗	✓	✗	0.856±0.017	0.881±0.010	0.836±0.023	0.912±0.005	0.855±0.012	0.818±0.021	4.8
	✗	✗	✓	0.809±0.011	0.896±0.009	0.805±0.008	0.842±0.005	0.910±0.005	0.940±0.001	5.3
	✓	✓	✗	0.854±0.008	0.872±0.012	0.826±0.021	0.936±0.003	0.786±0.150	0.784±0.137	5.7
	✓	✗	✓	0.855±0.009	0.914±0.012	0.904±0.005	0.937±0.003	0.955±0.009	0.941±0.010	2.5
	✗	✓	✓	0.885±0.017	0.940±0.010	0.899±0.006	0.897±0.003	0.949±0.004	0.944±0.004	2.8
	✓	✓	✓	0.899±0.011	0.942±0.008	0.932±0.003	0.941±0.003	0.960±0.003	0.938±0.003	**1.5**
PGD Attack	✓	✗	✗	0.644±0.012	0.754±0.016	0.762±0.020	0.560±0.033	0.899±0.017	0.900±0.003	5.8
	✗	✓	✗	0.749±0.026	0.855±0.008	0.835±0.033	0.736±0.011	0.868±0.007	0.839±0.023	5.5
	✗	✗	✓	0.809±0.005	0.882±0.006	0.812±0.008	0.842±0.009	0.913±0.010	0.943±0.006	2.8
	✓	✓	✗	0.805±0.021	0.852±0.011	0.857±0.020	0.724±0.007	0.863±0.043	0.714±0.100	5.7
	✓	✗	✓	0.848±0.004	0.904±0.006	0.915±0.004	0.759±0.005	0.882±0.008	0.924±0.019	3.0
	✗	✓	✓	0.779±0.045	0.926±0.004	0.886±0.014	0.775±0.021	0.877±0.009	0.925±0.019	3.5
	✓	✓	✓	0.874±0.004	0.931±0.008	0.930±0.011	0.853±0.008	0.880±0.012	0.926±0.019	**1.7**
Structack	✓	✗	✗	0.819±0.023	0.680±0.165	0.894±0.014	0.868±0.006	0.942±0.003	N.A	6.0
	✗	✓	✗	0.876±0.010	0.883±0.006	0.886±0.028	0.934±0.001	0.856±0.017	N.A	5.0
	✗	✗	✓	0.842±0.000	0.886±0.000	0.823±0.000	0.854±0.000	0.948±0.000	N.A	5.6
	✓	✓	✗	0.894±0.014	0.878±0.005	0.952±0.009	0.907±0.001	0.878±0.053	N.A	4.4
	✓	✗	✓	0.906±0.002	0.916±0.003	0.940±0.005	0.904±0.001	0.966±0.002	N.A	3.2
	✗	✓	✓	0.898±0.025	0.936±0.008	0.918±0.013	0.924±0.003	0.969±0.002	N.A	2.6
	✓	✓	✓	0.941±0.004	0.950±0.006	0.965±0.002	0.939±0.002	0.968±0.003	N.A	**1.2**
Metattack	✓	✗	✗	0.696±0.025	0.617±0.067	0.613±0.016	O.O.M	0.939±0.005	0.901±0.009	6.0
	✗	✓	✗	0.823±0.036	0.798±0.016	0.772±0.026	O.O.M	0.856±0.012	0.881±0.022	5.6
	✗	✗	✓	0.840±0.004	0.886±0.005	0.791±0.011	O.O.M	0.940±0.005	0.972±0.002	3.2
	✓	✓	✗	0.811±0.022	0.778±0.025	0.588±0.007	O.O.M	0.899±0.112	0.699±0.087	6.4
	✓	✗	✓	0.850±0.005	0.867±0.011	0.818±0.015	O.O.M	0.961±0.004	0.969±0.004	3.4
	✗	✓	✓	0.876±0.013	0.906±0.008	0.847±0.030	O.O.M	0.963±0.004	0.970±0.005	1.6
	✓	✓	✓	0.894±0.003	0.893±0.010	0.856±0.016	O.O.M	0.963±0.004	0.970±0.003	**1.4**

Table 2: The additional experimental results regarding Q1 with an attack rate of 5%. LEO ranks first in all attack methods w.r.t average rank (AR). O.O.T: out of time (> 3 hours). O.O.M: out of (GPU) memory. N.A: not applicable (specifically, the maximum number of attacks allowed by Structack is exceeded). For each setting (each column), the best and second-best results are in red and blue, respectively.

