A. Supplementary Materials

A.1. Further Discussion

Hypergraphs are commonly used to model complex relationships between entities, often relying on homophily, the tendency for similar nodes to cluster together, as a key structural property that enhances predictive accuracy. In domains like social networks, recommender systems, and biological networks, this property forms a crucial foundation for tasks such as community detection and classification.

It is well understood that a higher homophily generally helps GNNs achieve better performance on downstream tasks, as neighbors positively influence the representation of the central node during aggregation (Zheng et al., 2022; Zhu et al., 2022; Luan et al., 2022; Huang et al., 2024). Reducing homophily between nodes disrupts the neighborhood aggregation process of graph-based model, resulting in performance degradation. However, this disruption can also be leveraged as an indicator of a potential attack. Therefore, by self-constraining the actions of attacker with respect to homophily, the attack can achieve greater stealth and evade detection more effectively. Also, the effect of other attack methods on the homophiliy ratio as shown in Figure 8.

Although this paper utilizes homophily as a restrictive metric for designing attack strategies, the complex structure of hypergraphs presents opportunities for deeper exploration. Investigating more sophisticated metrics for attack stealth will help uncover the fundamental principles needed to protect the stability of hypergraph models.

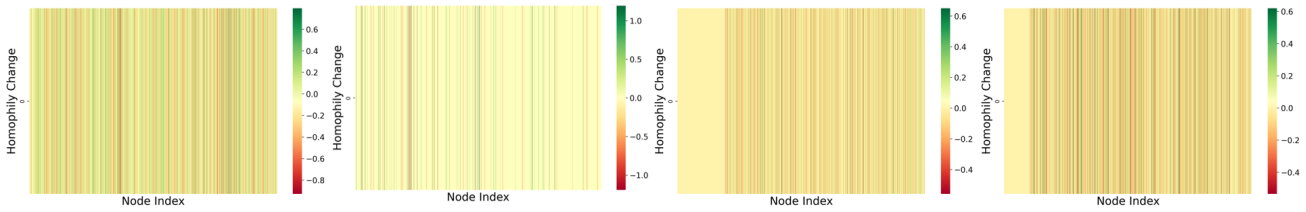

Figure 8. Impact of other attack methods on homophily ratio. The left two visuals show NDA attacks on the DBLP-CA dataset and Cora dataset, while the right two represent FGA attacks on the Pubmed dataset and Citeseer, respectively.

Relationship Between Changes in FHH and Changes in Hypernode Feature. Suppose that the hypergraph is perturbed by injecting nodes, and that the perturbation affects the set of neighbors $\mathcal{R}_v^t$ of node v. This perturbation causes a change in the node feature aggregation function, denoted as $f_t'(\cdot)$. By Taylor expansion, we know that the feature change $\Delta \mathbf{h}_v^{(k)}$ of supernode v is denoted as:

$$\Delta \mathbf{h}_v^{(k)} = \sum_{t=0}^{n} \frac{\partial \phi}{\partial f_t} \Delta f_t + O\left((\Delta f_t)^2\right). \tag{15}$$

According to the definition of Eq. 10, FHH measures the similarity between the features of node v and its neighboring feature aggregates. Then the change of FHH after perturbation can be approximated as:

$$\Delta FHH_v \approx \text{sim}(\mathbf{H}_v', \mathbf{X}_v') - \text{sim}(\mathbf{H}_v, \mathbf{X}_v). \tag{16}$$

Similarly, using the Taylor expansion, the result is obtained:

$$\Delta FHH_v \approx \sum_{t=0}^{n} \frac{\partial \phi}{\partial f_t} \Delta f_t + O\left((\Delta f_t)^2\right). \tag{17}$$

This suggests that changes in FHH are directly caused by changes in the characteristics of node v and its neighbors, which can be captured by the perturbation function Δf_t and the partial derivatives of the corresponding aggregation function $\frac{\partial \phi}{\partial f_t}$.

