Towards Understanding and Reducing Graph Structural Noise for GNNs

Mingze Dong [1 2] **Yuval Kluger** [1 2 3]

Abstract

Graph neural networks (GNNs) have emerged as a powerful paradigm to learn from relational data mostly through applying the message passing mechanism. However, this approach may exhibit suboptimal performance when applied to graphs possessing various structural issues. In this work, we focus on understanding and alleviating the effect of graph structural noise on GNN performance. To evaluate the graph structural noise in real data, we propose edge signal-to-noise ratio (ESNR), a novel metric evaluating overall edge noise level with respect to data features or labels based on random matrix theory. We have found striking concordance between the proposed ESNR metric and the GNN performance in various simulated and real data. To reduce the effect of the noise, we propose GPS (Graph Propensity Score) graph rewiring, which estimates the edge likelihood for rewiring data graphs based on self-supervised link prediction. We provide a theoretical guarantee for GPS graph rewiring and demonstrate its efficacy by comprehensive benchmarks.

1. Introduction

Graph neural networks (GNNs) represent a framework to learn from relational data by message passing on the provided data graph (Scarselli et al., 2008; Kipf and Welling, 2016a;b; Gilmer et al., 2017; Hamilton et al., 2017). Despite the prevalence of the message passing mechanism in GNNs, recent works pointed out several limitations of message passing neural networks, including limited power (Xu et al., 2018; Morris et al., 2019; Maron et al., 2019), oversmoothing (Nt and Maehara, 2019; Oono and Suzuki, 2019), and oversquashing (Alon and Yahav, 2020; Topping et al., 2021). Moreover, GNNs may even be outperformed by traditional NNs when the graphs exhibit significant heterophily (Zhu et al., 2021; Yan et al., 2021).

There are two general strategies towards overcoming the limitations of traditional GNNs. One is by designing more dedicated GNN architectures, such as H2GCN (Zhu et al., 2020) and CPGNN (Zhu et al., 2021) for heterophily, GIN (Xu et al., 2018) for GNN expressive power, and +FA (Alon and Yahav, 2020) for oversquashing respectively. In this work, our focus is on the other strategy named graph rewiring, which aim to modify the original data graph such that the new graph has better properties (Bi et al., 2022; Klicpera et al., 2019; Topping et al., 2021).

Despite existing works targeting various graph properties for improving GNN performance, the effect of graph structural noise on GNN training is not yet systematically characterized. This is partly due to the lack of structural noise metrics for the structural noise in real graph-structured data. Furthermore, existing graph denoising methods are based on either only the graph structure (such as DIGL (Klicpera et al., 2019)) or joint optimization resulting in task-dependent rewired graphs (Jin et al., 2020; Yu et al., 2021; Franceschi et al., 2019; Zheng et al., 2020; Luo et al., 2021; Arnaiz-Rodríguez et al., 2022; Dai et al., 2022). The potential of a expressive feature-aware graph rewiring method that learns a task-independent denoised graph remains to be explored.

Our contributions. In this work, we aim to both understand and minimize the effect of structural noise level in GNN training. Our key contributions in the work are two-folds:

- We propose a novel metric, edge signal-to-noise ratio (ESNR), to estimate the overall graph structural noise level with respect to either node features or node labels. We validate the metric in various synthetic and real data and show it has strikingly high consistency with GNN learning performance.
- We propose a graph rewiring framework named graph propensity score (GPS) that denoises graphs in a feature-aware manner based on self-supervised training. We provide both theoretical guarantee and extensive benchmarking showing the efficacy of the GPS framework combining with the ESNR metric.

[1]Interdepartmental Program in Computational Biology & Bioinformatics, Yale University, New Haven, CT, USA [2]Department of Pathology, School of Medicine, Yale University, New Haven, CT, USA [3]Applied Math Program, Yale University, New Haven, CT, USA. Correspondence to: Yuval Kluger <yuval.kluger@yale.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

2. Related works

Structural noise on GNN training. Numerous studies have investigated the effects of structural noise on GNN training, with most falling under the category of "robust GNNs". Several of these methods propose to rewire graphs through joint optimization of graph structures and network parameters with different perspectives, such as imposing no structural constraints (Yu et al., 2021), maximizing graph regularity and feature smoothness (Jin et al., 2020), formulating a bilevel program (Franceschi et al., 2019), enforcing graph sparsification (Zheng et al., 2020; Luo et al., 2021), and leveraging the lovasz bound (Arnaiz-Rodríguez et al., 2022). Another recent work (Dai et al., 2022) considers the setting with both graph structural noise and sparse label, proposing a MLP-based link predictor for constructing a denoised densified graph. (Zhang and Pei, 2021) gives a systematic comparison of different approaches for best robustness on noisy data based on simulated noise. Apart from structural noise's impact on GNN training, its impact on GNN interpretabilty is discussed in (Li et al., 2022).

Graph rewiring. Graph rewiring aims to decouple the computational graph from the input graph thus improving GNN's ability in task-specific learning. The graph rewiring framework differs from the aforementioned robust GNN approaches as they aim to improve GNN performance by learning the graph structure prior to the task. Various strategies have been proposed for graph rewiring with different theoretical or practical considerations. GraphSAGE (Hamilton et al., 2017) considers random sampling links from the graph to control the maximum link number for each node, thus increasing the efficiency of GNN training. DIGL (Klicpera et al., 2019) proposes the use of personalized Pagerank (PPR) (Haveliwala et al., 2003) and heat kernels on input graphs to locally smooth the graph structure. In order to resolve the over-squashing issue, SDRF (Topping et al., 2021) proposes the concept of balanced graph curvature and optimizes the graph structure to eliminate links with highly negative curvature. DHGR (Bi et al., 2022) aims to improve the data homophily by maximizing the graph's concordance with neighbor distributions of data features and labels, potentially causing label leakage. Another set of works (Wang et al., 2019; Fatemi et al., 2021; Kazi et al., 2022) considered cases where the graph structures are latent and need to be inferred from data.

3. Edge signal to noise ratio (ESNR)

3.1. Problem statement

Our first focus in this work is to develop an estimation framework in order to quantify edge noise level for graph-structured data (X, G), where $X \in \mathbb{R}^{n\times p}$ denotes the data feature attributes and $G = (V, E)$ denotes the data graph. For convenience, we define $A \in \{0, 1\}^{n\times n}$ as the graph adjacency matrix, which is used more often throughout the text. We start by discussing the considerations in defining our metric for edge noise level. Ideally, such a metric should satisfy the following properties:

Statistical characterization of the graph. The probabilistic model of the graph needs to be defined in order to derive the expression of edge noise in a rigorous manner. The model should be flexible enough to model real-world graphs while giving tractable estimators.

Feature / label awareness. Due to the existing data feature attributes, the edge noise label should not be solely determined by the graph edge indices. For instance, a graph's high-rank behavior may align with its feature attributes, and therefore, should not be misconstrued as noise.

We address the first desired property by adopting a Bayesian viewpoint of the graph generation. Specifically, in the directed graph case, we consider the adjacency matrix $\{A_{ij}\}$ as i.i.d. sampled from a latent probability matrix $\{P_{ij}\}$:

$$\forall ij,\ P(A_{ij} = 1) = P_{ij}, P(A_{ij} = 0) = 1 - P_{ij}. \quad (1)$$

In the undirected graph case, the adjacency matrix can be similarly defined as follows, where $A_{ji} \equiv A_{ij}$:

$$\forall i > j, P(A_{ij} = 1) = P_{ij}, P(A_{ij} = 0) = 1 - P_{ij} \quad (2)$$

By the definition, the adjacency matrix can be decomposed as the sum of the ground truth probability matrix term and a noise term with zero expectation $\mathcal{E}(P)$: $A = P + \mathcal{E}(P)$.

For addressing the second property, we define the row aggregated adjacency matrix C with respect to a node partition defined by a one-hot matrix $L \in \mathbb{R}^{m\times n}$. L can be defined by data label, clustering of data feature attributes, or other partitions of interest. The matrix C can also be decomposed as $C \equiv LA = LP + L\mathcal{E}(P) \equiv C^p + C^\epsilon$, where C^p represent the underlying ground truth signals and C^ϵ represent the contributions of noise. Our proposed task is to construct meaningful edge noise level indicators from data by estimating the signal strength C^p with respect to C^ϵ.

Interpreting the matrix C. We note even before decomposing the signal and noise terms, C provide meaningful summaries of the graph. For instance, when using the one-hot embedding of data labels as L, the edge homophily metric (Pei et al., 2020) can be directly calculated as follows:

$$\text{Edge homophily} = \frac{\text{tr}(\hat{C})}{\sum_{i,j}\hat{C}_{ij}}, \quad \text{where } \hat{C} \equiv CL^T. \quad (3)$$

Moreover, when L is constructed by data feature clustering labels, equation 3 can represent "unsupervised homophily" as an analogy of the original edge homophily.

The matrix C itself provides a summary of edge statistics at the single node level as each column of C represents the label neighborhood distribution of the node. The label neighborhood distribution has direct connections to GNN learning as discussed in (Ma et al., 2021). Notably, if we consider the singular value decomposition of $C = \sum_i \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$, then each $\boldsymbol{u}_i \in \mathbb{R}^m$ represents an orthogonal label neighborhood pattern with strength σ_i, invoking the need of understanding the spectral distribution for C^p and C^ϵ to distinguish true signal patterns from noise.

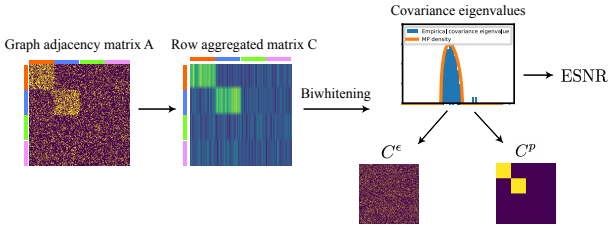

Figure 1: Illustration of the ESNR computation workflow. We first aggregate rows of the original graph adjacency matrix based on a predetermined label and perform the biwhitening transformation. The resulting matrix's bulk singular value spectrum follows the Marchenko-Pastur distribution (C^ϵ), while "outliers" denote meaningful signals (C^p). The ESNR is defined as the average element-wise spectrum strength of C^p relative to the MP upper-edge.

3.2. Spectrum separation via biwhitening

Intuitively, the spectral distribution of the matrix C^ϵ may be derived from either classical statistical theory or random matrix theory assuming each entry of C^ϵ is i.i.d. distributed. However, in our case, the distribution for each entry of C^ϵ is determined by different blocks of P thus not identically distributed, leading to heteroskedastic noise. A number of methods have been proposed to identify the number of matrix signals (ranks) or other spectrum statistics in the heteroskedastic noise setting (Bigot et al., 2017; Hong et al., 2020; Landa et al., 2022). In this work, we choose the *biwhitening* approach (Landa et al., 2022) as it requires little prior information except for the distribution form, which we do have for the graph as assumed by equation (1) and (2).

While it has been noted that the original biwhitening approach cannot apply to Bernoulli heteroskedastic noise in the original work (Landa et al., 2022), such an issue is avoided in our approach as we consider the matrix C instead of the original adjacency matrix. In fact, when the graph size is large and each P_{ij} is "uniformly" small, C^ϵ_{ij} can be approximated by independent or weakly dependent Poisson r.v.s, as stated by the following proposition:

Proposition 3.1. *Assuming equation (1) or (2), and* $\lim_{n\to\infty} \sum_j P_{ij}^2 = 0$ *for any* i. *Then for any column one-hot matrix* $L \in R^{m\times n}$ *such that* $\forall i, j, C^p_{ij} > 0$, *we have*

$$\forall i, j,\ C_{ij} \xrightarrow{d} \text{Poisson}(C^p_{ij}). \tag{4}$$

Furthermore, in the case of eq (1) (directed graph), each entry C_{ij} *is independently sampled. In the case of eq (2) (undirected graph),* C_{ij}*s are asymptotically uncorrelated.*

The proof of Proposition 3.1 is provided in Appendix A.1. We note that although the original theoretical guarantee for biwhitening approach is established for matrix with independent entries (corresponding to the directed graph case), the dependence across entries is also weak in the undirected graph case. Therefore we anticipate the performance of biwhitening will remain unharmed for undirected graphs, as also illustrated in the next section.

The scheme of applying Sinkhorn-Knopp (Sinkhorn, 1964; Cuturi, 2013) based biwhitening method for the aggregated adjacency matrix C is summarized in Algorithm 1.

Algorithm 1 Biwhitening for matrix $C \in \mathbb{R}^{m\times n}$.