Rand. Attack	SVD	0.688±0.015	0.626±0.017	0.795±0.007	0.835±0.004	0.935±0.003	0.953±0.002	5.8
	Cosine Sim.	0.803±0.014	0.893±0.010	0.795±0.008	0.831±0.004	0.560±0.010	0.486±0.006	6.5
	Degree Model	0.664±0.012	0.616±0.008	0.762±0.009	0.810±0.006	0.845±0.005	0.893±0.002	8.7
	Clustering Coeff. Model	0.592±0.021	0.614±0.006	0.607±0.019	0.677±0.011	0.766±0.005	0.841±0.002	10.5
	Homophily Model	0.642±0.007	0.630±0.008	0.674±0.010	0.722±0.009	0.697±0.015	0.640±0.016	9.7
	GCN	0.779±0.014	0.769±0.023	0.837±0.028	0.917±0.008	0.951±0.003	0.942±0.003	4.3
	GCNSVD	0.685±0.018	0.627±0.117	0.771±0.017	0.925±0.005	0.930±0.002	0.924±0.003	6.2
	RGCN	0.838±0.011	0.817±0.021	0.801±0.156	0.920±0.006	0.525±0.045	0.585±0.043	5.8
	MedianGCN	0.790±0.029	0.776±0.029	0.700±0.119	O.O.T	0.653±0.174	O.O.T	8.8
	GNNGuard	0.788±0.071	0.779±0.029	0.812±0.007	0.924±0.006	0.919±0.038	0.905±0.003	4.5
	MetaGC	0.728±0.018	0.704±0.018	0.719±0.035	0.831±0.004	0.973±0.002	0.950±0.009	5.8
	LEO (Proposed)	0.890±0.020	0.932±0.011	0.916±0.003	0.933±0.004	0.976±0.002	0.978±0.001	**1.0**
DICE	SVD	0.677±0.006	0.616±0.006	0.785±0.010	0.829±0.008	0.922±0.008	0.945±0.002	6.7
	Cosine Sim.	0.827±0.008	0.904±0.009	0.803±0.011	0.843±0.006	0.565±0.009	0.484±0.007	6.3
	Degree Model	0.664±0.014	0.618±0.009	0.757±0.016	0.806±0.009	0.838±0.007	0.885±0.004	8.3
	Clustering Coeff. Model	0.595±0.018	0.610±0.020	0.608±0.025	0.676±0.007	0.761±0.015	0.834±0.007	10.7
	Homophily Model	0.636±0.025	0.658±0.010	0.677±0.021	0.724±0.016	0.699±0.015	0.633±0.012	10.0
	GCN	0.802±0.013	0.808±0.019	0.842±0.016	0.925±0.005	0.952±0.004	0.942±0.005	3.5
	GCNSVD	0.692±0.014	0.758±0.030	0.781±0.022	0.938±0.002	0.950±0.005	0.921±0.003	5.2
	RGCN	0.847±0.013	0.792±0.156	0.749±0.185	0.925±0.001	0.535±0.050	0.571±0.142	6.7
	MedianGCN	0.830±0.027	0.773±0.036	0.748±0.093	O.O.T	0.770±0.172	O.O.T	8.2
	GNNGuard	0.804±0.076	0.769±0.035	0.813±0.009	0.935±0.008	0.918±0.036	0.918±0.018	4.8
	MetaGC	0.737±0.022	0.714±0.022	0.694±0.053	0.825±0.007	0.948±0.003	0.949±0.008	6.5
	LEO (Proposed)	0.914±0.017	0.941±0.006	0.925±0.018	0.944±0.003	0.971±0.005	0.962±0.007	**1.0**
PGD Attack	SVD	0.846±0.005	0.737±0.007	0.909±0.002	0.384±0.022	0.952±0.002	0.839±0.275	4.7
	Cosine Sim.	0.815±0.016	0.881±0.006	0.819±0.010	0.844±0.009	0.586±0.008	0.520±0.022	6.2
	Degree Model	0.774±0.019	0.726±0.014	0.894±0.007	0.269±0.023	0.966±0.005	0.794±0.384	5.7
	Clustering Coeff. Model	0.693±0.035	0.727±0.014	0.724±0.041	0.737±0.008	0.895±0.012	0.959±0.001	6.7
	Homophily Model	0.735±0.020	0.710±0.023	0.838±0.018	0.897±0.015	0.829±0.013	0.757±0.075	7.0
	GCN	0.645±0.033	0.718±0.123	0.790±0.025	0.611±0.038	0.935±0.005	0.853±0.161	7.5
	GCNSVD	0.770±0.021	0.783±0.020	0.878±0.010	0.615±0.035	0.949±0.008	0.792±0.356	4.7
	RGCN	0.585±0.024	0.743±0.119	0.626±0.135	0.532±0.039	0.538±0.030	0.582±0.061	9.8
	MedianGCN	0.657±0.113	0.763±0.045	0.724±0.086	O.O.T	0.822±0.184	O.O.T	9.7
	GNNGuard	0.764±0.014	0.741±0.033	0.832±0.016	0.512±0.043	0.942±0.002	0.733±0.412	6.7
	MetaGC	0.738±0.033	0.680±0.035	0.783±0.021	0.657±0.037	0.907±0.013	0.908±0.056	7.2
	LEO (Proposed)	0.877±0.018	0.935±0.005	0.935±0.012	0.865±0.011	0.911±0.018	0.922±0.054	**2.2**
Structack	SVD	0.842±0.000	0.747±0.000	0.917±0.000	0.934±0.000	0.952±0.000	N.A	5.2
	Cosine Sim.	0.831±0.000	0.892±0.000	0.836±0.000	0.833±0.000	0.566±0.000	N.A	7.8
	Degree Model	0.913±0.004	0.819±0.008	0.965±0.001	0.975±0.000	0.856±0.001	N.A	3.0
	Clustering Coeff. Model	0.640±0.012	0.688±0.017	0.750±0.008	0.756±0.004	0.711±0.004	N.A	10.4
	Homophily Model	0.779±0.008	0.735±0.009	0.864±0.004	0.868±0.002	0.705±0.005	N.A	8.2
	GCN	0.843±0.014	0.802±0.013	0.910±0.009	0.835±0.005	0.953±0.004	N.A	5.6
	GCNSVD	0.852±0.013	0.780±0.018	0.918±0.004	0.955±0.002	0.951±0.004	N.A	3.8