We believe that promoting a shift from heterophily to homophily is key to developing an unnoticeable attack. Our empirical findings indicate a significant reduction in the homophily ratio of hypergraphs following adversarial attacks. Homophily, which measures the similarity between a node's features and those of its neighboring nodes, is essential for preserving the structural integrity of hypergraphs. When homophily is disrupted, the structural coherence of the hypergraph is compromised,

making the attack more noticeable and less effective. Therefore, maintaining homophily is crucial for ensuring that the attack remains unnoticeable and achieves its desired impact. In hypergraphs, where nodes are connected by hyperedges that capture complex high-order relationships, even small changes in homophily can lead to significant topological shifts due to the collaborative nature of the hypergraph structure. As shown in Eq.(17), small perturbations may cause nonlinear changes, especially in higher-order structures where the effects may be amplified as the perturbations propagate through multiple nodes.

By integrating homophily constraint into the overall loss function, we ensure that our adversarial perturbations are optimized for both attack effectiveness and unnoticeable. The full adversarial objective is expressed as:

$$\max_{||\mathcal{G}'-\mathcal{G}||\leq\mathcal{B}} \mathcal{L}_{atk} = \mathcal{L}_{cls} - \lambda\mathcal{L}_{FHH}, \tag{18}$$

where $\mathcal{L}_{cls}$ is the classification loss and λ is a hyperparameter controlling the trade-off between classification performance and homophily preservation, and $||\mathcal{G}' - \mathcal{G}|| \leq \mathcal{B}$ ensures that the adversarial modifications stay within the allowed perturbation budget $\mathcal{B}$.

A.2. Experimental Setups

In our study, we follow the strict black-box setup, which prohibits any querying of the model and only allows access to the incidence and feature matrices of the input data. The default hyperparameter settings in our method are as follows, the parameter random seed s is 4202, the choice of hyperedge k is 10%, the ratio of the injected nodes α is 5%, the effect of controlling the loss of homophily λ and τ is 0.1. The comparison method is subjected to the same budgetary constraints. By default, the victim model and the target model are the same, where the victim model is the model used by the user and the victim model is the pre-trained model used by the agent model, which is the model attacked in our experiments. All experiments are conducted on a device with AMD EPYC 7543 32-core processor and a NVIDIA RTX A6000 GPU with 48 GB of RAM.

A.3. Experimental Results

We provide visualization results on other datasets, as shown in Figure 9 and Figure 10.

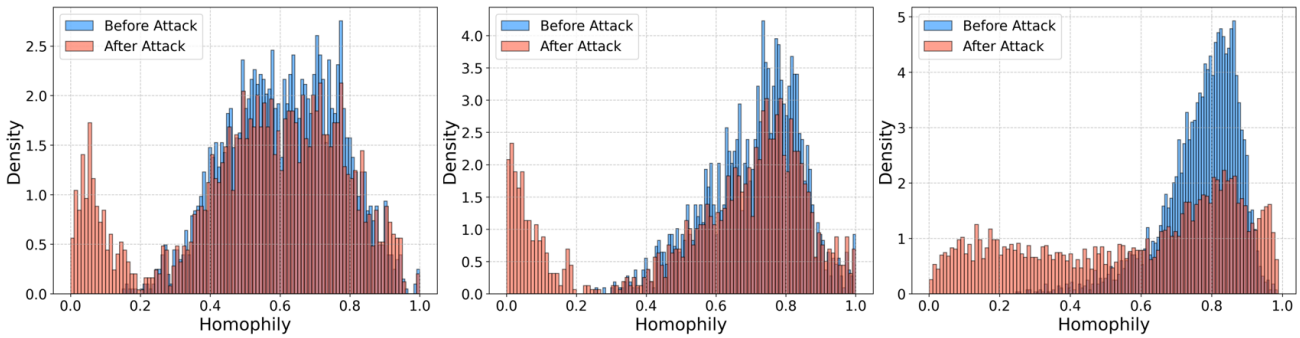

Figure 9. Changes in the homophily ratio. From left to right, Cora-CA dataset, Citeseer dataset, Pubmed dataset.

We also evaluated the portability of HyperNear to other HNN models in two additional datasets to complement the experimental results in the main text, as shown in Table 4.

