SoLAR: Surrogate Label Aware Rewiring for Graph-Task Alignment in GNNs

Celia Rubio-Madrigal*
celia.rubio-madrigal@cispa.de

Adarsh Jamadandi *
adarsh.jamadandi@cispa.de

Rebekka Burkholz
burkholz@cispa.de

CISPA Helmholtz Center for Information Security
Stuhlsatzenhaus 5
66123 Saarbrücken

Abstract

Spectral based graph rewiring has been proposed to improve the performance of message passing graph neural networks (GNNs) by alleviating problems like over-squashing and over-smoothing. While spectral gap maximization is a common objective in rewiring, as we show, also its minimization can lead to better generalization. We find that their effectiveness is tied to the alignment between node labels and the latent community structure of the graph, as the spectral gap primarily influences the community strength but not its alignment with the node classification task. In the context of stochastic block models, we prove that this alignment indirectly impacts the probability of rewiring edges between nodes with the same or different labels, making it the most dominant factor in explaining GNN performance. To enhance label-community alignment, we propose a novel label-aware rewiring approach - `SoLAR`, that deletes inter-class and adds intra-class edges based on predicted node classes from a surrogate model. As we show through extensive experiments, the most promising surrogate models result from iterative training/rewiring cycles of GNNs, we show consistent improvements over existing baselines on a variety of datasets.

1 Introduction

Graph Neural Networks (GNNs) are a class of deep learning models that commonly adopt the paradigm of message passing [22, 53, 19, 9]. Information is repeatedly aggregated and diffused on the graph to generate a graph level representation that can be leveraged to perform either node-level [34, 26, 61, 31] or graph-level tasks [18, 46, 32]. Although GNNs have found extensive applications in a wide array of fields, including but not limited to Chemistry [49], Biology [7] and even Physics [51, 57], they are also known to have several limitations. For instance, GNNs are known to fail to distinguish simple graph structures [35, 41, 46]. Some other problems include over-squashing [1, 60, 20], where topological bottlenecks in the input graph desensitize the node features to information present in distant nodes, and over-smoothing [37, 44, 45, 65, 33], where node features tend to become indistinguishable due to repeated aggregations and/or increased model depth.

A popular approach to circumvent problems like over-squashing and over-smoothing is to make the input graph more amenable to message passing by rewiring the graph [60, 2, 21, 43, 32, 31], to

*Equal contribution.

Preprint. Under review.

obtain a better computational structure. Most proposed rewiring approaches so far focus on spectral gap maximization to mitigate bottlenecks and thus attenuate a graph's community structure. As our first contribution, we point out that also spectral gap minimization and thus an amplification of community structure can improve the performance of a GNN and provide a systematic analysis in which scenarios one is preferred over the other.

We argue that current rewiring techniques overestimate their effectiveness by not accounting for the alignment between the nodes' ground truth labels and their cluster membership labels. If this alignment is high —sometimes referred to as the *cluster hypothesis* [12], reducing the latent community structure can be detrimental to the task. If the alignment is low, amplifying a misaligned community can also be counterproductive. It is the case that spectral based rewiring either dampens or amplifies the community structure regardless of the task.

We gain these insights by a theoretical analysis of random graphs drawn from the Stochastic Block Model (SBM), a paradigm model of graphs with community structure, and a node classification task where we can control two central quantities that determine the success of GNNs: the community strength and task-community alignment. Also on real world graphs we observe that the amount of edges that are rewired in support of task-community alignment correlate with the effectiveness of different spectral rewiring approaches. Our analysis thus highlights the limits of spectral gap based rewiring, as its success depends on the task-community alignment but cannot change this critical factor, which determines the amount of edges between nodes that have different or the same label. In fact, this quantity also determines the homophily of a graph, which is known to critically influence the performance of GNNs [39].

To improve the alignment between community structure and graph labels, thus indirectly increasing the homophily score (§C), we propose a novel graph rewiring approach (`SoLAR`) that overcomes the aforementioned limitations of spectral based rewiring and also obtains a better computational structure in terms of graph-task alignment. We present a graph nature (homophilic/heterophilic) agnostic rewiring strategy that adopts a surrogate model's predictions on test nodes as proxy labels to delete inter-class edges and/or add intra-class edges (Figure 1). Our experiments confirm that the `SoLAR` framework helps obtain significant performance boost on various real-world datasets.

In summary, our main contributions are:

- Complementing the graph rewiring literature on spectral gap maximization to fight over-squashing, we highlight real-world cases in which spectral gap minimization is more supportive of solving a node classification task than spectral gap maximization.
- Our theoretical insights into an exemplary task on SBMs and experimental evidence identifies the degree of task and community structure alignment as the most critical underlying factor to explain when spectral gap rewiring improves a learning task. We present an analysis of the type of edges added or pruned by maximization or minimization and the cases when that is preferable, suggesting that this alignment underlies their success.
- To obtain a better alignment by rewiring, we propose `SoLAR` (see Fig. 1), which rewires edges with the help of a surrogate model. Our theory and experiments on various real world benchmark datasets demonstrate the effectiveness of `SoLAR`, which systematically outperforms spectral gap rewiring baselines.