	RGCN	0.845±0.017	0.580±0.177	0.760±0.181	0.877±0.008	0.549±0.052	N.A	9.0
	MedianGCN	0.682±0.133	0.670±0.130	0.724±0.084	O.O.T	0.684±0.188	N.A	11.2
	GNNGuard	0.889±0.010	0.778±0.025	0.911±0.009	0.896±0.003	0.927±0.035	N.A	5.0
	MetaGC	0.752±0.008	0.727±0.030	0.771±0.030	0.875±0.007	0.981±0.001	N.A	7.4
	LEO (Proposed)	0.950±0.006	0.950±0.003	0.975±0.002	0.937±0.001	0.984±0.001	N.A	**1.4**
Metattack	SVD	0.805±0.004	0.637±0.027	0.768±0.041	0.925±0.004	0.966±0.003	0.989±0.000	4.5
	Cosine Sim.	0.835±0.006	0.875±0.006	0.789±0.006	0.870±0.005	0.551±0.007	0.453±0.006	5.8
	Degree Model	0.681±0.009	0.561±0.015	0.701±0.017	0.828±0.010	0.970±0.002	0.983±0.001	7.2
	Clustering Coeff. Model	0.693±0.072	0.721±0.014	0.632±0.266	0.901±0.007	0.948±0.005	0.973±0.001	6.7
	Homophily Model	0.703±0.015	0.661±0.019	0.777±0.012	0.894±0.006	0.771±0.010	0.666±0.017	7.2
	GCN	0.755±0.015	0.714±0.019	0.677±0.049	0.835±0.012	0.947±0.005	0.919±0.004	7.7
	GCNSVD	0.801±0.005	0.701±0.113	0.776±0.032	0.939±0.002	0.970±0.003	0.937±0.022	4.0
	RGCN	0.756±0.160	0.607±0.094	0.682±0.094	0.823±0.016	0.519±0.022	0.520±0.029	9.7
	MedianGCN	0.840±0.031	0.750±0.024	0.751±0.046	O.O.T	0.733±0.234	O.O.T	7.3
	GNNGuard	0.824±0.019	0.701±0.014	0.635±0.013	0.844±0.014	0.954±0.004	0.932±0.009	6.8
	MetaGC	0.683±0.028	0.678±0.030	0.668±0.046	nan±nan	0.963±0.006	0.938±0.013	8.5
	LEO (Proposed)	0.922±0.004	0.900±0.007	0.873±0.017	0.894±0.008	0.967±0.004	0.980±0.001	**2.2**

Table 3: The additional experimental results regarding Q1 with an attack rate of 15%. LEO ranks first in all attack methods w.r.t average rank (AR). O.O.T: out of time (> 3 hours). O.O.M: out of (GPU) memory. N.A: not applicable (specifically, the maximum number of attacks allowed by Structack is exceeded). For each setting (each column), the best and second-best results are in red and blue, respectively.

	Method	Datasets						AR
		Cora	Citeseer	Cora-ML	LastFMAsia	Chameleon	Squirrel	
Rand. Attack	SVD	0.697±0.006	0.638±0.006	0.794±0.005	0.835±0.001	0.937±0.003	0.953±0.001	5.0
	Cosine Sim.	0.804±0.005	0.892±0.005	0.794±0.004	0.834±0.000	0.561±0.004	0.487±0.001	6.2
	Degree Model	0.670±0.006	0.624±0.006	0.756±0.006	0.809±0.002	0.847±0.002	0.892±0.001	8.0
	Clustering Coeff. Model	0.594±0.005	0.609±0.006	0.617±0.013	0.688±0.004	0.775±0.003	0.857±0.001	10.3
	Homophily Model	0.641±0.010	0.636±0.006	0.675±0.008	0.732±0.003	0.697±0.012	0.624±0.003	9.3
	GCN	0.776±0.017	0.772±0.025	0.817±0.004	0.907±0.008	0.930±0.003	0.935±0.010	3.7
	GCNSVD	0.657±0.009	0.612±0.103	0.750±0.005	0.914±0.003	0.890±0.022	0.923±0.002	6.8
	RGCN	0.751±0.145	0.631±0.182	0.661±0.119	0.910±0.005	0.573±0.010	0.555±0.025	8.0
	MedianGCN	0.718±0.114	0.727±0.106	0.615±0.076	O.O.T	0.559±0.069	O.O.T	10.0
	GNNGuard	0.795±0.008	0.735±0.028	0.810±0.007	0.919±0.002	0.878±0.032	0.914±0.011	4.0
	MetaGC	0.730±0.007	0.702±0.016	0.712±0.034	0.826±0.009	0.969±0.002	0.948±0.008	5.5
	LEO (Proposed)	0.891±0.006	0.933±0.005	0.912±0.006	0.927±0.002	0.971±0.003	0.974±0.001	**1.0**
DICE	SVD	0.653±0.007	0.606±0.013	0.770±0.004	0.816±0.005	0.895±0.005	0.932±0.001	6.5
	Cosine Sim.	0.813±0.008	0.894±0.003	0.806±0.003	0.841±0.004	0.565±0.005	0.482±0.005	6.2
	Degree Model	0.652±0.009	0.606±0.012	0.761±0.007	0.806±0.006	0.844±0.008	0.885±0.002	8.2
	Clustering Coeff. Model	0.577±0.009	0.596±0.015	0.616±0.012	0.681±0.009	0.774±0.006	0.847±0.003	10.5
	Homophily Model	0.627±0.007	0.639±0.015	0.686±0.012	0.729±0.006	0.695±0.006	0.622±0.009	9.7
	GCN	0.773±0.015	0.755±0.022	0.836±0.018	0.907±0.005	0.937±0.013	0.936±0.001	3.3
	GCNSVD	0.654±0.014	0.694±0.028	0.754±0.008	0.928±0.003	0.923±0.004	0.921±0.001	5.2
	RGCN	0.815±0.022	0.830±0.014	0.696±0.176	0.910±0.004	0.549±0.012	0.557±0.018	6.7