1: Initialize: $\boldsymbol{x} = \mathbf{1}_m, \boldsymbol{y} = \mathbf{1}_n$, tolerance $\delta > 0$.
2: **while** $\max|\boldsymbol{x} \odot (C\boldsymbol{y}) - n \cdot \mathbf{1}_m| > \delta$ or $\max|\boldsymbol{y} \odot (C^T\boldsymbol{x}) - m \cdot \mathbf{1}_n| > \delta$ **do**
3: $\quad \boldsymbol{y} \leftarrow m \cdot \mathbf{1}_n \odot (1/C\boldsymbol{x})$ {$1/\boldsymbol{x}$: element-wise division}
4: $\quad \boldsymbol{x} \leftarrow n \cdot \mathbf{1}_m \odot (1/C\boldsymbol{y})$
5: **end while**
6: Return $C' = \text{diag}(\sqrt{\boldsymbol{x}})C\text{diag}(\sqrt{\boldsymbol{y}})$.

After applying biwhitening for matrix C in the directed case, under additional assumptions specifying the upper and lower bounds for entries of C, we show that (Appendix B) in the random matrix theory (RMT) regime where $\lim_{n\to\infty} \frac{m}{n} \to \beta \in (0, 1)$, the empirical covariance eigenvalue distribution of the noise component C^ϵ in the biwhitened matrix C' converges to the Marchenko-Pastur distribution:

$$\begin{aligned} \mu(x) &= \frac{1}{2\pi\beta x}\sqrt{(\beta_+ - x)(x - \beta_-)}, \\ \text{where } \beta \equiv \frac{m}{n} \in (0, 1), &\quad \beta_\pm = (1 \pm \sqrt{\beta})^2. \end{aligned} \tag{5}$$

Our proof (Appendix B) extends the original theory developed for biwhitening (Landa et al., 2022) to Poisson binomial distributions, taking advantage of its asymptotic equivalence with Poisson distributions (Proposition 3.1).

In order to separate the noise and signal terms for matrix C', we consider the singular value decomposition of C':

$$\begin{aligned} C' &\equiv U\text{diag}(\Sigma)V \\ &= U[\text{diag}(\Sigma - \alpha)_+ + (\Sigma - (\Sigma - \alpha)_+)]V. \end{aligned} \tag{6}$$

Here the first term $(\Sigma - \alpha)_+$ is obtained by truncating the total singular values with the threshold α that filters out the

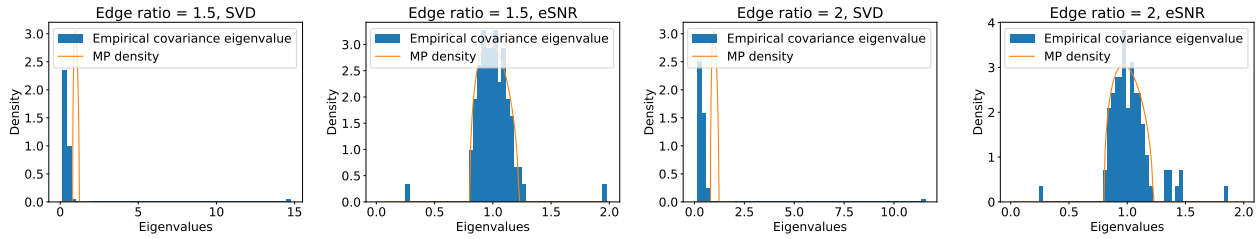

Figure 2: Comparisons of empirical covariance eigenvalue distributions of the original matrix (SVD) and the biwhitened matrix (ESNR) against MP distribution in high edge noise (edge ratio = 1.5) and low edge noise (edge ratio = 2) settings.

null distribution. A natural choice of α is the right boundary of the MP distribution: $\alpha = \sqrt{n\beta_+} = \sqrt{m} + \sqrt{n}$.

Our derivation primarily focuses on the RMT regime where we have $\lim_{n\to\infty} \frac{m}{n} \to \beta \in (0, 1)$. However, another "classical statistics" regime exists where $\lim_{n\to\infty} \frac{m}{n} = 0$. While we cannot distinguish the two regimes from finite sized data, we note here that the MP distribution converges to the point mass at 1 in distribution when $\beta \to 0$, in which case the upper boundary β_+ also tends to 1. As a result, the MP threshold $\alpha = \sqrt{n\beta_+} = \sqrt{m} + \sqrt{n}$ can cover both regimes and serve as an effective measure.

Finally, we define the edge signal to noise ratio (ESNR) by the average normalized value of matrix C''s spectrum. Different from the original definition of SNR, here we consider the ratio of the signal relative to the sum of signal and noise, rescaling the metric to the interval $[0, 1]$. The workflow of ESNR computation is summarized in Figure 1.

Definition 3.2. (ESNR) Denote the singular values of C' after substracting its mean value as $\{\Sigma_i\}_{i\in\{1,\dots,m\}}$, then

$$\text{ESNR} := \frac{1}{m}\sum_{i=1}^{m} \frac{\max(\Sigma_i - (\sqrt{m}+\sqrt{n}), 0)}{\Sigma_i} \tag{7}$$

3.3. Validation of ESNR

Here we first evaluate the proposed ESNR framework in the contextual stochastic block model, which is an extension of stochastic block model (SBM) and widely used in theoretical analysis for graph neural networks (Keriven et al., 2020; 2021; Ruiz et al., 2020; Wei et al., 2022):

Definition 3.3. (Contextual stochastic block model, CSBM): *Suppose $G(V, E)$ is a graph with n nodes and k communities. $Z \in \mathbb{R}^{n\times k}$ is a one-hot matrix encoding each node's membership. Each node's feature attribute X_i is independently sampled from $\mathbb{P}_{Z_i}$. The expectation μ_h for each $\mathbb{P}_{h\in\{1,2,\dots,k\}}$ exists and is different for each h. Suppose $C \in \mathbb{R}^{k\times k}$ is a symmetric matrix, with entry C_{ab} defining the probability of community a connecting with community b. Then the probabilistic edge matrix is defined by $P = ZCZ^T$, with each edge of the graph A_{ij} sampled with probability P_{ij}.*

When $C = (p-q)I + q\mathbf{1}\mathbf{1}^T$, and all communities are of equal sizes, we name the model as **simplified CSBM**. Note here p and q represent within-community and across-community edge probability respectively. In this case, the edge signal level should be qualitatively indicated by the probability gap $|p-q|$, meaning it can be both high in the $p \gg q$ regime (strong homophily) and the $q \gg p$ regime (strong heterophily), as in both cases the feature embeddings after message passing are separable. The intuition is rigorously stated as follows (Proposition 3.4, with a proof in Appendix A.2) by defining edge signal level as mutual information between edges and ground truth probabilities:

Proposition 3.4. (Mean edge mutual information of the original graph in simplified CSBM): *Assume the data is generated according to definition 3.3. Then we have*

$$\frac{1}{n^2}\sum_{ij} I(A_{ij}, P_{ij}) = h(\frac{p+(k-1)q}{k}) - \frac{1}{k}h(p) - \frac{k-1}{k}h(q).$$

Here h denotes the entropy function:

$$h(x) = x\log\tfrac{1}{x} + (1-x)\log\tfrac{1}{1-x}, \quad x \in (0,1).$$

By Jensen inequality, when the probability sum $\frac{p+(k-1)q}{k}$ is fixed, the larger the gap $|p-q|$, the higher the edge signal level. To see this property is also captured by the ESNR metric, we consider the asymptotic setting ($n \to \infty$), in which case under several technical assumptions (Theorem B.4 in Appendix B), due to the scaling factor convergence, a straightforward calculation reveals that the signal component C^p of the processed matrix C' satisfies:

$$\sigma_i(C^p) = \begin{cases} \frac{n|p-q|}{\sqrt{(p+(k-1)q)k}} & (i \le k-1), \\ 0 & (i = k). \end{cases} \tag{8}$$

Meanwhile, the covariance spectrum of the noise component C^ϵ converges to the MP distribution determined only by the ratio $\beta = k/n$ (Theorem B.2 in Appendix B). Therefore, the singular value Σ_i of C' is determined by eq (8) up to a constant fluctuation, meaning the ESNR (Definition 3.2), an increasing function of Σ_i, is also an increasing function of $|p-q|$ when k, n and $\frac{p+(k-1)q}{k}$ are fixed. Also, eq (8)

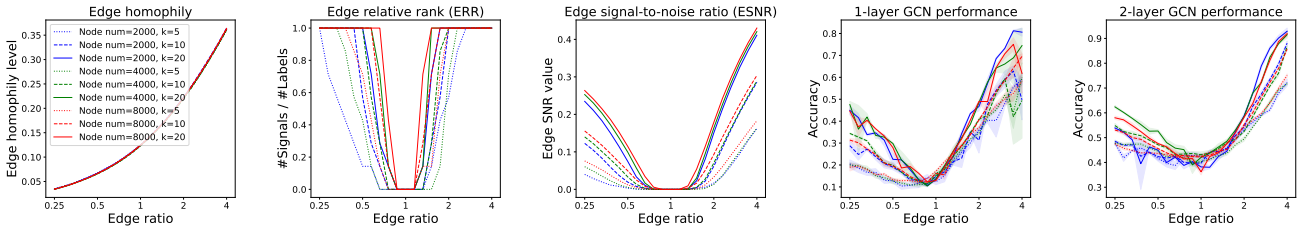

Figure 3: Comparisons of edge homophily, edge relative rank, and ESNR with 1 & 2 layer GCN performance in simulated CSBM data of different node number / average degree settings as functions of the edge ratio (p/q).

reveals that increasing p in the homophily scheme is more effective than increasing q in the heterophily scheme for improving ESNR, consistent with observed positive correlations between GNN performance and data homophily level (Zhu et al., 2021; 2020; Lim et al., 2021).

Before validating the relevance between ESNR and GNN performance, we aim to show qualitatively how the aggregated matrix of simulated CSBM is characterized by MP law before / after biwhitening for high (edge ratio = 1.5) and low structural noise (edge ratio = 2) respectively. In our simulation, the matrix L is defined from the hierarchical clustering of data feature attributes. Further experimental details can be seen in the Appendix D.1.

The simulation results are shown in Figure 2. For both low and high structural noise, the eigenvalue distribution of the original covariance matrix matches poorly with the Marchenko-Pastur distribution (indicated by the orange curve) due to the heteroskedastic noise. In contrast, after biwhitening scaling, the bulk covariance eigenvalue distributions show strong concordance with the MP distribution in both cases. Moreover, we observe the signal number (the number of eigenvalues that pass the right boundary of the MP distribution) show dramatic differences in the two settings (Figure 2), consistent with our theoretical analysis.

To further determine the validity of the proposed ESNR metric as a proxy for GNN performance, we conducted a comprehensive analysis of different metrics in simulated undirected CSBM datasets across various settings of graph sizes, average degrees, and edge ratios (Figure 3). Specifically, we compared our proposed ESNR metric against the edge homophily metric, the edge relative rank (ERR) defined by the ratio of the number of eigenvalues larger than MP upper bound and the total label number (Figure 3), and a vanilla entry-wise SNR metric (Appendix C). In this analysis, the matrix L is defined from the data label, with further experimental details available in the Appendix D.2.

In this setting, our experiments indicate that our proposed ESNR is the only measure that provides consistent trends with the true GNN performance, as it not only reveals the non-monotonic GCN performance along the edge ratio axis but correctly reflects the higher performance similarity observed in data with the same average degrees. Notably, neither the non-motonotic trend nor the relative GCN performance is reflected through the edge homophily metric (Figure 3) or the vanilla SNR (Appendix Figure 9), while the edge relative rank reveals only the non-motonotic pattern but not relative GCN performance (Figure 3). Finally, we observe that the ESNR is smaller in heterophily data compared with homophily data of the same absolute log edge ratio, consistent with the GNN training performance. This observation may suggest previous observations about the effect of homophily level on GNN performance may be partly spurious due to the difference in edge noise level.

Finally, we evaluated if ESNR can correctly reveal the impact of structural noise level on real data. These evaluations can be conducted on two fronts: either based on a single dataset or across multiple datasets. For the former evaluation, we here consider the effect of random edge dropout, which intuitively increases the edge noise level regardless of ground truth. Mathematically, the random edge dropout can be represented by a binary mask $M \in \mathbb{R}^{n\times n}$ with each entry M_{ij} i.i.d sampled from a Bernoulli distribution with parameter α. The parameter α indicates the preservation ratio of edges in the original graph ($1-\alpha$ stands for the edge dropout ratio). In this case, ESNR converges to a increasing function of α, as after biwhitening, the signal component C^p is proportional to $\alpha^{\frac{1}{2}}$, while the noise component spectrum is asymptotically independent of α by the MP law. In the simplified CSBM case, this scaling relationship can also be directly verified from eq (8).