A.4. Datasets

We evaluate our models on five real-world hypergraph datasets for hypernode classfication tasks, including DBLP (Rossi & Ahmed, 2015), Pubmed, Citeseer and Cora (Sen et al., 2008). These are standard academic web datasets, where each node represents a document. For the DBLP and Cora datasets, a co-authorship hypergraph is constructed, with all documents co-authored by the same author forming a single hyperedge. In the case of PubMed, Citeseer, and Cora, a co-citation hypergraph is created, where each hyperedge links all documents cited by the same author. The statistics of datasets are provided in Table 5.

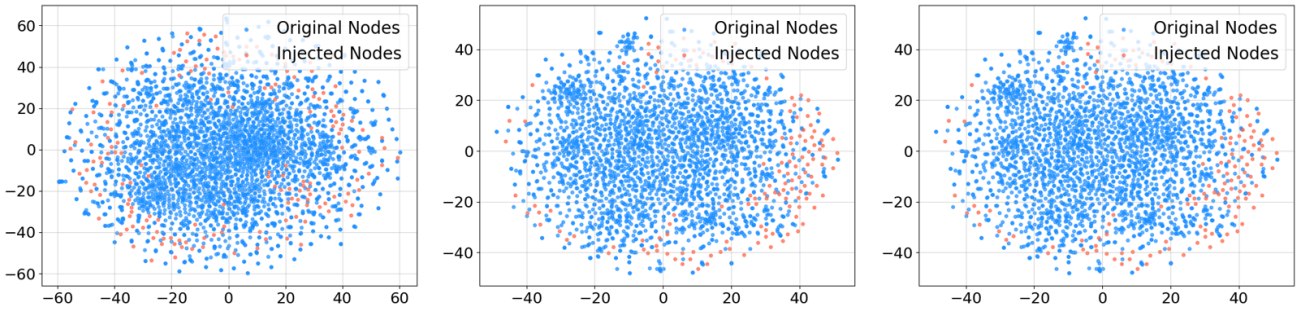

Figure 10. T-SNE visualization of data distribution after HyperNear attack. From left to right, Citeseer dataset, Cora dataset, Cora-CA dataset.

Table 4. Performance comparison of HyperNear on different HNNs.

Victim Model	Cora-CA		Citeseer	
	Before	After	Before	After
HyperGCN	75.26 ± 2.78	$57.36 \pm 7.42 \downarrow$	70.76 ± 0.92	$56.68 \pm 1.77 \downarrow$
HGNN	82.10 ± 0.86	$53.29 \pm 0.95 \downarrow$	72.71 ± 1.27	$46.14 \pm 1.53 \downarrow$
ED-HNN	84.08 ± 1.75	$62.38 \pm 1.18 \downarrow$	73.45 ± 1.36	$55.19 \pm 1.55 \downarrow$
UniGCNII	81.79 ± 1.29	$70.28 \pm 1.42 \downarrow$	73.26 ± 1.16	$68.50 \pm 1.37 \downarrow$

Table 5. Statistics of hypergraph datasets. The homophily ratio $\mathcal{H}$ is computed from Eq.(3) by labels.

Datasets	# Nodes ($\lvert\mathcal{V}\rvert$)	# Hyperedges ($\lvert\mathcal{E}\rvert$)	# Classes	Avg.hyperedge size	Label rate	Lable-based Homo. ($\mathcal{H}$)
Cora-CA	2,708	1,072	7	4.2±4.1	0.052	0.7797
DBLP-CA	43,413	22,535	6	4.7±6.1	0.040	0.8656
Citeseer	3,312	1,079	6	3.2±2.0	0.052	0.6814
Cora	2,708	1,579	7	3.0±1.1	0.052	0.7462
Pubmed	19,717	7,963	3	4.3±5.7	0.008	0.7765

A.5. Notation Table

Table 6 summarizes the notations and definitions throughout this paper for clarity.

Table 6. Notation Table.