	MedianGCN	0.687±0.139	0.663±0.119	0.707±0.098	O.O.T	0.625±0.184	O.O.T	9.5
	GNNGuard	0.781±0.003	0.717±0.071	0.812±0.012	0.925±0.001	0.876±0.043	0.916±0.014	4.3
	MetaGC	0.717±0.041	0.717±0.026	0.656±0.054	0.810±0.008	0.896±0.002	0.946±0.009	5.8
	LEO (Proposed)	0.890±0.009	0.936±0.010	0.930±0.006	0.933±0.002	0.945±0.009	0.916±0.005	**1.7**
PGD Attack	SVD	0.765±0.008	0.710±0.007	0.853±0.007	0.316±0.031	0.887±0.003	0.928±0.016	4.2
	Cosine Sim.	0.800±0.006	0.881±0.004	0.801±0.005	0.838±0.007	0.587±0.003	0.521±0.003	5.7
	Degree Model	0.710±0.008	0.667±0.003	0.840±0.009	0.267±0.019	0.934±0.008	0.940±0.005	5.7
	Clustering Coeff. Model	0.636±0.057	0.677±0.011	0.690±0.035	0.618±0.067	0.865±0.011	0.954±0.005	6.5
	Homophily Model	0.728±0.009	0.694±0.015	0.784±0.005	0.898±0.005	0.823±0.013	0.739±0.013	6.2
	GCN	0.613±0.029	0.741±0.007	0.752±0.017	0.517±0.014	0.862±0.011	0.881±0.014	6.8
	GCNSVD	0.734±0.017	0.699±0.112	0.829±0.018	0.484±0.015	0.883±0.008	0.924±0.008	4.7
	RGCN	0.591±0.011	0.501±0.008	0.660±0.092	0.409±0.019	0.523±0.049	0.604±0.056	10.8
	MedianGCN	0.545±0.029	0.670±0.120	0.733±0.019	O.O.T	0.785±0.167	O.O.T	11.0
	GNNGuard	0.703±0.032	0.734±0.025	0.776±0.013	0.400±0.022	0.879±0.006	0.905±0.015	6.5
	MetaGC	0.704±0.015	0.682±0.024	0.743±0.028	0.585±0.022	0.845±0.018	0.923±0.009	7.0
	LEO (Proposed)	0.863±0.008	0.910±0.009	0.914±0.006	0.841±0.008	0.849±0.030	0.915±0.028	**3.0**
Structack	SVD	0.786±0.000	0.773±0.000	0.851±0.000	0.844±0.000	N.A	N.A	6.5
	Cosine Sim.	0.829±0.000	0.894±0.000	0.825±0.000	0.872±0.000	N.A	N.A	4.2
	Degree Model	0.849±0.003	0.849±0.002	0.885±0.001	0.850±0.000	N.A	N.A	3.8
	Clustering Coeff. Model	0.553±0.009	0.693±0.006	0.572±0.006	0.656±0.001	N.A	N.A	11.0
	Homophily Model	0.687±0.004	0.752±0.006	0.715±0.002	0.728±0.002	N.A	N.A	9.2
	GCN	0.787±0.023	0.798±0.011	0.885±0.014	0.862±0.009	N.A	N.A	4.8
	GCNSVD	0.789±0.017	0.795±0.021	0.843±0.006	0.884±0.003	N.A	N.A	4.8
	RGCN	0.754±0.110	0.565±0.144	0.587±0.116	0.866±0.005	N.A	N.A	8.8
	MedianGCN	0.546±0.044	0.736±0.075	0.568±0.053	O.O.T	N.A	N.A	11.2
	GNNGuard	0.831±0.020	0.760±0.020	0.892±0.002	0.891±0.003	N.A	N.A	3.5
	MetaGC	0.743±0.009	0.726±0.028	0.740±0.042	0.835±0.008	N.A	N.A	9.0
	LEO (Proposed)	0.933±0.002	0.955±0.003	0.959±0.006	0.934±0.003	N.A	N.A	**1.0**
Metattack	SVD	0.643±0.012	0.554±0.012	0.710±0.013	O.O.M	0.921±0.002	0.941±0.007	7.4
	Cosine Sim.	0.839±0.002	0.891±0.004	0.790±0.009	O.O.M	0.557±0.004	0.448±0.004	5.4
	Degree Model	0.488±0.009	0.373±0.004	0.610±0.005	O.O.M	0.932±0.002	0.952±0.003	8.2
	Clustering Coeff. Model	0.607±0.120	0.697±0.009	0.731±0.040	O.O.M	0.916±0.002	0.967±0.001	5.8
	Homophily Model	0.702±0.012	0.656±0.009	0.755±0.009	O.O.M	0.755±0.015	0.589±0.007	6.2
	GCN	0.657±0.045	0.601±0.051	0.584±0.018	O.O.M	0.930±0.004	0.883±0.011	8.2
	GCNSVD	0.614±0.040	0.652±0.088	0.671±0.010	O.O.M	0.941±0.004	0.933±0.007	6.4
	RGCN	0.695±0.104	0.651±0.089	0.562±0.077	O.O.M	0.614±0.061	0.531±0.026	9.0
	MedianGCN	0.650±0.147	0.736±0.044	0.734±0.060	O.O.M	0.556±0.125	O.O.T	7.8
	GNNGuard	0.724±0.048	0.615±0.057	0.528±0.029	O.O.M	0.933±0.006	0.884±0.005	7.0
	MetaGC	0.700±0.021	0.684±0.024	0.665±0.052	O.O.M	0.948±0.004	0.925±0.012	5.2
	LEO (Proposed)	0.875±0.007	0.882±0.009	0.834±0.020	O.O.M	0.956±0.003	0.957±0.003	**1.4**

Table 4: The additional experimental results regarding Q1 with an attack rate of 20%. LEO ranks first in all attack methods w.r.t average rank (AR). O.O.T: out of time (> 3 hours). O.O.M: out of (GPU) memory. N.A: not applicable (specifically, the maximum number of attacks allowed by STRUCTACK is exceeded). For each setting (each column), the best and second-best results are in red and blue, respectively.