2 Related work

Graph rewiring. A key component for GNNs is the input graph, since it not only acts as the data for model training but is also the computational structure on which *message passing* [19] is performed. Real world graphs, however can be noisy and sub-optimal for downstream tasks. For example, recent studies have pointed out issues like over-squashing [1, 60, 20], where presence of topological bottlenecks affect how information is diffused. This highlights the importance of the graph topology and begs the question —how can we obtain an optimal computational structure that aligns with the downstream task? Graph rewiring has emerged as a popular technique to effect changes to the edge structure. This can be done based on various criteria. For instance, [60, 21, 43] propose to use different variants of Ricci curvature [25] to rewire the graph, while [6] argue for the effective resistance [11], and [4, 17] transform the input graph into an expander graph [50] for efficient message passing.

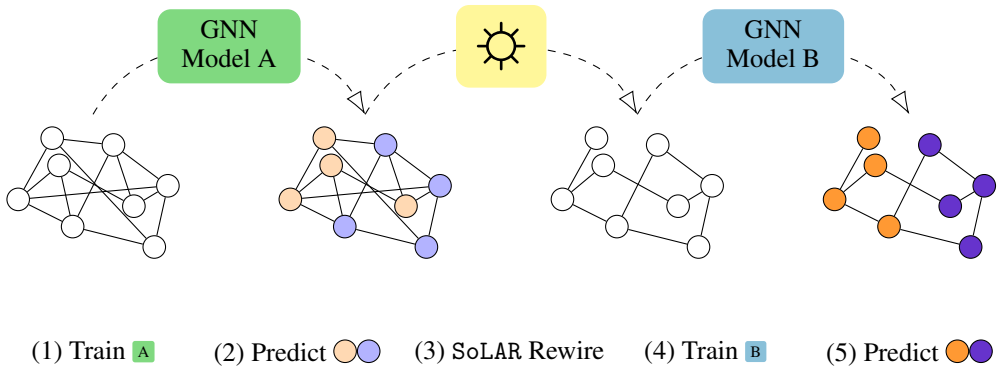


Figure 1: SoLAR: Surrogate Label Aware Rewiring. A first model A is trained on the original graph (1), and used to predict its test labels (2). The graph is then rewired (3) based on these predictions: adding same-class edges, and/or deleting different-class edges. A second model B is trained on the new graph (4). This trained model can be used to test performance (5), but also circle back to step (2) for an iterative version of SoLAR. We write **A☼B** to indicate different model combinations, used in the given order.

Spectral gap maximization. Contemporaneously, spectral based methods such as [32] aim to *maximize* the spectral gap by edge additions, as a larger spectral gap is inherently linked to faster mixing time [36] and thus better information flow. However, this can be detrimental in case of heterophilic graphs [39, 47] as we might add edges between nodes of different labels resulting in over-smoothing [37, 44, 45, 65, 33]. Spectral gap can also be maximized by deleting edges [31] and this has shown to be beneficial in slowing down detrimental over-smoothing while simultaneously mitigating over-squashing, especially in heterophilic settings. Contrarily, [2] advocate for spectral gap *minimization*, but do not explain when this is advantageous.

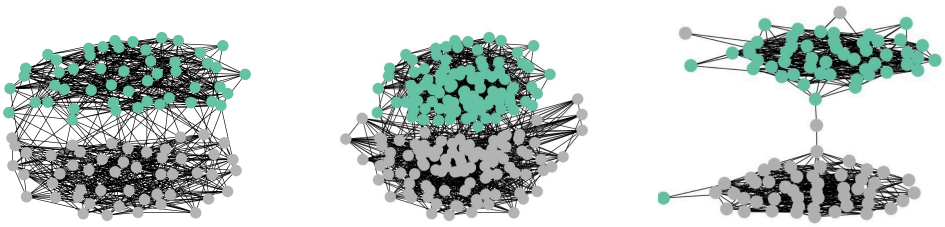
Community and task alignment. Our findings reveal that the underlying mechanism enhancing the GNN performance when rewiring actually depends on whether we modify edges connecting nodes with similar or dissimilar labels, and their influence on the latent community structure of the graph. In fact, [29] take a first step in this direction by analysing the inter-play between community and node-labels. They propose an information theoretic metric, and demonstrate its impact on performance by artificially creating and destroying communities in real-world graphs. This also highlights the importance of positive influence of same-label neighbours and how different label neighbours can be distracting for node classification [13]. We take this analysis several steps further and answer why spectral rewiring cannot induce graph-task alignment. We propose a novel proxy label based rewiring scheme which can provide us with a computational structure that aligns well with the task. This is reminiscent of [62, 28] which combines label propagation with GNNs to enhance the performance. In contrast to rewiring based on the cosine similarity of node features [23, 5] to improve homophily, we do not rely on a potentially non-robust decision what constitutes features as similar, and use more reliable information for rewiring resulting in higher performance (see Table 5 in §B).

Knowledge distillation. There are also notable differences between our strategy and approaches related to knowledge distillation and pre-training. We do not transfer knowledge encoded in the model [64, 63, 59] nor do we use context prediction/attribute masking proposed in [27] for pre-training GNNs. Note that there have also been several works that explore the application of GNNs in the context of community detection [10, 15, 55], which is not the focus of this paper.

3 Conceptual analysis

3.1 Spectral rewiring affects community strength

Spectral rewiring approaches usually focus on reducing over-squashing by maximizing the spectral gap of the input graph. However, maximizing the gap has a distinct effect on its latent community



(a) Original SBM. (b) Maximizing spectral gap. (c) Minimizing spectral gap.

Figure 2: Visualizing the effect of spectral graph maximization and minimization by deleting edges on a perfectly aligned SBM with 100 nodes.

structure. It is the case that, by maximizing the spectral gap, inter-community edges are added and intra-community edges are deleted, which attenuates the community strength (Theorem 1).