Notation	Description	Notation	Description
$\mathcal{G}$	a hypergraph	$\mathbf{x}$	vector of vertex features
$\mathcal{V}$	set of vertexs	y	label
$\mathbf{W}$	diagonal matrix of hyperedge weights	$\mathcal{H}$	label-based homophily ratio
$\mathbf{X}$	matrix of vertex features	$\mathbf{I}_e$	matrix between the original vertex and the new hyperedge
$\mathbf{H}$	incidence matrix	$\mathbf{I}_v$	matrix between the new vertex and the original hyperedge
$\mathbf{H}'$	incidence matrix after attack	$\widehat{\mathbf{H}}$	matrix between the new vertex and the new hyperedge
e	hyperedge	$\mathcal{R}_v$	set of all hyperedges containing vertex v
v	vertex	d_e	the average degree of hyperedge

A.6. Algorithm

The implementation of our algorithm is summarized in Algorithm 1.

Algorithm 1 HyperNear: Node InjEction Attack on HypeRgraph

Input: Node feature matrix $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}|\times d}$, Hypergraph incidence matrix $\mathbf{H} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{E}|}$, Injection ratio α, Selected hyperedges k, Perturbation standard deviation η, Random seed s, Budget $\mathcal{B}$, Hyperparameter λ
Output: Modified node feature matrix $\mathbf{X}'$, Modified hypergraph incidence matrix $\mathcal{H}'$
Set random seed s
$n_{\text{injected}} \leftarrow \alpha \times |\mathcal{V}|$ {Number of injected nodes}
$\mathbf{X}_{\text{new}} \leftarrow$ Generate features for injected nodes using $\mathbf{X}$ with perturbation η
$\mathcal{E}_{\text{selected}} \leftarrow$ Select k hyperedges from $\mathbf{H}$ randomly
for $i = 1$ to k **do**
  Inject m nodes from $\mathbf{X}_{\text{new}}$ into hyperedge $\mathcal{E}_{\text{selected}}[i]$
  Compute $\mathcal{L}_{cls}$ for the modified hypergraph
  Compute $\mathcal{L}_{FHH}$ based on feature homophily preservation
  if $||\mathcal{G}' - \mathcal{G}|| \leq \mathcal{B}$ **then**
    Calculate $\mathcal{L}_{atk} \leftarrow \mathcal{L}_{cls} - \lambda\mathcal{L}_{FHH}$
  else
    Revert the injection
  end if
end for
while adversarial objective not met **do**
  $\mathbf{H}_{update} \leftarrow$ Perform random split on the injected hyperedge in $\mathbf{H}$
  Compute $\mathcal{L}_{cls}$ for $\mathbf{H}_{update}$
  Compute $\mathcal{L}_{FHH}$ for the updated features
  $S_{inj} \leftarrow \arg\max_S \mathcal{L}_{cls}(f(\mathbf{H}_{update}), y)$ {Find the optimal splitting strategy}
  if adversarial loss increases or $||\mathcal{G}' - \mathcal{G}|| > \mathcal{B}$ **then**
    Reject the split
  else
    Accept the new hyperedge split S_{inj}
  end if
end while
return $\mathbf{X}'$, $\mathcal{H}'$

B. Analyzing Hypergraph Topological Vulnerabilities

In this appendix, we present a detailed derivation of the theoretical framework that models the topological vulnerability of hypergraphs under adversarial perturbations, specifically focusing on hypergraph injection attacks. These attacks target hypergraphs by perturbing the node features and structure, causing significant changes in the model's performance. The impact of these perturbations is formally captured in Theorem 3.1. We begin by revisiting the general process of hypergraph convolution, before introducing the perturbations and analyzing their impact on the node feature vectors.