We performed random edge dropout on the commonly used Planetoid datasets (Cora, Citeseer, and Pubmed). Our results in Figure 4 reveal a clear trend of continuous decreasing ESNR as the edge dropout ratio increases across all three datasets, consistent with the observed 1 / 2 layer GNN performance. Notably, the trend is neither revealed by edge homophily metric (constant across edge dropout ratio) nor ERR (does not show continuous dropping performance). Moreover, ESNR is even able to reproduce the higher sensitivity of edge dropout ratio to GNN performance in the Cora dataset compared to Citeseer and Pubmed datasets, which

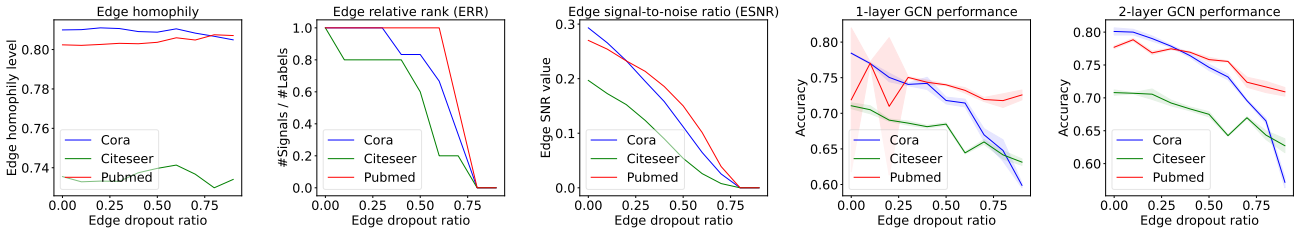

Figure 4: Comparisons of edge homophily, edge relative rank, and ESNR with 1 & 2 layer GCN performance in Planeoid datasets as functions of the edge dropout ratio $(1 - \alpha)$.

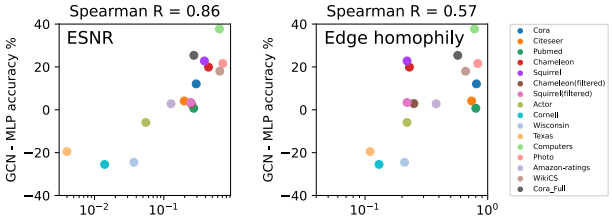

Figure 5: Comparisons of edge homophily and ESNR in terms of indicating GCN / MLP accuracy difference.

Table 1: Comparisons of homophily-based metrics and ESNR in terms of correlation with GCN / SOTA GNN minus MLP performance. EH: Edge homophily; NH: Node homophily; AH: Aggregated homophily.

	Pearson r	Pearson p-value	Spearman r	Spearman p-value
EH vs GCN	0.58	0.02	0.57	0.02
NH vs GCN	0.53	0.03	0.39	0.13
AH vs GCN	-0.06	0.83	-0.07	0.80
ESNR vs GCN	**0.85**	**0.00004**	**0.86**	**0.00002**
EH vs GNN	0.46	0.08	0.53	0.03
NH vs GNN	0.42	0.11	0.36	0.17
AH vs GNN	-0.28	0.30	-0.11	0.68
ESNR vs GNN	**0.81**	**0.0001**	**0.84**	**0.00005**

is not revealed by the other metrics (Figure 4).

For the evaluation across datasets, comparing GNN performance alone may be unfair due to feature quality difference across datasets. Therefore, we referred to previous benchmarking results of node classification accuracy difference between GCNs / state-of-the-art GNNs and multilayer perceptrons (MLPs, which does not use the graph information) in 14 datasets (Cora, Citeseer, Pubmed, Cornell, Texas, Wisconsin, Chameleon, Squirrel, Actor, Computers, Photo, Amazon-ratings, WikiCS, Cora_Full) (Ma et al., 2021; Shchur et al., 2018; Platonov et al., 2023; Dwivedi et al., 2020). For Chameleon and Squirrel datasets, we included both the raw datasets and the preprocessed datasets suggested by (Platonov et al., 2023), yielding 16 data points. For both GCN and SOTA-GNN, we observe that ESNR exhibits strikingly high concordance with the accuracy difference with MLP across all datasets, compared with alternative homophily-based metrics, including edge / node homophily and aggregated homophily (Figure 5, Table 1) (Pei et al., 2020; Luan et al., 2022). Further experimental details for both evaluations in real data can be seen in Appendix D.3 and D.4 respectively.

In summary, our evaluations in both synthetic and real data demonstrate the power of ESNR in interpreting GNN performance both within and across datasets. The code of ESNR is available at https://github.com/MingzeDong/ESNR.

4. Graph denoising via propensity score (GPS)

4.1. Graph propensity score

Next, we consider how to alleviate the negative effect of the graph structural noise on GNN training. In this work, we focus on a specific strategy - graph rewiring, which aims to revise the structure of original graph prior to GNN training for better performance, inspired from the concept of propensity score in causal inference.

Propensity score is a well addressed concept in the field of causal inference (Rosenbaum and Rubin, 1983; Imbens and Rubin, 2015). It aims to model the dependency between the treatment indicator (here we denote as z) and confounding factors (here we denote as X):

Definition 4.1. (Propensity score) *The propensity score is defined as the probability of receiving a treatment z given pre-treatment covariates X: $e(X) \equiv P(z = 1|X)$.*

In practice, the propensity score is obtained from data via prediction models such as logistic regression. As an analogy for propensity score, we propose the concept of graph propensity score, fitting the original definition into the GNN self-supervised edge prediction framework:

Definition 4.2. (Graph propensity score, GPS) *The graph propensity score is defined as the conditional probability of a link A_{ij} given a feature embedding X and an arbitrary non-overlapping link set $A_{S\setminus(i,j)}$:*

$$e_{ij}^{S}(X, A_{S\setminus(i,j)}) \equiv P(A_{ij} = 1|X, A_{S\setminus(i,j)}) \quad (9)$$

We note our defined graph propensity score can be estimated from data similar to the original propensity score. For instance, we may randomly separate the edge into S_1 and S_2, using edges in S_1 as message passing edges to predict edges in S_2 and vice versa.

Intuitively, an ideal GNN model obtained by self-supervised training is able to assign higher uncertainty level (small e_{ij}) to edges with high noise, and assign low uncertainty level (large e_{ij}) to edges with low noise. Globally, the number of selected edges should exhibit a negative correlation with the dataset's noise level, thereby resulting in a positive correlation with the ESNR. A visual illustration of the GPS graph rewiring framework is shown in Figure 6.

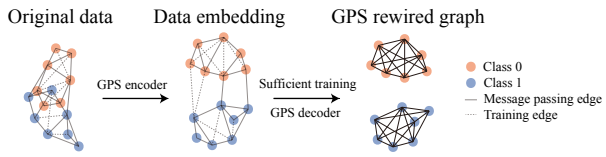

Figure 6: Illustration of the graph propensity score (GPS) graph rewiring framework. We separate the edge sets in the original data into training edges and message passing edges. We formulate a link prediction task based on the edge partition and use the trained model to generate weights for each node pair to impute edges.

4.2. Theoretical analysis of GPS in contextual stochastic block model

Here we provide preliminary theoretical analysis of Graph Convolution Network (GCN (Kipf and Welling, 2016a)) trained GPS in the simplified CSBM (Definition 3.3) in an asymptotic setting ($n \to \infty$, k fixed). We assume the data generated from the described CSBM is trained on a graph autoencoder with 1-layer GCN mean aggregation encoder and a continuous decoder. The edge set $A_{message}$ used for message passing and A_{train} for training are i.i.d. sampled from the separate edge sets in graph G. Further we use the cross entropy between the training edge set and the predicted values as the loss function used in training. Then we give the following theoretical guarantee of the rewired graph constructed from GPS, with the edges defined by

$$\hat{e}_{ij} = \text{Decoder}_{ij}(\text{Encoder}(X, A_{message})). \tag{10}$$

Theorem 4.3. (Mean edge mutual information of GPS rewired graph in simplified CSBM): *Suppose the graph propensity score $\hat{e}_{ij}$ is generated as equation (10). Then as $n \to \infty$, with probability 1 we have $\forall i, j, \hat{e}_{ij} = P_{ij}$, and*

$$\sum_{ij} \frac{I(\hat{e}_{ij}, P_{ij})}{n^2} = h(\frac{p + (k-1)q}{k}) > \sum_{ij} \frac{I(A_{ij}, P_{ij})}{n^2}.$$

The proof for theorem 4.3 can be seen at Appendix A.3.

Implications of our theory. By our theoretical analysis, we show that by minimizing the cross entropy, the graph neural network automatically estimates the probability parameters of simplified CSBM data based on both feature information and the message passing edge information. Moreover, in this setting, as each node has the same edge degree distribution and the feature attributes are noisy, the perfect estimation stated by theorem 4.3 cannot be achieved with either the feature information or the graph information alone. This result suggests the optimality of the GPS approach among self-supervised training schemes in this setting.

Is the probabilistic graph itself sufficient? However, even if we reveal the correct underlying probabilistic graph, it may still be a challenge for a GNN to learn from "ill-posed" graph-structured data due to other factors such as over-squashing (Alon and Yahav, 2020; Topping et al., 2021). In this work, our main contribution is towards overcoming the structural noise in the graph. The potential combination of the GPS approach with other graph rewiring methods remains a future direction to explore.

GPS vs DIGL. Finally, we note that DIGL (Klicpera et al., 2019) is also proposed as a graph rewiring approach for denoising through graph diffusion. However, DIGL is not feature-aware, and may fail for highly heterophilic datasets, where diffusion on the graphs blurs their structures significantly, as also discussed by (Topping et al., 2021). In contrast, GPS can work with both homophilic and heterophilic data, making it highly favorable in heterophilic data where alternative methods may fail. Additionally, combining GPS and DIGL by formulating a self-supervised graph distance prediction task may result in improved performance in homophilic data, which we propose as a potential future work.

4.3. From propensity score to graph rewiring

We may need additional transformation to turn graph propensity score (GPS) matrix into link indices usable for GNN training. Here we discuss several practical considerations.

Sparsification. As the output of link prediction task is a dense probability matrix E, we need to sparsify the matrix prior to graph neural network training. Different sparsity threshold selection can lead to dramatically different graphs. In this work, we use the k-NN thresholding scheme, preserving top k entries for each row of E to enforce uniform node connectivity. We note that other thresholding schemes can also be potentially used. For instance, we may select the optimal threshold for each row such that the expectation total number of links in the new graph is preserved.

Scalable GPS. GPS graph rewiring scheme formulates a self-supervised link prediction task. In practice, full computation of the edge likelihood requires high computational

Table 2: Node classification accuracy for different graph rewiring methods (Upper panel); ESNR of the original graphs and the GPS rewired graphs (Lower panel). *: Additional preprocessing according to (Platonov et al., 2023) was performed.

Methods	Cornell	Texas	Wisconsin	Chameleon*	Squirrel*	Actor	Cora	Citeseer	Pubmed
Baseline	43.5±3.1	66.2±3.5	52.9±3.0	39.4±1.0	42.0±0.4	28.9±0.6	**81.7±0.4**	**71.8±0.4**	**78.9±0.4**
k-NN	68.6±1.8	74.1±2.8	74.1±0.8	**48.4±1.3**	38.3±0.9	33.4±0.9	48.0±0.6	64.0±0.3	76.5±0.4
DIGL	38.9±3.0	62.7±5.0	52.9±0.0	38.0±1.7	40.4±0.8	26.9±0.4	78.0±0.3	70.6±1.0	77.2±0.2
SDRF	43.8±5.5	62.7±1.6	56.9±4.2	41.6±1.7	41.7±1.2	29.0±0.6	80.9±0.8	71.6±0.6	78.0±0.4
RS-GNN	42.4±9.8	64.8±3.6	57.3±6.1	45.3±1.2	40.4±1.1	36.1±1.6	78.1±0.3	67.2±1.3	71.1±1.9
GPS (Ours)	**74.6±3.0**	**80.0±1.8**	77.3±4.4	41.5±3.6	**43.0±0.9**	**38.3±0.7**	80.5±0.8	71.5±0.6	77.7±0.5
GPS-PE (Ours)	68.6±4.7	75.1±4.3	**78.8±1.5**	37.6±1.6	34.9±1.3	36.3±0.8	79.5±0.8	71.5±0.4	77.7±0.3
Graph ESNR	0.014	0.004	0.037	0.249	0.244	0.055	0.293	0.197	0.270

cost and is usually replaced by negative sampling, which may induce additional variance in the learned model. Therefore, apart from the vanilla GPS setting, we also consider a variant of GPS (GPS-PE) that formulates an self-supervised node regression task for predicting the graph's position encoding. After training, the graph is obtained by projecting back the full prediction into the graph space and perform the same sparsification procedure as described.