When there is a high task-community alignment —which has also been termed as the cluster hypothesis [12]— the addition of inter-community edges likely corresponds to more inter-class edges, while the removal of intra-community edges likely corresponds to less intra-class edges. Consequently, message passing happens on a less informative computational structure, rendering the rewiring detrimental to the performance of any classifier (Theorem 2). On the other hand, by minimizing the spectral gap inter-community edges are deleted and intra-community edges are added, which strengthens the community structure. If this structure is highly aligned with the labels, the rewiring is beneficial.

We illustrate the case of high task-community alignment of graphs with a paradigmatic example: the Stochastic Block Model (SBM $(p, q, \mathcal{C})$) which is a random graph model with planted communities. The nodes are partitioned into $\mathcal{C}$ communities —we adopt a binary SBM ($\mathcal{C} = 2$) unless explicitly stated otherwise. The edges are randomly sampled with probabilities p for intra-community edges and q for inter-community edges. We summarize our findings in the theorems below. Their proofs can be found in §A of the appendix.

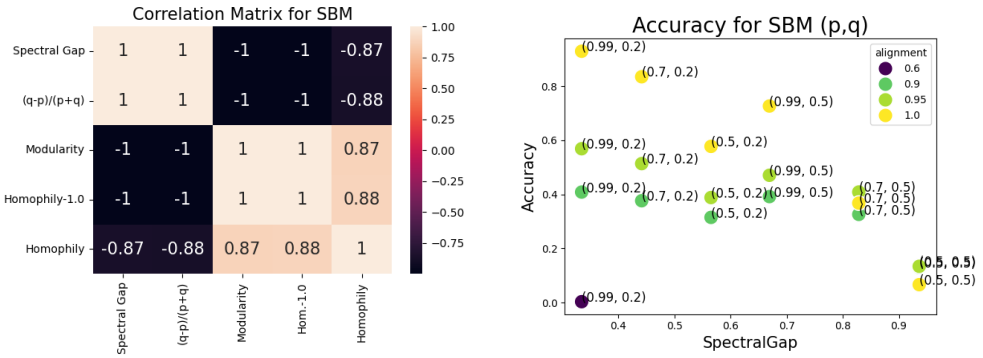
Theorem 1 (A less pronounced community structure corresponds to a higher spectral gap)**.** *Let G be a $(p$-$q)$-SBM with N nodes in 2 equally-sized communities and intra/inter-edge probabilities $p > q$. Let G^{del} be a $(p'$-$q)$-SBM where $p' < p$, and G^{add} be a $(p$-$q')$-SBM where $q' > q$. The (expected) spectral gap of G is smaller than those of G^{del} and G^{add}: $\lambda_1(G) < \lambda_1(G^{del})$, and $\lambda_1(G) < \lambda_1(G^{add})$. In fact, the spectral gap is approximately $\propto \frac{q-p}{q+p}$.*

Theorem 2 (A less pronounced community structure harms performance —if high task-community alignment)**.** *Let G be the $(p$-$q)$-SBM from Theorem 1. Let x_i be the single feature of node i, and ℓ_i its label, which corresponds to node i's community. Let f be an optimal classifier on the model's features, X, and $e(f, X)$ the (expected) proportion of misclassified nodes. After a step of sum aggregation, e is monotonically decreasing with respect to p, and increasing with respect to q.*

3.2 Varying the amount of task-community alignment

Theorem 2 applies to a Stochastic Block Model with a perfect alignment between its clusters and the task. However, in real-world graphs this assumption is not always present. The relationship between the task and the underlying community structure —which can even be missing— can take more complex forms. For instance, in heterophilic settings, same-label nodes do not need to be connected, so the effect of spectral rewiring on them is not straightforward. While spectral rewiring can influence performance by modifying how pronounced the latent community structure is, aggregation on the input graph is much more effective if we improve the mentioned alignment directly, which spectral rewiring fails to do.

This intuition is corroborated and quantified by our theory. Theorem 3 describes the behaviour of the proportion of misclassified nodes after a step of neighborhood aggregation. If we take $\psi = 1$ we obtain the same behaviour as in Theorem 2. If we take $\psi = 0$ we obtain an SBM with the labels exactly opposite their communities, so by renaming the communities we get back to a perfect alignment. If $\psi = 0.5$ then $P(M) = \Phi(0) = \frac{1}{2}$, so half the nodes are misclassified and this classifier



(a) Correlation of some scores for different values of (p, q).

(b) Accuracy of a GCN trained on different (p, q) (for 5 different seeds).

Figure 3: The effects of Theorem 1 (for spectral gap) and Theorems 2, 3 (for accuracy) on several 1000-node SBM-(p, q). Each SBM has different p and q, where $p = \{0.5, 0.7, 0.99\}$ and $q = \{0.2, 0.5\}$, and different alignment between the labels and the planted communities: $\{0.9, 0.95, 1\}$, as well as an example of 0.6 alignment which gets practically null performance. Spectral gap correlates perfectly with $\frac{q-p}{p+q}$, and negatively with the community structure and the homophily with perfect alignment. Thus, it is equivalent to plot Figure 3(b) with any of these as the x-axis.

is as good as a random choice. While, in this setup, most of the real distributions of neighbours follow binomials, we have reduced the formula with normal approximations to look at the continuous trends. All nuances are derived in the proof A.3. From it we can learn that the ψ parameter is crucial to the GNN performance.