B.1. Hypergraph Convolution Process

We start by defining the general hypergraph convolution process, which aggregates information from neighboring nodes in a hypergraph. The feature vector for node v at the k-th layer, $\mathbf{h}_v^{(k)}$, is computed as a function of the feature vector from the previous layer, $\mathbf{h}_v^{(k-1)}$, and the aggregated features of the neighboring nodes:

$$\mathbf{h}_v^{(k)} = \phi(\mathbf{h}_v^{(k-1)}, f(\{\mathbf{h}_u^{(k-1)} \mid u \in e_j, e_j \in \mathcal{R}_v\})), \tag{19}$$

where $\mathcal{R}_v = \cup_{e_j \in \mathcal{E}}\{e_j \mid v \in e_j\}$ represents the set of hyperedges that contain node v, and e_j (for $j = 1, 2, \ldots, |\mathcal{E}|$) denotes the j-th hyperedge. The functions $\phi(\cdot)$ and $f(\cdot)$ are vector-valued functions that update the node features based on the information from neighboring nodes.

This equation captures the process of updating the feature vector for node v by aggregating the feature vectors of its neighbors across all hyperedges that contain v.

B.2. Adversarial Injection Attack on Hypergraphs

In the context of adversarial attacks, we introduce the concept of a hypergraph injection attack. This attack targets specific areas of the hypergraph by injecting new nodes, perturbing the structure and feature vectors of the hypergraph without needing full access to the graph. The goal is to simulate the effect of small, targeted changes to the hypergraph that lead to significant changes in the model's output.

To model the effect of adversarial perturbations, we introduce a modified version of the aggregation function where the neighborhood sets are altered by injected nodes. Specifically, we define $\mathcal{R}_v^t$ as the set of t-hop neighbors of node v, and the corresponding aggregation function is denoted as $f_t(\cdot)$. The feature update process can then be rewritten as:

$$\begin{aligned}\mathbf{h}_v^{(k)} =&\phi\Big(f_0(\{\mathbf{h}_v^{(k-1)}\}), f_1(\{\mathbf{h}_u^{(k-1)} \mid u \in e_j, e_j \in \mathcal{R}_v^1\}),\\ &f_2(\{\mathbf{h}_u^{(k-1)} \mid u \in e_j, e_j \in \mathcal{R}_v^2\}), \ldots, f_n(\{\mathbf{h}_u^{(k-1)} \mid u \in e_j, e_j \in \mathcal{R}_v^n\})\Big).\end{aligned} \tag{20}$$

In this formulation, the feature vector $\mathbf{h}_v^{(k)}$ is updated by aggregating the features from the node's t-hop neighbors across different layers of the hypergraph.

B.3. Perturbation of the Hypergraph

Next, we introduce perturbations into the hypergraph. We assume that after the injection attack, the t-hop neighborhood of node v changes from $\mathcal{R}_v^t$ to $\mathcal{R}_v'^t$, and the corresponding aggregation function $f_t(\cdot)$ changes to $f_t'(\cdot)$. The change in the feature vector $\mathbf{h}_v^{(k)}$ due to these perturbations can be approximated using a first-order Taylor expansion:

$$\begin{aligned}\mathbf{h}_v'^{(k)} =&\mathbf{h}_v^{(k)} + \frac{\partial\phi}{\partial f_0}(f_0' - f_0) + \frac{\partial\phi}{\partial f_1}(f_1' - f_1) + \cdots + \frac{\partial\phi}{\partial f_n}(f_n' - f_n) + O\left((\Delta f_0)^2 + (\Delta f_1)^2 + \cdots + (\Delta f_n)^2\right)\\ =&\mathbf{h}_v^{(k)} + \sum_{t=0}^{n}\frac{\partial\phi}{\partial f_t}\cdot(f_t' - f_t) + O\left((\Delta f_t)^2\right),\end{aligned} \tag{21}$$

where $\Delta f_t = f_t' - f_t$ represents the perturbation in the aggregation function for the t-hop neighbors. The change in the feature vector $\Delta\mathbf{h}_v^{(k)}$ is then:

$$\Delta\mathbf{h}_v^{(k)} = \mathbf{h}_v'^{(k)} - \mathbf{h}_v^{(k)} = \sum_{t=0}^{n} \underbrace{\frac{\partial\phi}{\partial f_t}}_{\text{Sensitivity of } \phi \text{ to } t\text{-hop features}} \cdot \underbrace{\Delta f_t}_{\text{Change in } t\text{-hop aggregated features}} + \underbrace{O((\Delta f_t)^2)}_{\text{Higher-order effects (nonlinear)}}. \tag{22}$$

This equation quantifies how the perturbations in the hypergraph propagate through the network and alter the feature vectors of the nodes.