Ensemble graph rewiring. The optimization of GPS may yield high variance that leads to sub-optimal performance. In particular, when the original graph has high ESNR, it may serve as a better graph than the GPS rewired graph. As a result, in our implementation, we additively combine GPS-rewired graph adjacency matrices (E) and original graph adjacency matrices (A) to generate the final output graph:

$$E_{\text{final}} = wA + (1 - w)E \tag{11}$$

The weight w is set as a hyperparameter which can be obtained via random search. The ensemble design ensures the rewired graph to have a reasonable performance compared with the original graph in all settings.

5. Experimental results for GPS

5.1. Experimental setup

We performed extensive benchmarking of the GPS-based graph rewiring methods against a subset of alternative methods, including the GCN baseline (Kipf and Welling, 2016a), the k-NN graph as a representative of node similarity based graph rewiring, DIGL (Klicpera et al., 2019), SDRF (Topping et al., 2021), and a MLP-based link predictor proposed in RS-GNN (Dai et al., 2022). For the node classification task, we used a fixed GCN architecture across all benchmarked methods to make the results comparable. Moreover, we excluded edge dropout in the benchmarking for eliminating its potential effect on graph rewiring.

We evaluated the listed methods on nine datasets, including WebKB datasets Cornell, Texas, Wisconsin; WikipediaNetwork datasets Chameleon and Squirrel; Actor dataset and the Planetoid datasets Cora, Citeseer, and Pubmed. WebKB datasets are small datasets with significant heterophily / noise, Chameleon, Squirrel, and Actor datasets are heterophilic datasets with intermediate sizes; Planeoid datasets are homophilic datasets with intermediate sizes. Notably, for the Chameleon and Squirrel datasets, a recent work (Platonov et al., 2023) pointed out that there exists a large number of "duplicate nodes", which may harm the benchmarking validity. We followed the suggestion in the work to filter out these nodes as a preprocessing step. Statistics of the datasets used can be seen in Appendix Table 3.

For each dataset, we used the train/validation/test data split provided by pytorch_geometric. The hyperparameters were optimized by random search through RayTune, selecting the model with highest accuracy on the validation set. After selecting the hyperparameters, we tested the model performance over 10 independent runs to report the average test accuracy ± the standard deviation and the mean ESNR ± its standard deviation respectively. More experimental details and the hyperparameters can be seen in the Appendix D.5.

5.2. Node classification results

Our experimental results are summarized in Table 2. Both GPS and GPS-PE show substantial improvements over the baseline in most **heterophilic** datasets, with leading median node classification performance compared with alternative methods. However, the GCN baseline performed best in the **homophilic** datasets compared with all alternative methods, indicating that rewiring methods may not offer a distinct advantage in these cases. Taking the Cornell dataset as an example of heterophilic datasets, the superior performance of GPS rewired graphs is supported by the most distinct separation of node classes in the latent embedding (defined by the concatenation of original node features and neighborhood aggregated features), as visualized in UMAP (McInnes et al., 2018) in comparison to other approaches (Figure 7).

By additional evaluations of the ESNR metric for rewired graphs, we have found that ESNR no longer has a strong indicative role in explaining rewired graphs' GNN perfor-

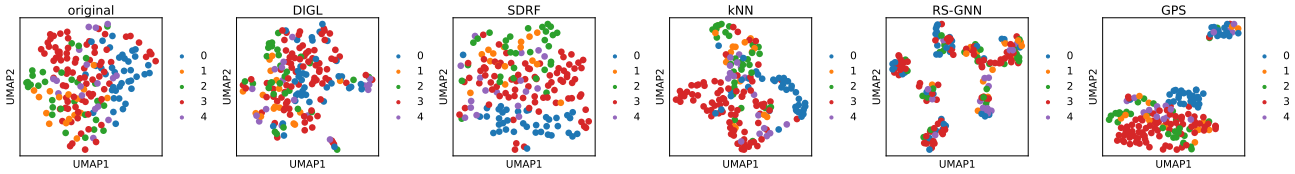

Figure 7: UMAP visualizations of the embeddings generated by each method's rewired graphs for the Cornell dataset. The embedding is defined by the concatenation of original node features and neighborhood aggregated features.

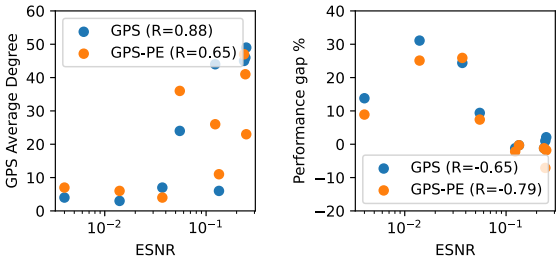

Figure 8: Scatter plots of ESNR versus average degrees of rewired graphs / performance gap. R: Spearman correlation.

mance (Appendix Table 11). For example, in the WebKB datasets and the Actor dataset, k-NN rewired graphs show the most substantial increase in ESNR, but not with comparable classification accuracy gain. This phenomenon is also observed in DIGL for the Planeoid datasets. These observations may be explained by that the graph rewiring process breaks edge independence in assumption 1 and 2. In particular, feature-aware graph rewiring methods may induce feature-level noise affecting all neighbors for each node. For DIGL, the edge noise may propagate in the rewired graphs due to the incorporation of higher-order neighbors.

To further understand the mechanism of performance improvements for the GPS based methods, we have compared the average degree and the accuracy gain of GPS rewired graphs with the ESNR of the original graph. Our comparison result reveals a positive correlation between the average degree of the rewired graph and the original graph ESNR. On the other hand, a negative correlation is observed between the performance gap and the original graph ESNR (Figure 8). The result suggests that GPS rewires graph in a global manner, influenced by the level of structural noise present in the graph. In a highly noisy dataset, GPS tends to "prune" the noisy edges, while primarily introducing new edges in datasets with high ESNR.

In summary, our proposed GPS based graph rewiring provides performance improvement for the heterophilic datasets but no improvement for homophilic datasets. The accuracy gains as well as global connectivities of GPS rewired graphs exhibit correlations with the graph ESNR. The code of GPS can be accessed at https://github.com/MingzeDong/GPS.

6. Conclusions

In this work, we aim to both understand and alleviate the effect of graph structural noise in GNN training by proposing a novel ESNR metric and a graph rewiring framework based on self-supervised GNN learning (GPS). We employ ESNR to demonstrate the significance of graph structural noise as a factor in determining GNN performances in various synthetic and real datasets. Furthermore, we show that the GPS-based graph rewiring schemes achieve superior performance in the node classification task for a number of datasets, which may be explained in terms of ESNR.

Limitations of ESNR. The relation between the ESNR and GNN with more complex architectures remains to be explored. Additionally, ESNR may work less effectively in rewired graphs which may induce redundancy or undermine the validity of graph generating assumptions 1 or 2. An improved ESNR metric that can filter the redundancy between data features and the graph remains to be explored.

Limitations of GPS. GPS is a method designed to improve GNN training by denoising the graph, therefore it may only show marginal impact on data with graphs of high SNR. Moreover, GPS is not designed for addressing the GNN training issues caused by ill-posed graph structure, therefore a combination of different methods with GPS is preferred for such data. Implementations of GPS based on more dedicated design may lead to improved rewired graphs for both heterophilic and homophilic datasets.

Broader impact. We anticipate wide use of ESNR as a universal metric for graph-structured data. Conceptually, ESNR highlights the fundamental role of graph structural noise on the GNN performance, which may enlighten future works. Further we anticipate GPS as a general framework to be adopted for rewiring noisy graph-structured data.

Acknowledgements

We thank Boris Landa and the anonymous reviewers for helpful discussions and feedback. Y.K. acknowledges support by NIH grants R01GM131642, UM1PA051410, R33DA047037, U54AG076043, U54AG079759, U01DA053628, P50CA121974, and R01GM135928.

References

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

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016a.

Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016b.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

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

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 4602–4609, 2019.

Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. *Advances in neural information processing systems*, 32, 2019.

Hoang Nt and Takanori Maehara. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550*, 2019.

Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*, 2019.

Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. *arXiv preprint arXiv:2006.05205*, 2020.

Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. *arXiv preprint arXiv:2111.14522*, 2021.

Jiong Zhu, Ryan A Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K Ahmed, and Danai Koutra. Graph neural networks with heterophily. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11168–11176, 2021.

Yujun Yan, Milad Hashemi, Kevin Swersky, Yaoqing Yang, and Danai Koutra. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. *arXiv preprint arXiv:2102.06462*, 2021.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in Neural Information Processing Systems*, 33:7793–7804, 2020.

Wendong Bi, Lun Du, Qiang Fu, Yanlin Wang, Shi Han, and Dongmei Zhang. Make heterophily graphs better fit gnn: A graph rewiring approach. *arXiv preprint arXiv:2209.08264*, 2022.

Johannes Klicpera, Stefan Weißenberger, and Stephan Günnemann. Diffusion improves graph learning. *arXiv preprint arXiv:1911.05485*, 2019.

Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 66–74, 2020.

Donghan Yu, Ruohong Zhang, Zhengbao Jiang, Yuexin Wu, and Yiming Yang. Graph-revised convolutional network. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 378–393. Springer, 2021.

Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. Learning discrete structures for graph neural networks. In *International conference on machine learning*, pages 1972–1982. PMLR, 2019.

Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. Robust graph representation learning via neural sparsification. In *International Conference on Machine Learning*, pages 11458–11468. PMLR, 2020.

Dongsheng Luo, Wei Cheng, Wenchao Yu, Bo Zong, Jingchao Ni, Haifeng Chen, and Xiang Zhang. Learning to drop: Robust graph neural network via topological denoising. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 779–787, 2021.

Adrián Arnaiz-Rodríguez, Ahmed Begga, Francisco Escolano, and Nuria Oliver. Diffwire: Inductive graph rewiring via the lov\'asz bound. *arXiv preprint arXiv:2206.07369*, 2022.

Enyan Dai, Wei Jin, Hui Liu, and Suhang Wang. Towards robust graph neural networks for noisy graphs with sparse labels. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 181–191, 2022.

Zeyu Zhang and Yulong Pei. A comparative study on robust graph neural networks to structural noises. *arXiv preprint arXiv:2112.06070*, 2021.

Yiqiao Li, Sunny Verma, Shuiqiao Yang, Jianlong Zhou, and Fang Chen. Are graph neural network explainers robust to graph noises? In *AI 2022: Advances in Artificial Intelligence: 35th Australasian Joint Conference, AI 2022, Perth, WA, Australia, December 5–8, 2022, Proceedings*, pages 161–174. Springer, 2022.

Taher Haveliwala, Sepandar Kamvar, and Glen Jeh. An analytical comparison of approaches to personalizing pagerank. Technical report, Stanford, 2003.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019.

Bahare Fatemi, Layla El Asri, and Seyed Mehran Kazemi. Slaps: Self-supervision improves structure learning for graph neural networks. *Advances in Neural Information Processing Systems*, 34:22667–22681, 2021.

Anees Kazi, Luca Cosmo, Seyed-Ahmad Ahmadi, Nassir Navab, and Michael Bronstein. Differentiable graph module (dgm) for graph convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287*, 2020.

Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. Is homophily a necessity for graph neural networks? *arXiv preprint arXiv:2106.06134*, 2021.

Jérémie Bigot, Charles Deledalle, and Delphine Féral. Generalized sure for optimal shrinkage of singular values in low-rank matrix denoising. *The Journal of Machine Learning Research*, 18(1):4991–5040, 2017.

David Hong, Yue Sheng, and Edgar Dobriban. Selecting the number of components in pca via random signflips. *arXiv preprint arXiv:2012.02985*, 2020.

Boris Landa, Thomas TCK Zhang, and Yuval Kluger. Biwhitening reveals the rank of a count matrix. *SIAM Journal on Mathematics of Data Science*, 4(4):1420–1446, 2022.

Richard Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The annals of mathematical statistics*, 35(2):876–879, 1964.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

Nicolas Keriven, Alberto Bietti, and Samuel Vaiter. Convergence and stability of graph convolutional networks on large random graphs. *Advances in Neural Information Processing Systems*, 33:21512–21523, 2020.

Nicolas Keriven, Alberto Bietti, and Samuel Vaiter. On the universality of graph neural networks on large random graphs. *Advances in Neural Information Processing Systems*, 34:6960–6971, 2021.

Luana Ruiz, Luiz Chamon, and Alejandro Ribeiro. Graphon neural networks and the transferability of graph neural networks. *Advances in Neural Information Processing Systems*, 33:1702–1712, 2020.

Rongzhe Wei, Haoteng Yin, Junteng Jia, Austin R Benson, and Pan Li. Understanding non-linearity in graph neural networks from the bayesian-inference perspective. *arXiv preprint arXiv:2207.11311*, 2022.