Theorem 3 (The effect of different task-community alignments on performance)**.** *Let G be the $(p$-$q)$-SBM from Theorem 1 $(p > q)$. Let x_i be the single feature of node i where $x_i \sim \mathcal{N}(-1, 1)$ or $x_i \sim \mathcal{N}(1, 1)$ depending on its class, and ℓ_i its label, which may correspond to node i's community with a fixed probability ψ. After a step of sum aggregation, the proportion of misclassified nodes of the best classifier f is approximately*

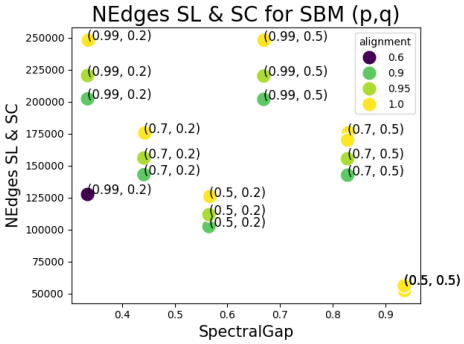
$$P(M) \approx 1 - \psi + (2\psi - 1)\Phi\left(\frac{-\frac{N}{2}(2\psi - 1)(p - q)}{\sqrt{\frac{N}{2}(p + q + p(1 - p) + q(1 - q) + 2(p - q)^2\psi(1 - \psi))}}\right)$$

3.3 Experiments on SBM for different p and q

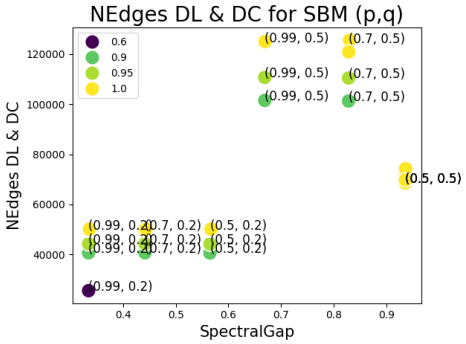
The previously stated theorems are also supported by empirical results. We create an SBM graph where the node features are generated from two normal distributions according to their class, which corresponds to the community cluster with a fixed probability (the alignment). Figure 2 visually shows what is proved in Theorem 1: that maximizing the spectral gap results in a weaker latent community structure, while minimization enhances it. But how does changing the spectral gap affect performance? For different values of p and q, we train a 2-layered GCN [34] and measure the Normalized Mutual Information (NMI) [15] between the ground truth labels and the predictions made by the GCN, which we show in Figure 3(b).

Following the results for the proof of Theorem 1, in Figure 3(a) the spectral gap correlates with $\frac{q-p}{q+p}$, and also in this case with the community strength of the SBM (negatively), as well as with the graph's normalized homophily score *when the alignment is perfect*. When it is weaker, the homophily also decreases homogeneously. In Figure 3(b) we compare the spectral gap of these different SBM graphs against the accuracy of a GCN trained on it, using a fixed train-test split.

In cases of high homophily and high alignment, it is beneficial to minimize the spectral gap, as the communities that get strengthened also correspond to the task labels. However, the spectral gap does not completely correlate with the GCN accuracy, as it can only affect the community strength. We



(a) Number of edges that connect nodes with the Same Labels (SL) and the Same Community (SC).



(b) Number of edges that connect nodes with Different Labels (DL) in Different Communities (DC).

Figure 4: The total number of edges that connect nodes with the same or opposite labels in an SBM-(p, q) setup for different values of p and q. The number of SLSC edges correlates with the accuracy in some cases where the *proportion* of these edges (homophily) does not. The communities are calculated with a modularity maximization algorithm, so they correspond to the latent structures and not the generated ones.

can also see that the lack of task-graph alignment very quickly dampens the GNN perfomance, as shown by the different hues in the scatter plot. Changing the alignment only from 1.0 to 0.95 reduces dramatically the influence of different (p, q) on the performance.

But even within the same theoretical alignment, the topology of the graph has more nuance on the accuracy values. For instance, the SBM-$(0.5,\ 0.2)$ has a lower spectral gap (and higher homophily) than the SBM-$(0.99,\ 0.5)$, although a worse test performance. This might be explained by a higher amount of same-label edges —in spite of having a lower proportion of them over all edges, as seen in Figure 4. More same-label edges better support the task, so it is natural to propose a rewiring approach that aims to maximize this quantity, especially if these edges stay within the latent communities. In the same spirit we can try to minimize the number of opposite-label edges too. Both approaches lead to our framework `SoLAR` (§4).

3.4 Analysis on real-world datasets

When training GNNs on real world datasets, the trends in accuracy scores become more complex. It is not straightforward to know when minimization or maximization works the best nor do we get an insight into how many edge modifications are required to see a change in the GNN performance. On one hand, very homophilic datasets might be similar to the SBM setup described in the previous theorems, so spectral maximization is detrimental in the long run —as seen for Cora and Citeseer in Figure 5, where the alignment between labels and communities gets heavily reduced, and so does the accuracy. On the other hand, improving the connectivity might be key for some tasks, where, for example, information needs to travel across different clusters. All different kinds of spectral rewiring methods are usually effective for small number of edge changes, as they might locally have a denoising effect for some edges.

However, the trend variability for spectral rewiring might be explained by the type of edges it adds or deletes, in terms of both the labels and the communities they connect. We show in Figure 6 the number of edges that connect nodes with the same or different labels and communities, for spectral minimization, maximization, and random rewiring of 500 edges, for both Cora and Chameleon. We use the spectral gap optimization algorithms presented in [31], as they are computationally cheaper to approximate. We can easily substitute this with any method that performs spectral gap optimization like FoSR [32]. The amount of edges for each type clearly changes from the homophilic to the heterophilic case for the different methods.