	Method	Datasets						AR
		CORA	CITESEER	CORA-ML	LASTFMASIA	CHAMELEON	SQUIRREL	
RAND. ATTACK	SVD	0.698±0.003	0.641±0.007	0.796±0.005	0.835±0.001	0.939±0.002	0.953±0.001	4.7
	COSINE SIM.	0.805±0.005	0.891±0.003	0.796±0.005	0.834±0.001	0.562±0.004	0.487±0.001	5.8
	DEGREE MODEL	0.668±0.006	0.624±0.009	0.758±0.006	0.810±0.001	0.847±0.002	0.892±0.001	8.0
	CLUSTERING COEFF. MODEL	0.587±0.009	0.617±0.009	0.613±0.011	0.693±0.003	0.777±0.004	0.861±0.002	10.0
	HOMOPHILY MODEL	0.640±0.012	0.638±0.015	0.674±0.009	0.732±0.003	0.698±0.010	0.613±0.003	9.5
	GCN	0.766±0.014	0.698±0.113	0.765±0.025	0.907±0.002	0.928±0.001	0.934±0.008	4.7
	GCNSVD	0.663±0.005	0.696±0.010	0.753±0.006	0.911±0.002	0.872±0.003	0.923±0.002	6.3
	RGCN	0.796±0.053	0.755±0.146	0.579±0.119	0.909±0.002	0.544±0.017	0.552±0.020	7.3
	MEDIANGCN	0.675±0.126	0.602±0.118	0.609±0.080	O.O.T	0.602±0.140	O.O.T	10.8
	GNNGUARD	0.764±0.047	0.703±0.019	0.774±0.036	0.916±0.003	0.878±0.033	0.910±0.015	4.3
	METAGC	0.730±0.006	0.703±0.015	0.721±0.034	0.823±0.008	0.967±0.001	0.948±0.008	5.2
	LEO (Proposed)	0.887±0.006	0.931±0.003	0.906±0.007	0.926±0.003	0.972±0.001	0.971±0.001	**1.0**
DICE	SVD	0.642±0.003	0.603±0.008	0.749±0.006	0.812±0.003	0.874±0.005	0.923±0.001	7.0
	COSINE SIM.	0.813±0.007	0.895±0.003	0.803±0.004	0.845±0.004	0.565±0.003	0.483±0.001	6.0
	DEGREE MODEL	0.649±0.007	0.612±0.005	0.754±0.006	0.809±0.003	0.842±0.005	0.885±0.001	7.5
	CLUSTERING COEFF. MODEL	0.584±0.020	0.612±0.012	0.629±0.012	0.683±0.005	0.777±0.004	0.851±0.002	10.0
	HOMOPHILY MODEL	0.629±0.010	0.646±0.013	0.683±0.014	0.734±0.007	0.698±0.006	0.616±0.005	9.3
	GCN	0.751±0.012	0.709±0.118	0.816±0.007	0.897±0.004	0.930±0.008	0.935±0.001	3.2
	GCNSVD	0.654±0.014	0.647±0.082	0.745±0.011	0.924±0.003	0.908±0.026	0.920±0.002	5.2
	RGCN	0.811±0.014	0.546±0.113	0.797±0.096	0.905±0.008	0.549±0.027	0.572±0.019	7.7
	MEDIANGCN	0.621±0.133	0.684±0.094	0.692±0.098	O.O.T	0.637±0.183	O.O.T	10.0
	GNNGUARD	0.733±0.065	0.747±0.024	0.807±0.004	0.919±0.002	0.875±0.039	0.915±0.015	3.8
	METAGC	0.733±0.019	0.714±0.028	0.653±0.085	0.803±0.007	0.869±0.005	0.945±0.007	6.0
	LEO (Proposed)	0.897±0.014	0.937±0.005	0.922±0.010	0.928±0.001	0.930±0.010	0.907±0.017	**1.8**
PGD ATTACK	SVD	0.733±0.008	0.695±0.008	0.827±0.006	0.322±0.024	0.857±0.008	0.915±0.021	4.8
	COSINE SIM.	0.794±0.004	0.884±0.004	0.800±0.004	0.838±0.004	0.584±0.004	0.520±0.002	5.5
	DEGREE MODEL	0.689±0.010	0.653±0.007	0.816±0.003	0.282±0.021	0.925±0.009	0.930±0.005	6.0
	CLUSTERING COEFF. MODEL	0.583±0.131	0.664±0.009	0.701±0.018	0.358±0.008	0.852±0.010	0.949±0.005	7.8