B.4. Weighted Aggregation and Perturbation Effects

For the case where the aggregation function uses a weighted average, the function $f_t(\cdot)$ can be written as:

$$f_t(\{\mathbf{h}_u^{(k-1)} \mid u \in e_j, e_j \in \mathcal{R}_v\}) = \sum_{\{u \in e_j \mid e_j \in \mathcal{R}_v\}} w_u^t \mathbf{h}_u^{(k-1)}, \tag{23}$$

where w_u^t is the weight associated with the hyperedge between node v and its t-hop neighbors. Under perturbations, the change in the aggregation function is given by:

$$\Delta f_t = \sum_{\{u \in e_j \mid e_j \in \mathcal{R}_v\}} \left(\Delta w_u^t \mathbf{h}_u^{(k-1)} + w_u^t \Delta\mathbf{h}_u^{(k-1)}\right). \tag{24}$$

This expression demonstrates that the changes in both the structure (i.e., the weights) and the feature vectors of the neighbors amplify the perturbation effect.

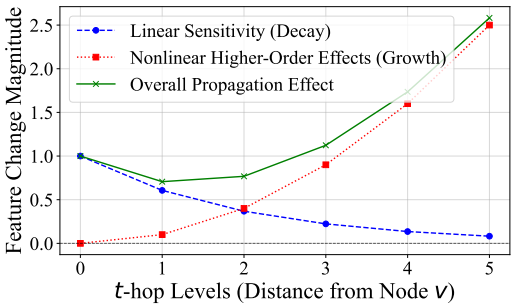

Figure 11. Illustration of perturbation propagation in HNNs. In Theorem 3.1, the first term models the primary contribution through linear propagation, which dominates for small perturbations. In contrast, the higher-order terms reflect nonlinear effects that, while smaller in magnitude, can become significant under large perturbations or high variability in hyperedge weights.

B.5. Optimization Objective for Perturbations

A key aspect of adversarial perturbations is to minimize the magnitude of the change in the feature vector $\Delta \mathbf{h}_v^{(k)}$ while ensuring that the perturbations in the aggregation function exceed a certain threshold. This can be formulated as the following optimization problem:

$$\min_{\Delta \mathbf{h}_v^{(k)}} \parallel \Delta \mathbf{h}_v^{(k)} \parallel \quad \text{s.t.} \quad \sum_{t=0}^{n} \parallel \Delta f_t \parallel \geq \epsilon, \tag{25}$$

where $\parallel \Delta \mathbf{h}_v^{(k)} \parallel$ represents the magnitude of the change in the feature vector, and $\parallel \Delta f_t \parallel$ is the perturbation size in the aggregation function for the t-hop neighbors. The constraint ensures that the perturbation exceeds a predefined threshold ϵ, which is necessary to produce significant changes in the output.

B.6. Perturbation Propagation in HNNs

Figure 11 illustrates the sensitivity brought about by t-hop neighbors of node v versus the change in uncertainty brought about by nonlinear higher-order effects on HNNs. These theorem and corollary offer a theoretical perspective on how perturbations in hypergraph topology propagate through the network. While the results illustrate the potential for small manipulations in hyperedges or node features to influence model behavior, the exact extent of this effect depends on specific model designs and datasets. As observed in Sec. 5.3, experimental results confirm that our injection attack framework, HyperNear, effectively exploits these vulnerabilities across multiple HNN models, demonstrating strong disruptive capabilities under diverse scenarios.