In the first row (MinGap), we see that minimization adds same-community edges more than the other two methods. When adding edges in homophilic settings (Cora) this is preferred, because these same-community edges are mostly same-label (Same C: 152/21). However, in heterophilic

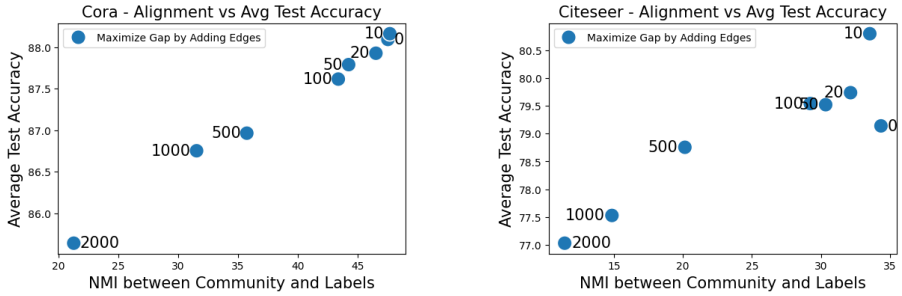
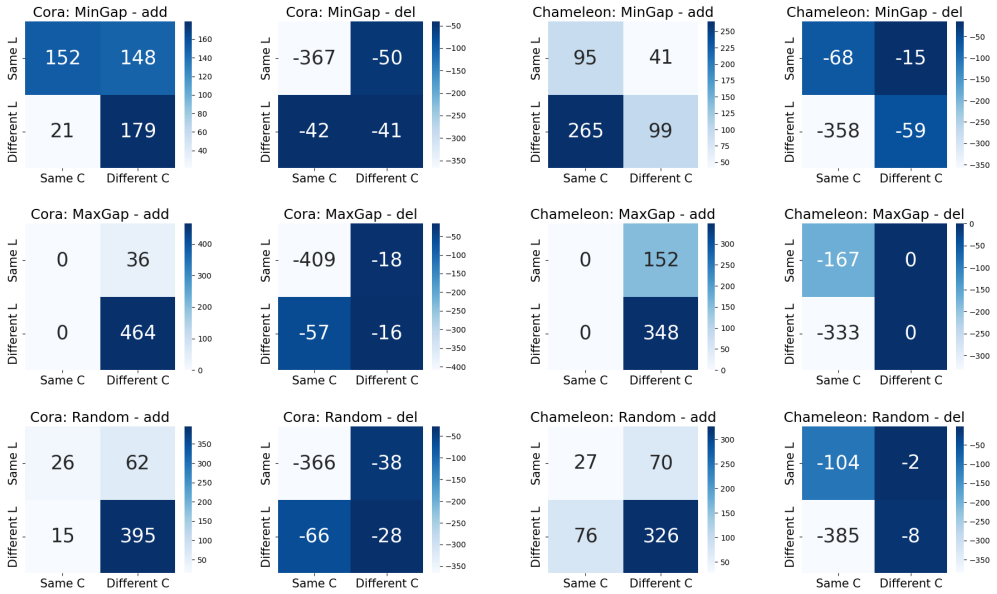


Figure 5: Maximizing the spectral gap (using [31]) on Cora and Citeseer reduces both the task-graph alignment and the test accuracy.



(a) Cora: additions vs. deletions

(b) Chameleon: additions vs. deletions

Figure 6: Alignment matrices for Cora (homophilic) and Chameleon (heterophilic) by a 500-edge rewiring method. In each row: spectral minimization and maximization from [31], and random rewiring. In each column: additions and deletions. Each alignment matrix compares the number of edges added/deleted in terms of the type of nodes it connects: with the Same or Different L(abel), and with the Same or Different C(ommunity).

settings (Chameleon) the opposite is true: making the community structure more pronounced adds edges connecting different labels (Same C: 95/265). Deletions are, however, more similar to random rewiring, with the exception of a subtle increase in the pruning of different-community edges for the heterophilic setting, compared to random (Different C: -15/-59).

In the second row, MaxGap exclusively adds different-community edges. In homophilic settings this is detrimental, as most of them will be from different classes (Different C: 36/464). However, in heterophilic settings it might connect nodes of the same class, which helps align the community structure with the task (Different C: 152/348). MaxGap also prunes almost exclusively same-community edges, which is again detrimental for the homophilic case (Same C: -409/-57) and more helping in the heterophilic case (Same C: -167/333). The fact that spectral pruning maximization helps especially in heterophilic settings is supported by the experimental evidence in [31].

The alignment matrices serve as a guiding principle to determine if spectral gap maximization or minimization should be preferred. However, spectral gap optimization fails to transform the input graph into a computational structure that is well aligned for the downstream task. As a resolution, we

advocate for a rewiring strategy that aims to maximize the amount of same-label and minimize the amount of different-label edges directly.