	HOMOPHILY MODEL	0.710±0.006	0.695±0.008	0.767±0.006	0.897±0.004	0.803±0.016	0.722±0.011	5.8
	GCN	0.608±0.018	0.669±0.102	0.730±0.018	0.514±0.010	0.855±0.019	0.865±0.006	7.3
	GCNSVD	0.700±0.010	0.734±0.011	0.817±0.009	0.481±0.020	0.864±0.012	0.909±0.023	4.2
	RGCN	0.590±0.025	0.600±0.146	0.597±0.080	0.421±0.020	0.506±0.022	0.579±0.047	10.7
	MEDIANGCN	0.578±0.082	0.704±0.055	0.679±0.033	O.O.T	0.706±0.168	O.O.T	10.3
	GNNGUARD	0.689±0.024	0.715±0.017	0.746±0.017	0.431±0.020	0.863±0.008	0.881±0.015	5.8
	METAGC	0.702±0.031	0.679±0.025	0.726±0.042	0.577±0.022	0.824±0.017	0.920±0.011	6.2
	LEO (Proposed)	0.853±0.004	0.915±0.007	0.906±0.008	0.837±0.003	0.846±0.022	0.907±0.035	**3.2**
STRUCTACK	SVD	0.774±0.000	0.785±0.000	0.816±0.000	0.860±0.000	N.A	N.A	5.5
	COSINE SIM.	0.826±0.000	0.892±0.000	0.818±0.000	0.863±0.000	N.A	N.A	3.8
	DEGREE MODEL	0.833±0.002	0.850±0.004	0.824±0.001	0.874±0.000	N.A	N.A	3.0
	CLUSTERING COEFF. MODEL	0.555±0.004	0.683±0.007	0.575±0.005	0.680±0.003	N.A	N.A	10.5
	HOMOPHILY MODEL	0.659±0.003	0.715±0.007	0.686±0.004	0.768±0.001	N.A	N.A	9.0
	GCN	0.783±0.029	0.648±0.188	0.831±0.002	0.844±0.006	N.A	N.A	6.5
	GCNSVD	0.767±0.013	0.722±0.126	0.801±0.003	0.880±0.004	N.A	N.A	5.8
	RGCN	0.595±0.121	0.498±0.016	0.527±0.034	0.860±0.006	N.A	N.A	10.2
	MEDIANGCN	0.609±0.102	0.650±0.141	0.575±0.040	O.O.T	N.A	N.A	10.5
	GNNGUARD	0.796±0.016	0.747±0.046	0.830±0.004	0.874±0.003	N.A	N.A	3.8
	METAGC	0.739±0.007	0.727±0.023	0.732±0.041	0.837±0.008	N.A	N.A	7.8
	LEO (Proposed)	0.924±0.005	0.950±0.007	0.945±0.012	0.928±0.002	N.A	N.A	**1.0**
METATTACK	SVD	0.598±0.009	0.523±0.011	0.695±0.013	O.O.M	0.895±0.008	0.900±0.011	7.4
	COSINE SIM.	0.841±0.003	0.895±0.005	0.789±0.009	O.O.M	0.554±0.004	0.441±0.005	5.6
	DEGREE MODEL	0.438±0.010	0.313±0.006	0.575±0.009	O.O.M	0.908±0.005	0.932±0.002	8.6
	CLUSTERING COEFF. MODEL	0.698±0.007	0.697±0.015	0.586±0.224	O.O.M	0.910±0.005	0.964±0.002	4.2
	HOMOPHILY MODEL	0.695±0.006	0.663±0.009	0.749±0.012	O.O.M	0.743±0.015	0.568±0.008	6.4
	GCN	0.624±0.029	0.529±0.047	0.570±0.029	O.O.M	0.911±0.006	0.854±0.007	7.8
	GCNSVD	0.550±0.027	0.636±0.028	0.624±0.010	O.O.M	0.919±0.008	0.917±0.002	6.6
	RGCN	0.693±0.019	0.652±0.087	0.573±0.086	O.O.M	0.578±0.023	0.538±0.009	8.8
	MEDIANGCN	0.592±0.101	0.683±0.081	0.618±0.030	O.O.M	0.583±0.128	O.O.T	8.8
	GNNGUARD	0.662±0.005	0.417±0.121	0.468±0.040	O.O.M	0.910±0.008	0.841±0.008	8.6
	METAGC	0.701±0.017	0.690±0.023	0.660±0.054	O.O.M	0.930±0.003	0.920±0.013	3.6
	LEO (Proposed)	0.860±0.009	0.869±0.003	0.820±0.023	O.O.M	0.942±0.006	0.939±0.002	**1.4**

Table 5: The additional experimental results regarding Q2 & Q3 on Rand. Attack.