Note: Optimization Objective for Adversarial Attacks

To design effective injection attacks, the following optimization problem formalizes the trade-off:

$$\min_{\Delta \mathbf{h}_v^{(k)}} \parallel \Delta \mathbf{h}_v^{(k)} \parallel \quad \text{s.t.} \quad \sum_{t=0}^{n} \parallel \Delta f_t \parallel \geq \epsilon, \tag{26}$$

where $\parallel \Delta \mathbf{h}_v^{(k)} \parallel$ represents the magnitude of the change in the feature vector, and $\parallel \Delta f_t \parallel$ is a measure of the size of the perturbation in the aggregation function $f_t(\cdot)$, which we want to be at least ϵ, a predefined threshold for significant change.

Objective: Identify minimal perturbations that lead to significant changes in the output.

C. Related Works

Adversarial Attack on Graph Data. Existing attack methods (Dai et al., 2018; Zügner et al., 2020; Sun et al., 2022; Gosch et al., 2023) on graph data can be categorized into two types: graph modification attacks (Chen et al., 2018; Zügner et al., 2018; Zügner & Günnemann, 2019) and graph injection attacks (Sun et al., 2020; Zou et al., 2021; Ju et al., 2023; Zhang et al., 2024; Fang et al., 2024). Modification attacks achieve the desired attack effect by altering the graph structure, while injection attacks aim to disrupt the model's performance by injecting new nodes. Both types of attacks can significantly decrease the model's accuracy, however, injection attacks generally require less access privilege from the attacker. Thus, this paper focuses primarily on designing an injection attack framework.

Most prior work has only addressed simple graph structures (Wu et al., 2019), with edges representing direct connections between node pairs. In contrast, hypergraphs, which model higher-order relationships, have seen limited exploration in the context of adversarial attacks. This gap is particularly critical given the increasing adoption of HNNs in real-world applications, as the absence of robust understanding about potential vulnerabilities could expose these models to significant security risks.

Hypergraph Neural Networks. HNNs have been developed as an advanced extension of GNNs, specifically designed to capture high-order relationships that go beyond the pairwise interactions typical of traditional graph structures (Gao et al., 2020; Antelmi et al., 2023; Kim et al., 2024). Hypergraphs use hyperedges to simultaneously connect multiple nodes, enabling HNNs to model the more intricate and varied relationships found in real-world data. This feature makes HNNs particularly effective in domains such as social networks (Li et al., 2013; Yang et al., 2019), bioinformatics (Zheng et al., 2019; Shao et al., 2020; Xiao et al., 2019), and recommendation systems (Yu et al., 2021; La Gatta et al., 2022), where interactions often involve groups of nodes rather than individual pairs.

Various models have been proposed: HGNN (Feng et al., 2019; Gao et al., 2022), HyperConv (Bai et al., 2021), and HyperGCN (Yadati et al., 2019) process hypergraphs by converting them into traditional graphs through the hypergraph Laplacian operator. HyperMSG (Arya et al., 2020) leverages hypergraph structures by aggregating messages in a two-stage process. UniGNN (Huang & Yang, 2021) proposes a unified framework for both graph and hypergraph neural networks. HyperGCL (Wei et al., 2022) enhances the generalization ability of hypergraphs through contrastive learning, while ED-HNN (Wang et al., 2023) models high-order relationships using a hypergraph diffusion operator. HCoN (Wu et al., 2022) introduces hypergraph reconstruction error to train a classifier. These approaches enhance the ability of HNNs to handle large-scale data with complex multi-node relationships, enabling the development of hypergraph applications. Despite these advances, the security and robustness of HNNs remain underexplored.

Remarks. While some initial efforts (Hu et al., 2023; Chen et al., 2023) have explored related threats, these studies mainly focus on probing different data modeling approaches rather than designing attacks specifically for hypergraph models, which are central to many applications. Consequently, research addressing the unique structural characteristics of hypergraphs in adversarial scenarios remains scarce.

Our work directly tackles this gap by focusing on the structural vulnerabilities of HNNs under adversarial manipulation, with an emphasis on black-box attack settings. Black-box attacks (Xu et al., 2022; Wen et al., 2024), where adversaries generate adversarial samples without any knowledge of the target model's internal parameters or architecture, pose realistic and severe threats in practical scenarios. To the best of our knowledge, this is the first work to conduct adversarial attacks on hypergraphs in a black-box setting.