Derek Lim, Xiuyu Li, Felix Hohne, and Ser-Nam Lim. New benchmarks for learning on non-homophilous graphs. *arXiv preprint arXiv:2104.01404*, 2021.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of gnns under heterophily: are we really making progress? *arXiv preprint arXiv:2302.11640*, 2023.

Vijay Prakash Dwivedi, Chaitanya K Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. 2020.

Sitao Luan, Chenqing Hua, Qincheng Lu, Jiaqi Zhu, Mingde Zhao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. Revisiting heterophily for graph neural networks. *arXiv preprint arXiv:2210.07606*, 2022.

Paul R Rosenbaum and Donald B Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

Guido W Imbens and Donald B Rubin. *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press, 2015.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Lucien Le Cam. An approximation theorem for the poisson binomial distribution. *Pacific Journal of Mathematics*, 10 (4):1181–1197, 1960.

Lucien Le Cam. On the distribution of sums of independent random variables. In *Bernoulli 1713, Bayes 1763, Laplace 1813*, pages 179–202. Springer, 1965.

J Michael Steele. Le cam's inequality and poisson approximations. *The American Mathematical Monthly*, 101(1): 48–54, 1994.

Richard Sinkhorn. Diagonal equivalence to matrices with prescribed row and column sums. *The American Mathematical Monthly*, 74(4):402–405, 1967.

Boris Landa. Scaling positive random matrices: concentration and asymptotic convergence. *Electronic Communications in Probability*, 27:1–13, 2022.

Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2):cnab014, 2021.

Jie Tang, Jimeng Sun, Chi Wang, and Zi Yang. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 807–816, 2009.

Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3:127–163, 2000.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

Galileo Namata, Ben London, Lise Getoor, Bert Huang, and U Edu. Query-driven active surveying for collective classification. In *10th international workshop on mining and learning with graphs*, volume 8, page 1, 2012.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52, 2015.

Péter Mernyei and Cătălina Cangea. Wiki-cs: A wikipedia-based benchmark for graph neural networks. *arXiv preprint arXiv:2007.02901*, 2020.

Aleksandar Bojchevski and Stephan Günnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815*, 2017.

Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? *arXiv preprint arXiv:2105.14491*, 2021.

Ronald R Coifman and Stéphane Lafon. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006.

F Alexander Wolf, Philipp Angerer, and Fabian J Theis. Scanpy: large-scale single-cell gene expression data analysis. *Genome biology*, 19:1–5, 2018.

Appendix

A. Proofs of results in the main text

A.1. Proof of proposition 3.1

Proposition 3.1. *Assuming equation (1) or (2), and* $\lim_{n\to\infty}\sum_i P_{ij}^2 = 0$ *for any* j*. Then for any column one-hot matrix* $L \in R^{m\times n}$ *such that* $\forall i,j, C_{ij}^p > 0$*, we have*

$$\forall i,j,\ C_{ij} \xrightarrow{d} \text{Poisson}(C_{ij}^p). \tag{12}$$

Furthermore, in the case of eq (1) (directed graph), each entry C_{ij} *is independently sampled. In the case of eq (2) (undirected graph), each entry* C_{ij} *is asymptotically uncorrelated.*

Proof. Here we use Le Cam's theorem as a lemma:

Lemma A.1. *(Le Cam's theorem (Le Cam, 1960; 1965; Steele, 1994)) Suppose* $X_i (i = 1, 2, 3, ...)$ *are independent Bernoulli variables with parameters* $p_i (i = 1, 2, 3, ...)$*. Further suppose* $\lambda_n = \sum_{i=1}^n p_i$*, and* $S_n = \sum_{i=1}^n X_i$*. Then*

$$\sum_{i=0}^{\infty} |\Pr(S_n = k) - \frac{\lambda_n^k e^{-\lambda_n}}{k!}| < 2\sum_{i=1}^{n} p_i^2. \tag{13}$$

This can also be written in terms of total variation distance:

$$\text{TV}(S_n, \text{Poisson}(\lambda_n)) < 2\sum_{i=1}^{n} p_i^2. \tag{14}$$

Note $C_{ij} = \sum_{L_k=1} A_{kj}$ is the sum of independent Bernoulli variables A_{kj} with parameters P_{kj}. Further noting $\sum_{L_k=1} P_{kj} = C_{ij}^p$, Applying Lemma A.1, we have

$$\text{TV}(C_{ij}, \text{Poisson}(C_{ij}^p)) < 2\sum_{k:L_k=i}^{n} P_{kj}^2. \tag{15}$$

As $\lim_{n\to\infty}\sum_k P_{kj}^2 = 0$ implies $\forall i, \lim_{n\to\infty}\sum_{L_k=i} P_{kj}^2 = 0$, as $n \to \infty$, we have

$$\forall i, \lim_{n\to\infty} \text{TV}(C_{ij}, \text{Poisson}(C_{ij}^p)) = 0. \tag{16}$$

Finally, because convergence in total variation implies weak convergence, we have equation (12) holds.

Next we consider the independence of entries C_{ij}. In the case of directed graphs, we have each entry of C_{ij} sums over non-overlapping A_{kj}s due to L is row one-hot. Therefore we have the independence for all entries C_{ij} holds.

We note in the case of undirected graph, two different entries (C_{ij}, C_{kl}) at most have one shared edge. In the case of no shared edge across the two entries, the two entries are independent thus asympotically uncorrelated; in the case of one shared edge, denote the edge as a, then we have

$$\rho(C_{ij}, C_{kl}) = \frac{\mathbb{E}(C_{ij} - \mathbb{E}C_{ij})(C_{kl} - \mathbb{E}C_{kl})}{\sqrt{\text{Var}C_{ij}}\sqrt{\text{Var}C_{kl}}} = \frac{\mathbb{E}(a - \mathbb{E}a)(a - \mathbb{E}a)}{\sqrt{\text{Var}C_{ij}}\sqrt{\text{Var}C_{kl}}} = \frac{\text{Var}a}{\sqrt{\text{Var}C_{ij}}\sqrt{\text{Var}C_{kl}}} \tag{17}$$

As $n \to \infty$, because $\forall i,j, C_{ij}^p > 0$ and $\lim_{n\to\infty}\sum_i P_{ij}^2 = 0$ leading to $\forall i,j, \lim_{n\to\infty} P_{ij} = 0$, denote the probability of a as P_a, then we have

$$\lim_{n\to\infty} \rho(C_{ij}, C_{kl}) = \lim_{n\to\infty} \frac{P_a(1 - P_a)}{\sqrt{C_{ij}^p}\sqrt{C_{kl}^p}} = 0. \tag{18}$$

Hence (C_{ij}, C_{kl}) are asymptotically uncorrelated and the proof is complete. □

A.2. Proof of proposition 3.4

Proposition 3.4. (Mean edge mutual information of the original graph in simplified CSBM): *Assume the data is generated as definition 3.3. Then we have*

$$\frac{1}{n^2}\sum_{ij} I(A_{ij};P_{ij}) = h(\frac{p+(k-1)q}{k}) - \frac{1}{k}h(p) - \frac{k-1}{k}h(q).$$

Here h denotes the entropy function: $h(x) = x\log\frac{1}{x} + (1-x)\log\frac{1}{1-x}, \quad x \in (0,1)$.

Proof. We have

$$\begin{aligned}
\frac{1}{n^2}\sum_{ij} I(A_{ij};P_{ij}) &= \frac{1}{n^2}\sum_{ij} H(A_{ij}) - H(A_{ij}|P_{ij}) \\
&= \frac{1}{n^2}\sum_{i}[\sum_{j} h(\frac{p+(k-1)q}{k}) - \sum_{j} H(A_{ij}|P_{ij})] \\
&= \frac{1}{n^2}\sum_{i}[\sum_{j} h(\frac{p+(k-1)q}{k}) - \sum_{Z_j=Z_i} H(A_{ij}|P_{ij}) - \sum_{Z_j\neq Z_i} H(A_{ij}|P_{ij})] \\
&= \frac{1}{n^2}\sum_{i}[\sum_{j} h(\frac{p+(k-1)q}{k}) - \frac{n}{k}h(p) - \frac{n(k-1)}{k}h(q)] \\
&= h(\frac{p+(k-1)q}{k}) - \frac{1}{k}h(p) - \frac{k-1}{k}h(q).
\end{aligned} \tag{19}$$

□

A.3. Proof of theorem 4.3

Theorem 4.3. (Mean edge mutual information of GPS rewired graph in simplified CSBM): *Assume the graph propensity score $\hat{e}_{ij}$ is generated as equation (10). Then as $n \to \infty$, with probability 1 we have $\forall i,j, \hat{e}_{ij} = P_{ij}$, and*

$$I_{GPS} \equiv \frac{1}{n^2}\sum_{ij} I(\hat{e}_{ij},P_{ij}) = h(\frac{p+(k-1)q}{k}) > \frac{1}{n^2}\sum_{ij} I(A_{ij},P_{ij}). \tag{20}$$

Proof. Denote the edges used for training as $A_{message}$ are i.i.d. sampled from the true edge set with rate α. We define the mapping learnt by the GNN link prediction task as:

$$(Y_i,Y_j) = f_{i,j}(X,A_{message}), \quad \hat{e}_{ij} = g(Y_i,Y_j) = g\circ f_{i,j}(X,A_{message}).$$

Here we define the encoder f as GCN mean aggregation:

$$f_i(X,A_{message}) = \frac{1}{N(i)+1}\sum_{j\in N(i)\cup i} X_j.$$

Using the definition of CSBM, we have

$$f_i(X,A_{message}) = \frac{X_i + \sum_l \sum_{j:y_j=l} \mathbb{1}_{A_{ij}} X_j}{1+\sum_j A_{ij}} = \frac{X_i + \sum_l \frac{n}{k}(\frac{k}{n}\sum_{j:y_j=l}\mathbb{1}_{A_{ij}} X_j)}{1+\sum_j A_{ij}}.$$

Because $\mathbb{1}_{A_{ij}}X_j$ are i.i.d. for X_j in one label, using strong law of large numbers we have

$$(\frac{k}{n}\sum_{j:y_j=l}\mathbb{1}_{A_{ij}}X_j) \xrightarrow[n\to\infty]{a.s.} \mathbb{E}\mathbb{1}_{A_{ij}}X_j = [\mathbb{1}_{l=y_i}(p-q)+q]\alpha\mu_l.$$

Note strong law of large number on $\sum_j A_{ij}$ gives

$$\frac{k}{n}\sum_{j:y_j=l} A_{ij} \xrightarrow[n\to\infty]{a.s.} \mathbb{1}_{l=y_i}\alpha(p-q)+\alpha q.$$

Therefore with probability 1 we have

$$\lim_{n\to\infty} f_i(X, A_{message}) = \frac{\frac{k}{n\alpha}X_i + \sum_l [\mathbb{1}_{l=y_i}(p-q)+q]\mu_l}{\frac{k}{n\alpha}+\sum_l \mathbb{1}_{l=y_i}(p-q)+q} = \frac{\frac{k}{n\alpha}X_i + q\sum_{l\neq i}\mu_l + p\mu_{y_i}}{\frac{k}{n\alpha}+q(k-1)+p} = \frac{q\sum_{l\neq i}\mu_l + p\mu_{y_i}}{q(k-1)+p} \tag{21}$$

Now we define $Z_i := \frac{q\sum_{l\neq i}\mu_l + p\mu_{y_i}}{q(k-1)+p}$. As μ_l is different for each l, when $n\to\infty$, we have Z_i converge to a different constant for each possible label y_i.