4 SoLAR: Surrogate Label Aware Rewiring

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected and unweighted graph with $|\mathcal{V}|$ nodes and $|\mathcal{E}|$ edges. The adjacency matrix $A \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ encodes the graph topology. We use a 2-layered GCN [34, 14] which operates on the degree normalized adjacency matrix $\hat{A} = \tilde{D}^{-1/2}(A + I)\tilde{D}^{-1/2}$ and X is the associated node features. The task is to perform node classification in a transductive setting. That is, given a set of nodes $\mathcal{V}_{train}$ whose labels $\mathcal{Y}_{train}$ are available, we are required to predict the labels of nodes $\mathcal{V}_{test} = \mathcal{V} \backslash \mathcal{V}_{train}$. Let $\mathbf{Z}$ be the predictions made by the GCN

$$\mathbf{Z} = \mathit{softmax}(\hat{A}\sigma(\hat{A}X\mathbf{\Theta^{(0)}})\mathbf{\Theta^{(1)}} \tag{1}$$

where $\sigma(\cdot)$ is a non-linear activation function such as ReLU and $\mathbf{\Theta}$ the weight matrix. We propose a simple yet highly effective strategy that uses predictions made by a surrogate GNN model as proxy labels for graph rewiring —since we do not have access to test node labels in a semi-supervised setting. The process works in two stages. In the first stage, we instantiate a surrogate GNN $Z_1 = f_{surrogate}(\mathcal{G}, \mathbf{\Theta})$ (such as in Equation (1)), and train it to convergence to obtain a set of predicted labels. We then use the predicted labels, $Z_1 = \mathcal{Y}_{proxy}$, to rewire the graph by either deleting inter-class edges and/or adding intra-class edges to obtain a rewired graph $\hat{\mathcal{G}} = (\mathcal{V}, \hat{\mathcal{E}})$ —we use the predictions on the test set, as we already have access to the ground truth labels for the nodes in the train set. In the second stage, we instantiate a second 'training' GNN —$f_{train}(\hat{\mathcal{G}}, \mathbf{\Theta})$, which operates on the new computational structure. Note that $f(\mathcal{G}, \mathbf{\Theta})$ can be any model which is expressive enough. The process is illustrated in Figure 1.

Apart from GCN models, in Section §5 we experiment with GATv2 [8] to ablate the effect of model expressivity on the quality of the proxy labels. The above outlined process can be either done in a one-shot way, where the pre-requisite number of edges are deleted/added at once to obtain the modified graph or can be done iteratively, that is, train-rewire-train (cf. §C.1). We present results on node classification using both the methods.

5 Experiments

We perform node classification on the following homophilic datasets: Cora [40], Citeseer [54] and Pubmed [42], Co-author CS, Physics and Amazon Photos [56]. We also report results on heterophilic graphs Chameleon, Squirrel, Actor and the WebKB datasets consisting of Cornell, Wisconsin and Texas [48]. Additionally, we include three large heterophilic graphs: Roman-empire and Amazon-ratings introduced in [48], and Penn94 [38]. Our backbone models are GCN [34] and GATv2 [8]. We present results for both one-shot SoLAR and Iterative SoLAR (in Section §C.1).

As we have access to the ground truth labels for the train set and the proxy labels for test set, we already have a good estimation of how many inter-class/intra-class edges are present in a graph. This allows us to make quick decisions about the number of edges to modify, without having to tune it like in other rewiring approaches [60, 32, 43, 21, 31]. For instance, Cora allows for deleting roughly 1700 inter-class edges without disconnecting the graph, so with this edge modification budget as our constraint we can choose to have as many train-rewire-train iterations.

We adopt 60/20/20 splits for training, validation and testing respectively. The final test accuracy is reported averaged over 100 splits of the data. See Section §D for training details and hyperparameters used. We test two GNN models to see the effect of quality labels from an expressive model has on the generalization performance. The top performance is highlighted in bold. We compare GCN and GATv2 operating on the original graph with FoSR [32], which approximates the spectral gap change using a first order approximation and adds edges that maximize this proxy, PROXYDELMIN [31] deletes edges that minimize the spectral gap, and our proposed rewiring method: GCN☼GCN and GATv2☼GATv2, where the first model gives the proxy labels for rewiring the graph and the second model uses the rewired graph for training on the downstream task. Note that we use the ground truth labels for the train nodes and the surrogate labels for the test nodes only. We report results for both predicted-inter-class edge deletions and predicted-intra-class edge additions. When deleting

Table 1: Node classification using one-shot SoLAR on large heterophilic graphs.

Method	Roman-Empire	Amazon-Ratings	Penn94
GCN	77.74±0.60	47.66±0.54	82.29±0.77
GATv2	82.52±0.50	47.66±0.95	81.85±3.02
GCN+FoSR	73.60±1.11	49.68±0.73	69.73±7.83
GATv2+FoSR	81.88±1.07	51.36±0.62	72.56±5.55
GCN☼GCN+Delete	80.90±0.14	50.30±0.09	83.59±1.40
GCN☼GCN+Add	81.13±0.21	49.86±0.11	**83.65±1.69**
GATv2☼GATv2+Delete	**84.32±0.80**	**52.06±0.00**	83.58±1.60
GATv2☼GATv2+Add	**84.27±0.40**	**52.08±0.09**	**83.60±1.32**

inter-class edges we ensure we do not disconnect the graph and leave few edges to preserve the original structural integrity of the graph.

In Table 2 we present the results for the homophilic graphs, and in Table 1 and Table 3 we present results for the heterophilic graphs. Evidently, our proposed rewiring boosts the GNN performance across all datasets tested. Especially on large heterophilic datasets like Roman-empire and Amazon-ratings, we can see the combination of GATv2☼GATv2 giving the best performance. This highlights the importance of the expressiveness of the surrogate model which endows the proxy labels: a more powerful model like GATv2 will have stronger effect on the quality of the labels.

Table 2: Node classification on homophilic graphs using one-shot SoLAR.