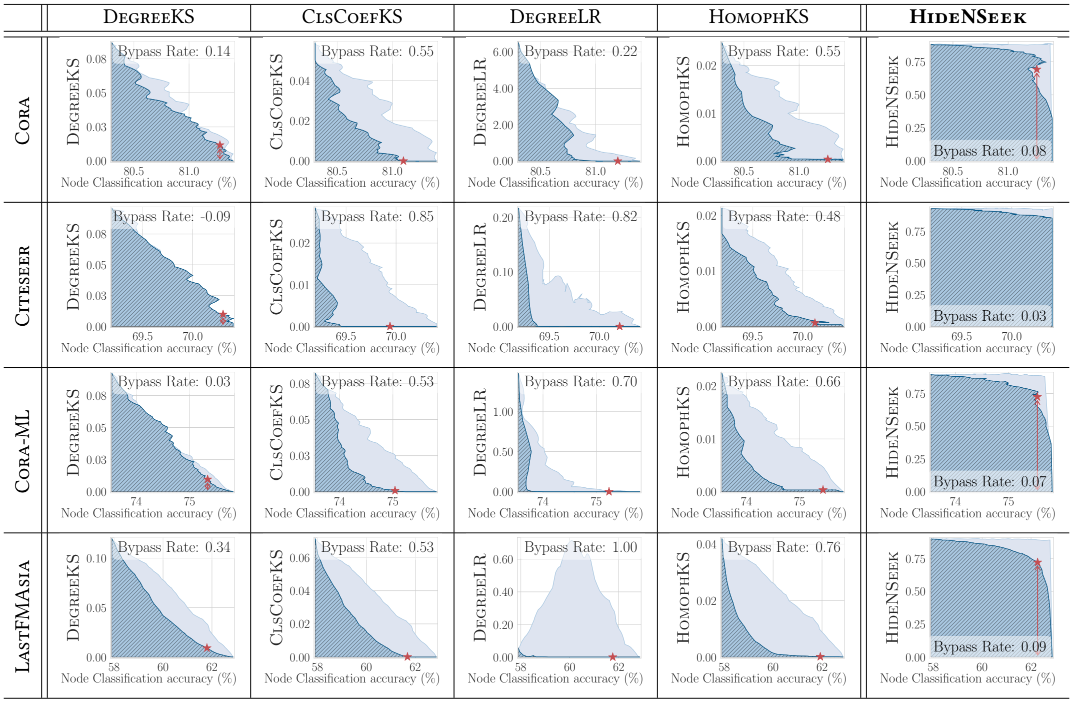


Table 6: The additional experimental results regarding Q2 & Q3 on DICE.

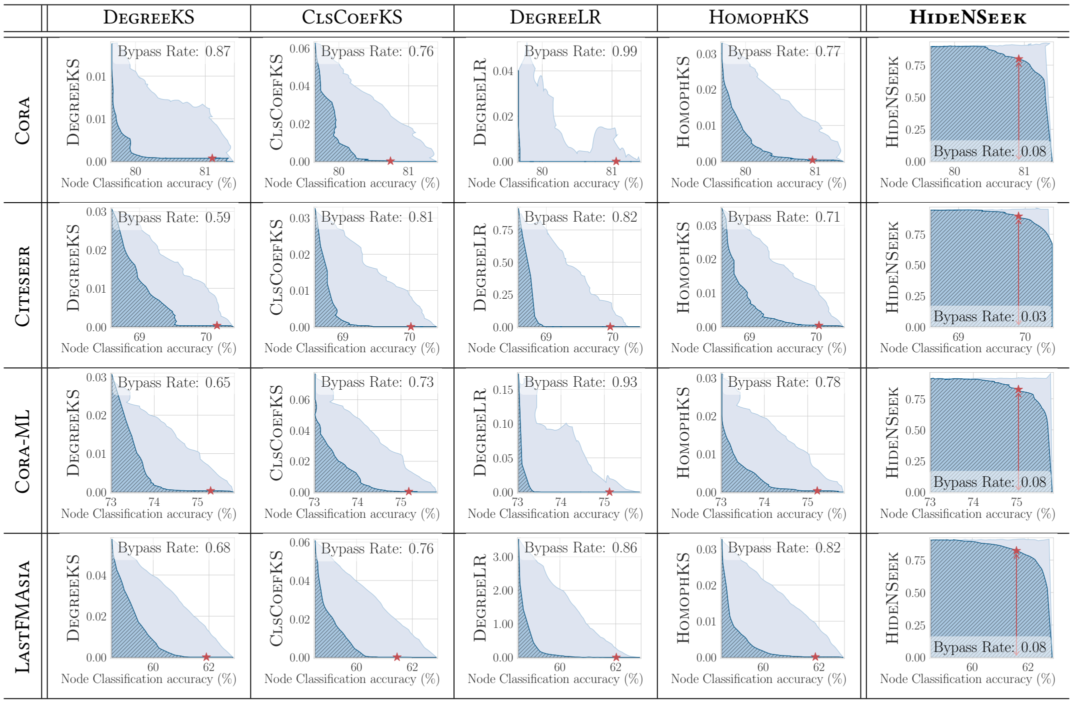


Table 7: The additional experimental results regarding Q2 & Q3 on Structack.