Then due to the continuity of the decoder, we are able to write the decoder output as a matrix $\Gamma \in \mathbb{R}^{k\times k}$, whose entry Γ_{mn} is determined by the output of the decoder $g(Z_i, Z_j)$ given $y_i = m, y_j = n$. In this case the cross entropy loss function can be formulated as:

$$L = \frac{-1}{|\text{Trainset}|}\sum_{l_1,l_2}\Big(\sum_{y_i=l_1,y_j=l_2,(i,j)\in Trainset} A_{ij}\log\Gamma_{l_1l_2} + (1-A_{ij})\log(1-\Gamma_{l_1l_2})\Big) \tag{22}$$

Because the edges training set is also independently sampled and does not overlap with the edges used in message passing, with probability 1 above converges to

$$\frac{-1}{k^2}\sum_i p\log\Gamma_{ii} + (1-p)\log(1-\Gamma_{ii}) + \frac{-1}{k^2}\sum_{ij:i\neq j} q\log\Gamma_{ij} + (1-q)\log(1-\Gamma_{ij}).$$

Thus it is easy to verify the minimizer Γ^* of the above loss function is of value

$$\Gamma^*_{ii} = p, \quad \Gamma^*_{ij(j\neq i)} = q.$$

Finally, when the training is complete, for each possible index pair (no matter whether it is in the training set) (i, j), with probability 1 we have

$$\hat{e}_{ij} = g(Z_i, Z_j) = \Gamma_{y_iy_j} = \mathbb{1}_{y_i=y_j}(p-q)+q = P_{ij}.$$

Furthermore, in this case,

$$I_{GPS} \equiv \frac{1}{n^2}\sum_{ij} I(\hat{e}_{ij}; P_{ij}) = \frac{1}{n^2}\sum_{ij} H(P_{ij}) = h(\frac{p+(k-1)q}{k}).$$

Hence the proof is complete. □

B. Extending the biwhitening theoretical guarantee to Poisson binomial distributions

Here we aim to extend the theoretical guarantee derived for Poisson matrices in (Landa et al., 2022) to the Poisson binomial case. The proposition 3.1 cannot be directly used for the theoretical guarantee of the biwhitening scheme. The main reason is that the convergence is an asymptotic result and cannot be directly used for finite ns. However, by a more careful integration of the proposition and the proof strategy in (Landa et al., 2022), the validity of biwhitening in aggregated adjacency matrices can be revealed with mild assumptions. Specifically, proposition 3.1 reveals that $\text{Var}(C^\epsilon) \to C^p$ as $n\to\infty$. Denoting $C^{p\prime} := \text{Var}(C^\epsilon)$, we have the following proposition holds, showing the uniqueness of the scaling factors upon a multiplicative scalar for future convenience:

Proposition B.1. *Suppose the assumptions made in proposition 3.1 holds. Then $\exists N > 0$, such that $\forall n > N$, there exists a pair (u, v) of positive vectors that satisfies*

$$\forall i, \frac{1}{n}\sum_j u_i^2 C^{p\prime}_{ij} v_j^2 = 1; \quad \forall j, \frac{1}{m}\sum_i u_i^2 C^{p\prime}_{ij} v_j^2 = 1. \tag{23}$$

Furthermore, (u, v) is unique up to a positive scalar in the sense that it can only be replaced with $(au, a^{-1}v)$ for any $a > 0$.

Proof. The proof of the proposition follows immediately from theorem 1 in (Sinkhorn, 1967) as noted by (Landa et al., 2022). The only additional step needed is proving all entries of $C^{p\prime}$ is strictly positive. For this, note the assumption that $\lim_{n\to\infty}\sum_j P_{ij}^2 = 0$ for any i implies $\exists N$, such that $\forall n > N, \sum_j P_{ij}^2 < 1$. Therefore, there cannot be 1s in the adjacency matrix A for $n > N$. Combining with the assumption that $\forall n,\ C_{ij}^p > 0$, we have the variance for each entry of C^ϵ is strictly larger than 0. In this case, by taking $A = C^{p\prime}$, $r_i = n$, $c_j = m$, $x_i = u_i^2$, $y_j = v_j^2$, taking advantage of the fact that $\sum_{i=1}^m r_i = mn = \sum_{j=1}^n c_j$, we have the proposition holds by theorem 1 in (Sinkhorn, 1967). □

Now, the theorem 2.2 in (Landa et al., 2022) can be adopted and extended as the first step of the route:

Theorem B.2. *Suppose that there exist universal constants $l, L > 0$ such that $l \le \max_{i,j} C_{ij}^p \le L \min_{i,j} C_{ij}^p, \forall i, j, n$. Then as $n \to \infty$, the empirical distribution of $\tilde{\Sigma} \coloneqq n^{-1}C^\epsilon(C^\epsilon)^T$ converges a.s. to the MP distribution with parameter $\gamma = \lim_{n\to\infty}\frac{m}{n} \in (0, 1]$ and noise variance $\sigma^2 = 1$, i.e., $F_{\gamma,1}$. Furthermore, $\lambda_1\{\tilde{\Sigma}\} \xrightarrow{a.s.} \beta_+ = (1+\sqrt{\gamma})^2$., meaning the largest eigenvalue λ_1 a.s. converges to the upper edge of the MP distribution.*

Proof. From the proposition 4.1 in (Landa et al., 2022), it suffices to verify the moment of C^ϵ is bounded through the relationship for any i, j, n, k:

$$\mathbb{E}|C_{ij}^\epsilon|^{2k} \le L_k(\mathrm{Var}C_{ij})^k \tag{24}$$

To verify equation (24) indeed holds for large enough n, we note that the Poisson case is already covered in the proof of theorem 2.2 in (Landa et al., 2022)). Due to the convergence in distribution for C_{ij}^ϵ to a centered Poisson distribution, it suffices to prove that C_{ij}^ϵ converges to centered Poisson distributions in terms of arbitrary kth moment. Therefore, we only need to verify the uniform integrability of C_{ij}^ϵ. To do this, here we show

$$\forall k, \quad \mathbb{E}|C_{ij}^\epsilon|^{2k+2} < \infty \tag{25}$$

holds for finite n. This can be proven in a number of ways. Here we consider the moment generating function of uncentered C_{ij}/C_{ij}^p. Here the denominator C_{ij}^p is introduced to ensure the limit distribution exists as $n \to \infty$. Note

$$\begin{aligned} M_{C_{ij}/C_{ij}^p}(t) &= \prod_{k:L_{ik}=1}(1 - P_{kj} + P_{kj}e^{t/C_{ij}^p}) \\ &\le (1 - \frac{C_{ij}^p}{|S(k) : L_{kj} = 1|} + \frac{C_{ij}^p}{|S(k) : L_{kj} = 1|}e^{t/C_{ij}^p})^{|S(k):L_{kj}=1|}(\text{Jensen Ineq}) \end{aligned} \tag{26}$$

Due to our assumption we must have each entry P_{kj} approaches zero and $|S(k)| \to \infty$. This leads to

$$\begin{aligned} M_{C_{ij}/C_{ij}^p}(t) &\le (1 + \frac{C_{ij}^p(e^{t/C_{ij}^p} - 1)}{|S(k)|})^{|S(k)|} \\ &< e^{C_{ij}^p(e^{t/C_{ij}^p}-1)} = M_{\mathrm{Poisson}(C_{ij}^p)/C_{ij}^p}(t). \end{aligned} \tag{27}$$

Therefore the existence of arbitrary kth moment of C_{ij}/C_{ij}^p is revealed as C_{ij}/C_{ij}^p and $\mathrm{Poisson}(C_{ij}^p)/C_{ij}^p$ are both strictly positive and any order of moment for $\mathrm{Poisson}(C_{ij}^p)/C_{ij}^p$ exists. Furthermore the existence of the raw moment leads to existence of the centered moment for C_{ij}/C_{ij}^p.

Note the C_{ij}^p denominator term is cancelled in eq (24). With the convergence of moments, we immediately have

$$\begin{aligned} \mathbb{E}|C_{ij}^\epsilon|^{2k} &= (1 + o_n(1))\mathbb{E}|\mathrm{Poisson}(C_{ij}^p) - C_{ij}^p|^{2k} \\ &\le (1 + o_n(1))L_k^{\text{Poisson}}(C_{ij}^p)^k(\text{from proof of theorem 2.2 in (Landa et al., 2022)}) \\ &= (1 + o_n(1))L_k^{\text{Poisson}}(\mathrm{Var}C_{ij} + \sum_{i:L_{ij}=1} P_{ij}^2)^k \\ &= (1 + o_n(1))L_k^{\text{Poisson}}(\mathrm{Var}C_{ij} + o_n(1))^k \\ &= (1 + o_n(1))L_k^{\text{Poisson}}(\mathrm{Var}C_{ij})^k. \end{aligned} \tag{28}$$

Therefore the equation (24) holds for the model and the proof is complete. □

Next, we aim to finish the second and third step of the proof by extending lemma 2.5 in (Landa et al., 2022). Our main result here is theorem B.4, whose proof is huge simplified with lemma B.3 established in (Landa, 2022).

Lemma B.3. *Let $\tilde{A} \in \mathbb{R}^{m\times n}$ be a positive random matrix with independent entries, $A = \mathbb{E}\tilde{A}$, and (u, v) be the pair of positive factors that satisfy $||u||_1 = ||v||_1$ and scales A to row sums r and column sums c. Suppose that $\tilde{A}_{ij} \in (a_{ij}, b_{ij})$ a.s. for all i, j. Then there exists a pair of positive random vectors $(\tilde{u}, \tilde{v})$ that scales $\tilde{A}$ to row sums r and columns c s.t. for any $\delta \in (0, 1]$, with probability at least*

$$1 - 2m\exp(-\frac{\delta^2 s^2}{c_1^2||c||^2}) - 2n\exp(-\frac{\delta^2 s^2}{c_1^2||r||^2}), \tag{29}$$

we have that $\forall i, j$,

$$\frac{|\tilde{u}_i - u_i|}{u_i} \le \frac{c_2\delta s}{m\min_i r_i}, \quad \frac{|\tilde{v}_j - v_j|}{v_j} \le \frac{c_2\delta s}{n\min_j c_j}, \tag{30}$$

where $c_1 = \sqrt{2}(\frac{bd}{a^2})$, $c_2 = 1 + 2(\frac{b}{a})^{7/2}$.

Theorem B.4. *Suppose that there exist universal constants $K, k, \delta > 0$, such that $k(\log n)^{1+\delta} \le \max_{ij} C_{ij}^p \le K\min_{ij} C_{ij}^p$ for all i, j, n. Then with probability tending to 1, there is a pair of positive scaling factors $(\hat{u}, \hat{v})$ and a scalar $a_n > 0$ that satisfy*

$$\forall i, j, \ \lim_{n\to\infty} |\frac{\hat{u}_i^{(n)}}{u_i^{(n)}} - 1| = 0, \quad \lim_{n\to\infty} |\frac{\hat{v}_j^{(n)}}{v_j^{(n)}} - 1| = 0. \tag{31}$$

Proof. Here our strategy is to apply lemma B.3 for matrix C. However, lemma B.3 requires the matrix entries to be strictly positive and upper bounded, which does not hold in our case. Therefore similar with the approach in (Landa et al., 2022), we construct a truncated random matrix $\hat{C}$ with its entries both upper and lower bounded:

$$\hat{C}_{ij} = \begin{cases} C_{ij}^p/\alpha & (C_{ij} < C_{ij}^p/\alpha) \\ C_{ij} & (C_{ij} \in [C_{ij}^p/\alpha, C_{ij}^p * \beta)) \\ C_{ij}^p * \beta & (C_{ij} \ge C_{ij}^p * \beta) \end{cases} \tag{32}$$

Here $\alpha, \beta > 1$ are constants. Now with the assumption on C_{ij}^p, we are able to apply the lemma B.3 to $\hat{C}_{ij}$ as it is both upper and lower bounded. Now it suffices to verify $\hat{C} = C$ with probability 1 as $n \to \infty$.