Method	Cora	Citeseer	Pubmed	CS	Physics	Photo
GCN	87.94±3.35	79.38±3.48	81.99±1.42	92.44±0.67	93.64±0.16	92.89±1.23
GATv2	89.13±3.13	81.92±4.81	81.83±1.04	91.90±1.59	94.07±0.44	91.22±2.18
GCN+FoSR	88.74±2.70	79.48±3.77	82.22±1.24	93.54±0.80	94.72±0.21	90.57±3.82
GATv2+FoSR	89.72±2.91	81.75±4.86	81.29±2.31	92.35±1.21	93.96±0.40	90.48±2.57
GCN+Proxydelmin	89.35±2.92	79.56±2.96	82.89±1.53	93.66±0.83	94.61±0.27	92.46±2.16
GATv2+Proxydelmin	89.61±3.00	81.57±3.56	81.59±1.43	93.31±1.21	94.43±0.33	93.86±1.61
GCN☼GCN+Delete	**90.17±2.82**	82.22±4.01	82.61±1.16	93.00±0.21	93.96±0.10	93.75±0.99
GCN☼GCN+Add	**90.06±2.56**	**83.26±4.44**	**83.05±2.50**	92.46±0.56	**95.47±0.31**	92.13±0.32
GATv2☼GATv2+Delete	**90.06±3.31**	**83.01±4.32**	82.41±2.46	**94.16±1.79**	**95.01±0.54**	**93.78±1.30**
GATv2☼GATv2+Add	89.63±3.16	81.78±4.44	81.32±1.66	92.79±1.58	94.25±0.46	**93.36±1.93**

Table 3: Node classification on heterophilic graphs using one-shot SoLAR.

Method	Cornell	Texas	Wisconsin	Chameleon	Squirrel	Actor
GCN	68.31±8.13	73.47±10.13	66.14±9.23	54.64±6.94	43.25±6.32	28.26±3.22
GATv2	86.84±9.78	89.01±10.43	87.56±9.20	61.79±10.20	45.71±5.12	29.41±2.98
GCN+FoSR	71.64±9.80	73.93±10.23	65.85±7.73	54.40±6.58	42.80±6.40	28.66±3.21
GATv2+FoSR	76.12±6.51	78.15±7.81	74.08±9.01	46.48 ± 4.97	47.40±7.17	27.45±3.61
GCN+Proxydelmin	81.94±7.96	83.46±10.90	70.63±7.68	53.64±6.00	41.26±5.35	28.58±2.93
GATv2+Proxydelmin	85.40±7.64	83.44±9.52	79.78±11.26	66.15±12.01	45.02±5.13	**32.37±4.36**
GCN☼GCN+Delete	68.35±8.54	74.12±9.89	67.85±7.14	57.19 ± 6.45	44.50±6.29	29.25±3.50
GCN☼GCN+Add	69.42±8.93	74.20±10.26	68.51±7.20	56.43 ± 6.16	44.04±6.34	28.16±3.22
GATv2☼GATv2+Delete	**87.40±9.89**	**90.14±10.64**	**88.32±9.08**	**68.89±11.50**	**49.10±5.59**	30.31±4.29
GATv2☼GATv2+Add	87.12±9.59	87.97±10.95	87.76±9.57	66.35±11.18	46.44±6.00	29.46±4.67

6 Discussion

It is clear from our analyses (§3) and supporting experiments (§5) that our rewiring strategy helps achieve better GNN performance, as it transforms the input graph to a computational structure which is better aligned with the learning task. Our approach works seamlessly for both homophilic and heterophilic settings— contrary to methods which use non-robust feature similarity measures [28, 5] or require expensive k-hop rewiring during training [24] to be effective. Our approach is also more powerful in that it is capable of directly influencing measures (cf. §C) that are critical for GNN performance, as opposed to methods that rely purely on the topological characteristics of the input

graph [60, 32, 43, 21, 31] like the spectral gap. However, it is important to note that the quality of rewiring largely depends on the surrogate model's ability to provide accurate labels. If the predicted labels are too noisy, they will also not be very informative for the rewiring, and may even amplify issues that were already present in the initial model. In this case, it could be interesting to take features into account, combining this work with [23, 5]. Furthermore, this rewiring does not explicitly address other problems that arise in GNNs, such as over-squashing and over-smoothing; the impact of these issues may still persist during training. Another approach for a more complex rewiring strategy could be to combine spectral and label-aware rewiring in order to preserve sufficient connectivity.

References

[1] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In *International Conference on Learning Representations*, 2021.

[2] Adrián Arnaiz-Rodríguez, Ahmed Begga, Francisco Escolano, and Nuria Oliver. Diffwire: Inductive graph rewiring via the lovász bound, 2022.

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

[4] Pradeep Kr. Banerjee, Kedar Karhadkar, Yu Guang Wang, Uri Alon, and Guido Montúfar. Oversquashing in gnns through the lens of information contraction and graph expansion. In *2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, page 1–8. IEEE Press, 2022.

[5] Wendong Bi, Lun Du, Qiang Fu, Yanlin Wang, Shi Han, and Dongmei Zhang. Make heterophily graphs better fit gnn: A graph rewiring approach, 2022.

[6] Mitchell Black, Zhengchao Wan, Amir Nayyeri, and Yusu Wang. Understanding oversquashing in gnns through the lens of effective resistance. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[7] Pietro Bongini, Niccolò Pancino, Franco Scarselli, and Monica Bianchini. Biognn: How graph neural networks can solve biological problems. In *Artificial Intelligence and Machine Learning for Healthcare*, pages 211–231. Springer, 2023.