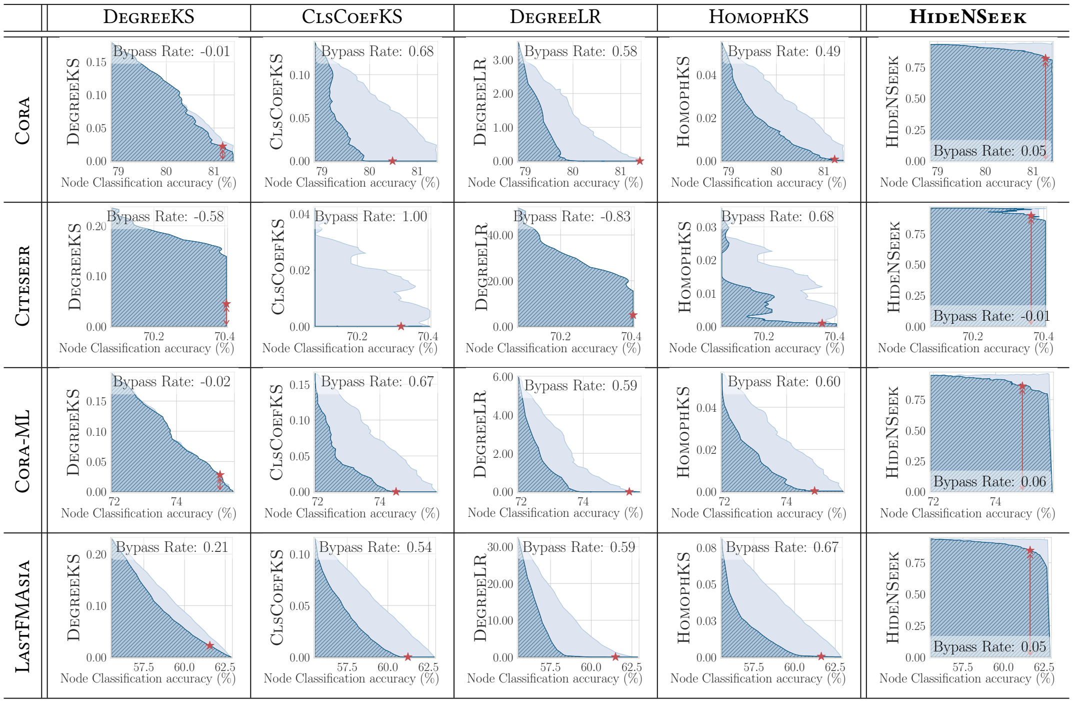
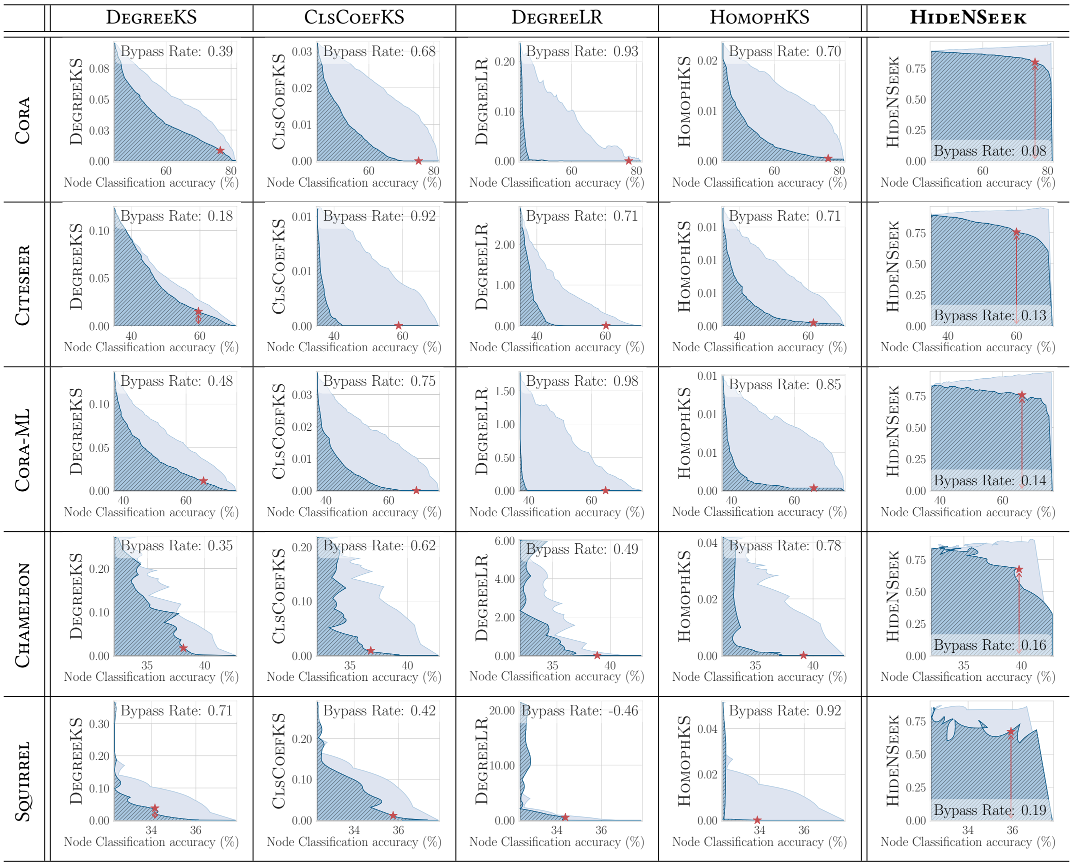


Table 8: The additional experimental results regarding Q2 & Q3 on Metattack.


DegreeKS
ClsCoefKS
DegreeLR
HomophKS
HideNSeek
Cora
Bypass Rate: 0.39
Bypass Rate: 0.68
Bypass Rate: 0.93
Bypass Rate: 0.70
Bypass Rate: 0.08
Citeseer
Bypass Rate: 0.18
Bypass Rate: 0.92
Bypass Rate: 0.71
Bypass Rate: 0.71
Bypass Rate: 0.13
Cora-ML
Bypass Rate: 0.48
Bypass Rate: 0.75
Bypass Rate: 0.98
Bypass Rate: 0.85
Bypass Rate: 0.14
Chameleon
Bypass Rate: 0.35
Bypass Rate: 0.62
Bypass Rate: 0.49
Bypass Rate: 0.78
Bypass Rate: 0.16
Squirrel
Bypass Rate: 0.71
Bypass Rate: 0.42
Bypass Rate: -0.46
Bypass Rate: 0.92
Bypass Rate: 0.19
Node Classification accuracy (%)