We note the (relaxed) Chernoff inequality can be perfectly adopted here to generate exponential decay bounds for tails of C_{ij}:

Lemma B.5. *(Relaxed Chernoff inequality) Let $\{X_i\}$ be a sequence of independent Bernoulli random variables with mean $\{p_i\}$. Denote $S_n = \sum_{i=1}^n X_i$ and $\mu = \sum_{i=1}^n p_i$, then for any $\delta \in (0, 1)$, we have*

$$\mathbb{P}(S_n \ge (1+\delta)\mu) \le \exp(-\frac{\delta^2\mu}{2+\delta}), \quad \mathbb{P}(S_n \le (1-\delta)\mu) \le \exp(-\frac{\delta^2\mu}{2}). \tag{33}$$

Using the lemma on C_{ij}, we immediately have

$$\begin{aligned} \mathbb{P}(\hat{C}_{ij} \ne C_{ij}) &\le \exp(-\frac{(\beta-1)^2 C_{ij}^p}{1+\beta}) + \exp(-\frac{(\alpha-1)^2 C_{ij}^p}{2\alpha^2}) \\ &= \exp(-C_{ij}^p)^{\frac{(\beta-1)^2}{1+\beta}} + \exp(-C_{ij}^p)^{\frac{(\alpha-1)^2}{2\alpha^2}}. \end{aligned} \tag{34}$$

As a result, we have

$$\begin{aligned}\mathbb{P}(\hat{C} \neq C) &\leq \sum_{ij} \mathbb{P}(\hat{C}_{ij} \neq C_{ij}) \\ &= \sum_{ij} \exp(-\frac{(\beta-1)^2 C_{ij}^p}{1+\beta}) + \exp(-\frac{(\alpha-1)^2 C_{ij}^p}{2\alpha^2}) \\ &\leq mn \exp(-\min_{ij} C_{ij}^p)^{\frac{(\beta-1)^2}{1+\beta}} + mn \exp(-\min_{ij} C_{ij}^p)^{\frac{(\alpha-1)^2}{2\alpha^2}} \\ &\leq n^2 \exp(-\min_{ij} C_{ij}^p)^{\frac{(\beta-1)^2}{1+\beta}} + n^2 \exp(-\min_{ij} C_{ij}^p)^{\frac{(\alpha-1)^2}{2\alpha^2}} \\ &\leq n^2 \exp(-k(\log n)^{1+\delta}/K)^{\frac{(\beta-1)^2}{1+\beta}} + n^2 \exp(-k(\log n)^{1+\delta}/K)^{\frac{(\alpha-1)^2}{2\alpha^2}} \xrightarrow{n\to\infty} 0.\end{aligned} \tag{35}$$

Therefore, applying the lemma B.3, for any $\delta \in (0,1)$, with probability at least

$$1 - 2m\exp(-\frac{\delta^2 s^2}{c_1^2||c||^2}) - 2n\exp(-\frac{\delta^2 s^2}{c_1^2||r||^2}) - \mathbb{P}(\hat{C} \neq C), \tag{36}$$

we have $\forall i, j$,

$$\frac{|\tilde{u}_i(C) - u_i(C_p)|}{u_i(C_p)} \leq \frac{c_2 \delta s}{m \min_i r_i(C_p)}, \quad \frac{|\tilde{v}_j(C) - v_j(C_p)|}{v_j(C_p)} \leq \frac{c_2 \delta s}{n \min_j c_j(C_p)}. \tag{37}$$

Finally, we note here we still have a gap, as we are discussing in this theorem the scaling of the random matrix C is asymptotically identical to the scaling of deterministic C^p, but the MP law (theorem B.2) is established with the ground truth variance $\mathrm{Var}(C^\epsilon)$. Therefore, we need to verify the convergence of scaling factors of $\mathrm{Var}(C^\epsilon)$ to those of C^p. Luckily this is simple given the entry-wise convergence of $\mathrm{Var}(C^\epsilon)$ to C^p as $n \to \infty$. Specifically, for the scaling factors (u, v) of the matrix C_p, we have

$$\begin{aligned}\frac{1}{c_j}\sum_i u_i \mathrm{Var}(C^\epsilon) v_j &= \sum_i u_i (C_{ij}^p - \sum_{k:L_{ik}=1} P_{kj}^2) v_j \\ &= \frac{1}{c_j}\sum_i u_i (C_{ij}^p - \sum_{k:L_{ik}=1} P_{kj}^2) v_j \\ &= \frac{1}{c_j}\sum_i (1 - \frac{\sum_{k:L_{ik}=1} P_{kj}^2}{C_p^{ij}}) u_i C_{ij}^p v_j \\ &= \frac{1}{c_j}\sum_i (1 - o_n(1)) u_i C_{ij}^p v_j \xrightarrow[\frac{1}{c_j}\sum_i u_i C_{ij}^p v_j = 1]{n\to\infty} \sum_i u_i C_{ij}^p v_j.\end{aligned} \tag{38}$$

Similarly we have

$$\frac{1}{r_i}\sum_j u_i \mathrm{Var}(C^\epsilon) v_j \xrightarrow{n\to\infty} \frac{1}{r_i}\sum_j u_i C_{ij}^p v_j. \tag{39}$$

As a result, the lemma 9 in (Landa, 2022) can be used to establish the scaling factor consistency (stated below):

Lemma B.6. *Let $\tilde{A} \in \mathbb{R}^{m\times n}$ be a positive matrix and denote $a = \min_{ij} \tilde{A}_{ij}$, $b = \max_{ij} \tilde{A}_{ij}$, Suppose there exists $\epsilon \in (0,1)$ and positive vectors (u, v) such that*

$$\forall i, j, \ |\frac{1}{c_j}\sum_i u_i \tilde{A}_{ij} v_j - 1| \leq \epsilon; \quad |\frac{1}{r_i}\sum_j u_i \tilde{A}_{ij} v_j - 1| \leq \epsilon. \tag{40}$$

Then $\tilde{A}$ can be scaled to row sums r and column sums c by a pair $(\tilde{u}, \tilde{v})$ that for all i, j satisfies:

$$\frac{|\tilde{u}_i - u_i|}{u_i} \leq \frac{\epsilon}{1-\epsilon} + \frac{4\epsilon s\sqrt{b}}{a^2 C_1^{3/2} C_2^{3/2} m \min_i r_i} \leq \epsilon(2 + \frac{4s}{m \min_i r_i}(\frac{b}{a})^{7/2}), \tag{41}$$

$$\frac{|\tilde{v}_j - v_j|}{v_j} \leq \frac{\epsilon}{1-\epsilon} + \frac{4\epsilon s\sqrt{b}}{a^2 C_1^{3/2} C_2^{3/2} n \min_j c_j} \leq \epsilon(2 + \frac{4s}{n \min_j c_j}(\frac{b}{a})^{7/2}), \tag{42}$$

where $s = ||r||_1 = ||c||_1,\ C_1 = \min_i u_i/\bar{r}$ *and* $C_2 = \min_j v_j/\bar{c}$.

The proof of the lemma can be seen in (Landa, 2022). In our case, we have $r_i = n, c_j = m$, therefore the upper bound is $\epsilon(2+4(\frac{b}{a})^{7/2})$ for both $\frac{|\tilde{u}_i - u_i|}{u_i}$ and $\frac{|\tilde{v}_j - v_j|}{v_j}$. Note here the argument holds with probability 1 as the convergence of $\mathrm{Var}(C_{ij}^{\epsilon})$ to C_{ij}^{p} is deterministic. However, now we do not have the convergence speed as it is not specified in our assumptions. In summary, as $n \to \infty$, we have now

$$\frac{|\tilde{u}_i(Var(C^{\epsilon})) - u_i(C_p)|}{u_i(C_p)} \to 0, \quad \frac{|\tilde{v}_j(Var(C^{\epsilon})) - v_j(C_p)|}{v_j(C_p)} \to 0. \tag{43}$$

Combining eqs (37) and (43) by triangular inequality and picking consistent δ with (Landa et al., 2022), we finally have that below holds with probability tending to 1:

$$\frac{|\tilde{u}_i(C) - u_i(Var(C^{\epsilon}))|}{u_i(Var(C^{\epsilon}))} \to 0, \quad \frac{|\tilde{v}_j(C) - v_j(Var(C^{\epsilon}))|}{v_j(Var(C^{\epsilon}))} \to 0. \tag{44}$$

Thus the proof is complete.

□

C. Vanilla ESNR

C.1. Definition of vanilla entry-based SNR

In the section 3.1 of the main text, we propose the task of constructing indicators of signal to noise ratio by separating the signal component C^p and the noise component C^{ϵ} the matrix C. We note that rather than our approach (which can be deciphered as evaluating the nuclear norm of the signal component C^p), another possible approach is to construct an estimator of the Frobenius norm, yielding the objective $\frac{||C^p||_F}{||C^{\epsilon}||_F}$. We note that each entry of C approximates Poisson distribution as discussed by Proposition 3.1. In this case, a straightforward entry-wise estimation of both $||C^p||_F$ and $||C^{\epsilon}||_F$, by noting:

$$\mathbb{E}C_{ij} = \mathrm{Var}C_{ij}; \tag{45}$$

$$\mathbb{E}C_{ij}^2 = (\mathbb{E}C_{ij})^2 + \mathrm{Var}C_{ij}. \tag{46}$$

Therefore, we have

$$\begin{aligned} ||C^p||_F^2 &= \sum_{ij} C_{ij}^{p2} = \sum_{ij} (\mathbb{E}C_{ij})^2 = \sum_{ij} \mathbb{E}C_{ij}^2 - \mathbb{E}C_{ij} \sim \sum_{ij} C_{ij}^2 - C_{ij}; \\ ||C^{\epsilon}||_F^2 &= \sum_{ij} C_{ij}^{\epsilon 2} = \sum_{ij} \mathbb{E}(C_{ij} - \mathbb{E}C_{ij})^2 = \sum_{ij} \mathrm{Var}C_{ij} = \sum_{ij} \mathbb{E}C_{ij} \sim \sum_{ij} C_{ij}. \end{aligned} \tag{47}$$

Therefore an indicator of the entry-wise signal to noise ratio can be defined as:

$$\text{Vanilla ESNR} := \frac{\sum_{ij} C_{ij}^2 - C_{ij}}{\sum_{ij} C_{ij}}. \tag{48}$$

C.2. Potential issues of the vanilla entry-wise SNR

The vanilla entry-wise SNR seems to be a simpler and theoretically guaranteed estimator of the desired signal to noise ratio. However, we note several issues of the vanilla entry-wise SNR compared with our proposed ESNR:

1. The entry-wise SNR is dramatically biased by the total number of edges, which affects the magnitude of $|C_{ij}|$. Particularly, in the case of edge ratio = 1 for the CSBM data, the entry-wise SNR is still affected by the average degree.

2. The entry-wise SNR is heavily influenced by extreme values in matrix C. Notably, the issue is alleviated in our proposed ESNR as we normalize each singular value to the range $(0, 1)$.

A result of using the vanilla ESNR in simulated CSBM data of different node number / average degree settings as functions of the edge ratio (same setting as main figure 3) is shown in Appendix Figure 9. We can see it indeed exhibits significant issue of biased by the average degree, even in the setting of edge ratio = 1, where the true GCN performance is not affected by average degree in the setting. As a result, the vanilla SNR does not reveal the correct trend of either 1 or 2 layer GCN performance (Figure 3).

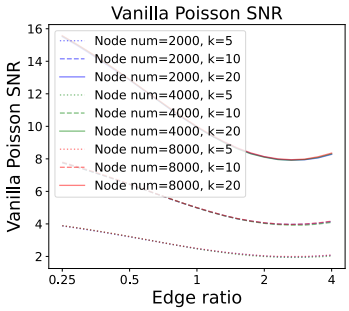

Figure 9: Vanilla entry-wise SNR in simulated CSBM data of different node number / average degree settings as functions of the edge ratio. All settings are consistent with the settings in Figure 3.

D. Experimental details

D.1. Evaluation of empirical covariance eigenvalue distributions with MP law

In this experiment, we simulated CSBM data with $n = 8000, k = 8$ with identical community sizes (1000). The average degree was set to be 10. The feature distributions for the communities were set to be Gaussian mixtures with identical noise $\sigma = 1$ with dimension 300. The expectations of the 8 communities were defined by the binary encoding of the community number, e.g.(010), (110). Each digit of the encoding was further duplicated 100 times, yielding 8 vectors of form $(\overbrace{000...0}^{100}\overbrace{111...1}^{100}\overbrace{000...0}^{100})$. We first multiplied a constant to each of the vector and set the scaled vector to be the expectation for the Gaussian mixture. In this experiment the constant is set to be 1. Finally, CSBM data were simulated in two settings of edge ratio (1.5 and 2) respectively.

For ESNR, we here implemented the data feature label based version, which is by hierarchical clustering the data feature first (via `sklearn.cluster.AgglomerativeClustering`) into $\text{round}(\sqrt{n}) = 89$ clusters. Then the Sinkhorn-Knopp based biwhitening was performed on the matrix plus `1e-9` with default parameters (`max_iter = 1000, epsilon =1e-3`). After biwhitening scaling, we subtracted the matrix with the mean value of the matrix and perform singular value decomposition. Then the singular values were squared and normalized with sample number to compute the empirical covariance eigenvalue distribution. We compare our approach with the squared normalized version of the original singular values (denoted as SVD in the figure).

D.2. Evaluation of ESNR for simulated CSBM data

Our simulation approach was consistent as described in Appendix D.1, with the only difference of setting a different gap (0.75). In this experiment, we chose different settings of n(4000,8000,16000) and different average degree (5,10,20). The edge ratio was set to be `2 ** np.arange(-2,2.2,0.2)`. We generated data for all possible combinations. Also as we needed to train GCNs on these data, we splitted each simulated data into train/val/test tests with ratio 0.6/0.2/0.2.

Our implementation of edge homophily was from `torch_geometric.utils.homophily`. Our computation of the singular values of biwhitened matrices was consistent as previously described in Appendix D.1. In practice, we added a small threshold (0.01) to α which corresponds to the right boundary of the MP distribution. After that, the ERR was computed as the signal number that passes the new boundary $\alpha + 0.01$ and the ESNR was computed according to our

definition 3.2 with parameter $\alpha + 0.01$.

For both 1 layer GCN and 2 layer GCN, we use fixed architectures:

- 1 layer GCN:

  Layer 1: `GCNConv(in_channels=data.x.shape[1], out_channels=8);`

  Activation function: `nn.ReLU();`

- 2 layer GCN:

  Layer 1: `GCNConv(in_channels=data.x.shape[1], out_channels=100);`

  Activation function: `nn.ReLU();`

  Layer 2: `GCNConv(in_channels=100, out_channels=8).`

In both cases, we used cross entropy as the loss function and trained the network by Adam algorithm. The rest hyperparameters (learning rate, weight decay) were obtained by random search through RayTune over 20 models for each simulated data. Finally we chose the model with best validation accuracy and run 10 rounds to get the mean and standard deviation of test accuracy. We ran 200 epochs for each model and decide where to stop by picking the test accuracy with maximal validation accuracy.

D.3. Evaluation of ESNR for Planetoid data with edge dropout

In this experiment, we used three Planeoid datasets (Cora, Citeseer, Pubmed), with data statistics available in Table 3. We used different masking ratio on the data edge indices: (0,0.1,0.2,...,0.9). After subsampling, we evaluate the edge homophily, ERR and ESNR as described in the previous section. Here we used the same 1/2 layer GCN architectures as previously described, except that we changed the output dimension to adapt to the class number of each dataset. The hyperparameters were obtained by random search through RayTune over 25 models for each simulated data. Finally we used the same procedure as described above to output the test accuracy. The result is shown in Figure 4.

D.4. Evaluation of ESNR across datasets for predicting GCN / GNN performance

Here we evaluated if ESNR can indicate the performance difference between GCN / SOTA-GNN and MLP by using the experimental results from (Ma et al., 2021; Shchur et al., 2018; Platonov et al., 2023; Dwivedi et al., 2020) of 14 datasets. The dataset statistics are provided in Table 3. The WebKB datasets are from `http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-11/www/wwkb`, and the references for other datasets are provided below. We compared edge homophily, node homophily (both from `torch_geometric.utils.homophily`), and aggregated homophily (from the implementation in (Luan et al., 2022)) with our ESNR on consistency with the reported accuracy difference and performed Pearson and Spearman correlation analysis. The implementation of ESNR here was same as described in Appendix D.2 and Appendix D.3.

D.5. Node classification task for GPS

Here we compared different graph rewiring methods performance on the node classification task. We included baseline (original graph), kNN graph of data features, DIGL(Klicpera et al., 2019), SDRF(Topping et al., 2021), RS-GNN (Dai et al., 2022) and our methods in the benchmarking. We compared their performance on the same nine datasets shown in Table 3.

Our implementation k-NN was based on `torch_geometric.nn.knn_graph` on first 10 truncated SVD components of data. We adopted the DIGL and SDRF implementation as provided in their original work. The architectures we used for GPS and GPS-PE are as follows:

- GPS:

  Layer 1: `GCNConv(in_channels=data.x.shape[1], out_channels=512);`

  Activation function: `nn.ReLU();`

  Layer 2: `GCNConv(in_channels=512, out_channels=128);`

Table 3: Datasets used in the work.

Datasets	Nodes	Edges	Features	Classes	Homophily score (Pei et al., 2020)
Cornell	140	219	1703	5	0.11
Texas	135	251	1703	5	0.06
Wisconsin	184	362	1703	5	0.16
Chameleon (Rozemberczki et al., 2021)	2277	36101	2325	5	0.25
Squirrel (Rozemberczki et al., 2021)	5201	217073	2089	5	0.22
Chameleon(filtered) (Platonov et al., 2023)	890	13584	2325	5	0.25
Squirrel(filtered) (Platonov et al., 2023)	2223	65718	2089	5	0.22
Actor (Tang et al., 2009)	4388	21907	931	5	0.24
Cora (McCallum et al., 2000)	2485	5069	1433	7	0.83
Citeseer (Sen et al., 2008)	2120	3679	3703	6	0.72
Pubmed (Namata et al., 2012)	19717	44324	500	3	0.79
Computer (McAuley et al., 2015)	13752	491722	767	10	0.78
Photo (McAuley et al., 2015)	7650	238162	745	8	0.83
Amazon-ratings (Platonov et al., 2023)	24498	90350	300	5	0.38
WikiCS (Mernyei and Cangea, 2020)	11701	431726	300	10	0.65
Cora_Full (Bojchevski and Günnemann, 2017)	19793	126842	8710	70	0.57

Activation function: `nn.Sigmoid();`

Decoder: `ATDecoder(128,32);`

- GPS-PE:

  Layer 1: `GCNConv(in_channels=data.x.shape[1], out_channels=512);`

  Activation function: `nn.ReLU();`

  Layer 2: `GCNConv(in_channels=512, out_channels=10);`

  Activation function: `nn.Sigmoid()`.

The ATDecoder coincides with the design of `gatv2conv` (Brody et al., 2021) as discussed in the main text. Additionally, to compute the positional encoding for GPS-PE, we used first 10 data diffusion map coordinates (Coifman and Lafon, 2006) obtained by truncated SVD on the transformed graph adjacency matrix.

To train both GPS and GPS-PE, we separated the data edge index as two sets of equal size S_1 and S_2. we trained the networks such that it minimizes the sum of two loss function (negative sampling recon_loss for GPS, MSE loss for GPS-PE) obtained by using S_1 to train S_2 and using S_2 to train S_1 respectively. After the training was finished through the Adam algorithm (500 epoches), the new graph was combined with the original graph with a learnable weight and is kNN thresholded as the rewired graph output.

Finally, for the RS-GNN, as the original work includes downstream modification of the network architecture, we have implemented a version of RS-GNN graph rewiring by modifying the GPS architecture, using a 2 layer MLP as the encoder and an inner product decoder. The MLP is obtained by substituting the GCNConv layer in GPS with MLP, with equal latent dimension sizes.

For all methods, we used cross entropy as the loss function and trained the network by Adam algorithm. The hyperparameters were obtained by random search through RayTune over 100 models (50 for the baseline graph) for each dataset. Finally we chose the model with best validation accuracy and ran 10 rounds to get the mean and standard deviation of test accuracy. We ran 200 epochs for each model and decide where to stop by picking the test accuracy with maximal validation accuracy. The ESNR was obtained by another 10 independent runs using the optimized hyperparameters.

For the UMAP visualization, we simulated rewired graphs with optimal hyperparameters for each method to generate the rewired graphs. The embedding is defined by concatenating the PCA embedding of the original features and the mean aggregated features by each rewired graph. The UMAP visualization is implemented by the function `scanpy.tl.umap` with default settings (Wolf et al., 2018; McInnes et al., 2018).

D.5.1. HYPERPARAMETER SETTING

Here lists all the hyperparameters selected by random search and used in our study. *: Additional preprocessing according to (Platonov et al., 2023) was performed.

Table 4: Optimized hyperparameters used in GCN training.

Methods	Learning rate	Weight decay
Cornell	0.00732	0.000155
Texas	0.0552	0.263
Wisconsin	0.000148	0.261
Chameleon*	0.00987	0.0176
Squirrel*	0.000260	0.000215
Actor	0.0753	0.0100
Cora	0.0511	0.0597
Citeseer	0.00364	0.0304
Pubmed	0.00613	0.00548

Table 5: Optimized hyperparameters used in kNN-GCN training.

Methods	Learning rate	Weight decay	Neighbor number
Cornell	0.000225	0.130	14
Texas	0.0799	0.00922	16
Wisconsin	0.0253	0.0889	17
Chameleon*	0.000334	0.000410	18
Squirrel*	0.000260	0.000215	17
Actor	0.000279	0.000322	7
Cora	0.000191	0.000961	3
Citeseer	0.000671	0.0485	18
Pubmed	0.0142	0.00253	5

Table 6: Optimized hyperparameters used in DIGL-GCN training.

Methods	Learning rate	Weight decay	α	ϵ
Cornell	0.000300	0.00144	0.00490	0.00527
Texas	0.0142	0.0354	0.000207	0.00192
Wisconsin	0.00213	0.343	0.0842	0.00152
Chameleon*	0.0196	0.000377	0.677	0.000169
Squirrel*	0.0206	0.0115	0.772	0.000595
Actor	0.000998	0.0141	0.350	0.00131
Cora	0.000643	0.000396	0.451	0.000258
Citeseer	0.000178	0.242	0.325	0.000982
Pubmed	0.00176	0.00125	0.830	0.000185

Table 7: Optimized hyperparameters used in SDRF-GCN training.

Methods	Learning rate	Weight decay	Loops	τ	Removal bound
Cornell	0.0825	0.0974	25	122	0.519
Texas	0.00346	0.0122	70	19	0.498
Wisconsin	0.0713	0.245	145	102	0.610
Chameleon*	0.0456	0.0136	115	40	0.539
Squirrel*	0.0173	0.00775	10	98	13.3
Actor	0.0779	0.00714	1625	177	0.153
Cora	0.000434	0.00689	140	6	0.116
Citeseer	0.000369	0.120	145	72	0.755
Pubmed	0.00178	0.00836	55	119	0.611

Table 8: Optimized hyperparameters used in RS-GCN training.

Methods	Learning rate	Weight decay	Edge learning rate	Edge weight decay	Neighbor number
Cornell	0.000296	0.000536	0.0174	0.0256	10
Texas	0.0139	0.00136	0.0176	0.00303	13
Wisconsin	0.000196	0.205	0.0215	0.00545	3
Chameleon*	0.00226	0.0270	0.0238	0.000351	26
Squirrel*	0.00358	0.0498	0.00106	0.000209	9
Actor	0.00830	0.0140	0.00899	0.00435	10
Cora	0.000270	0.0160	0.000471	0.0000114	26
Citeseer	0.000586	0.0725	0.00141	0.000314	21
Pubmed	0.00134	0.0977	0.000200	0.0000236	10

Table 9: Optimized hyperparameters used in GPS-GCN training.

Methods	Learning rate	Weight decay	Edge learning rate	Edge weight decay	Weight	Neighbor number
Cornell	0.02553	0.0398	0.0604	0.000446	0	3
Texas	0.0157	0.129	0.000336	0.00573	0	4
Wisconsin	0.00245	0.104	0.00617	0.00752	0	7
Chameleon*	0.00254	0.0332	0.0302	0.0147	0.776	49
Squirrel*	0.0696	0.0187	0.0516	0.0875	0.794	46
Actor	0.0583	0.00688	0.000952	0.00115	0	24
Cora	0.00905	0.0175	0.00435	0.0166	0.944	44
Citeseer	0.0269	0.0878	0.00272	0.00185	0.849	6
Pubmed	0.00428	0.0103	0.000248	0.0613	0.175	45

Table 10: Optimized hyperparameters used in GPSPE-GCN training.

Methods	Learning rate	Weight decay	Edge learning rate	Edge weight decay	Weight	Neighbor number
Cornell	0.00858	0.167	0.00260	0.000687	0	6
Texas	0.000864	0.0101	0.000108	0.00785	0	7
Wisconsin	0.000129	0.100	0.00593	0.00185	0	4
Chameleon*	0.00389	0.0120	0.00351	0.0195	0.761	23
Squirrel*	0.0294	0.00492	0.0294	0.0494	0.454	41
Actor	0.0179	0.00426	0.00343	0.0802	0	36
Cora	0.0562	0.00608	0.00132	0.0973	0.260	26
Citeseer	0.0230	0.0500	0.000865	0.0243	0.903	11
Pubmed	0.0215	0.0107	0.0252	0.000830	0.526	47

D.5.2. FULL ESNR TABLE

Here shows the full table of ESNR evaluation.

Table 11: ESNR for different graph rewiring methods.

Methods	Cornell	Texas	Wisconsin	Chameleon*	Squirrel*	Actor	Cora	Citeseer	Pubmed
Baseline ESNR	.014±.000	.004±.000	.037±.000	.249±.000	.244±.000	.055±.000	.293±.000	.197±.000	.270±.000
k-NN	**.268±.003**	**.318±.003**	**.326±.002**	.195±.018	.118±.004	.098±.003	.145±.003	.459±.001	.297±.000
DIGL	.017±.000	.038±.000	.100±.000	**.351±.000**	**.250±.000**	.160±.000	**.758±.000**	**.618±.000**	**.598±.000**
SDRF	.011±.002	.004±.002	.034±.005	.248±.000	.244±.000	.058±.000	.292±.001	.195±.000	.270±.000
RS-GNN	.142±.010	.180±.026	.116±.029	.311±.017	.194±.005	.139±.000	.677±.002	.603±.002	.480±.006
GPS (Ours)	.077±.026	.113±.000	.130±.000	.205±.000	.096±.000	.159±.000	.290±.015	.225±.008	.467±.006
GPS-PE (Ours)	.116±.000	.132±.000	.108±.000	.248±.000	.230±.000	**.166±.000**	.339±.000	.252±.000	.473±.000