[8] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations*, 2022.

[9] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Velickovic. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *CoRR*, abs/2104.13478, 2021.

[10] Joan Bruna and X Li. Community detection with graph neural networks. *stat*, 1050:27, 2017.

[11] Ashok K. Chandra, Prabhakar Raghavan, Walter L. Ruzzo, Roman Smolensky, and Prasoon Tiwari. The electrical resistance of a graph captures its commute and cover times. *computational complexity*, 6(4):312–340, 1996.

[12] O. Chapelle, B. Scholkopf, and A. Zien, Eds. Semi-supervised learning (chapelle, o. et al., eds.; 2006) [book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

[13] Hao Chen, Yue Xu, Feiran Huang, Zengde Deng, Wenbing Huang, Senzhang Wang, Peng He, and Zhoujun Li. Label-aware graph convolutional networks. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 1977–1980, New York, NY, USA, 2020. Association for Computing Machinery.

[14] Tianlong Chen, Yongduo Sui, Xuxi Chen, Aston Zhang, and Zhangyang Wang. A unified lottery ticket hypothesis for graph neural networks. In *International Conference on Machine Learning*, 2021.

[15] Zhengdao Chen, Lisha Li, and Joan Bruna. Supervised community detection with line graph neural networks. In *International Conference on Learning Representations*, 2019.

[16] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Phys. Rev. E*, 70:066111, Dec 2004.

[17] Andreea Deac, Marc Lackenby, and Petar Veličković. Expander graph propagation. In *The First Learning on Graphs Conference*, 2022.

[18] Federico Errica, Marco Podda, Davide Bacciu, and Alessio Micheli. A fair comparison of graph neural networks for graph classification. In *International Conference on Learning Representations*, 2020.

[19] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1263–1272. JMLR.org, 2017.

[20] Francesco Di Giovanni, Lorenzo Giusti, Federico Barbero, Giulia Luise, Pietro Lio', and Michael Bronstein. On over-squashing in message passing neural networks: The impact of width, depth, and topology, 2023.

[21] Jhony H. Giraldo, Konstantinos Skianis, Thierry Bouwmans, and Fragkiskos D. Malliaros. On the trade-off between over-smoothing and over-squashing in deep graph neural networks. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, CIKM '23, page 566–576, New York, NY, USA, 2023. Association for Computing Machinery.

[22] M. Gori, G. Monfardini, and F. Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734 vol. 2, 2005.

[23] Jiayan Guo, Lun Du, Wendong Bi, Qiang Fu, Xiaojun Ma, Xu Chen, Shi Han, Dongmei Zhang, and Yan Zhang. Homophily-oriented heterogeneous graph rewiring. In *Proceedings of the ACM Web Conference 2023*, WWW '23, page 511–522, New York, NY, USA, 2023. Association for Computing Machinery.

[24] Benjamin Gutteridge, Xiaowen Dong, Michael M Bronstein, and Francesco Di Giovanni. DRew: Dynamically rewired message passing with delay. In *International Conference on Machine Learning*, pages 12252–12267. PMLR, 2023.

[25] Richard Hamilton. The ricci flow on surfaces. In *Mathematics and general relativity, Proceedings of the AMS-IMS-SIAM Joint Summer Research Conference in the Mathematical Sciences on Mathematics in General Relativity*, 1988.

[26] William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive Representation Learning on Large Graphs. In *NIPS*, pages 1024–1034, 2017.

[27] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *International Conference on Learning Representations*, 2020.

[28] Qian Huang, Horace He, Abhay Singh, Ser-Nam Lim, and Austin R. Benson. Combining label propagation and simple models out-performs graph neural networks. *CoRR*, abs/2010.13993, 2020.

[29] Hussain Hussain, Tomislav Duricic, Elisabeth Lex, Denis Helic, and Roman Kern. The interplay between communities and homophily in semi-supervised classification using graph neural networks. *Applied Network Science*, 6(1):80, 2021.

[30] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[31] Adarsh Jamadandi, Celia Rubio-Madrigal, and Rebekka Burkholz. Spectral graph pruning against over-squashing and over-smoothing, 2024.

[32] Kedar Karhadkar, Pradeep Kr. Banerjee, and Guido Montufar. FoSR: First-order spectral rewiring for addressing oversquashing in GNNs. In *The Eleventh International Conference on Learning Representations*, 2023.

[33] Nicolas Keriven. Not too little, not too much: a theoretical analysis of graph (over)smoothing. In *The First Learning on Graphs Conference*, 2022.

[34] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*, 2017.

[35] Adrien Leman. The reduction of a graph to canonical form and the algebra which appears therein. 1968.

[36] David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, 2006.

[37] Guohao Li, Matthias Müller, Ali Thabet, and Bernard Ghanem. Deepgcns: Can gcns go as deep as cnns? In *The IEEE International Conference on Computer Vision (ICCV)*, 2019.

[38] Derek Lim, Felix Matthew Hohne, Xiuyu Li, Sijia Linda Huang, Vaishnavi Gupta, Omkar Prasad Bhalerao, and Ser-Nam Lim. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[39] Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. Is homophily a necessity for graph neural networks? In *International Conference on Learning Representations*, 2022.

[40] Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

[41] 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. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):4602–4609, Jul. 2019.

[42] Galileo Namata, Ben London, Lise Getoor, and Bert Huang. Query-driven active surveying for collective classification. 2012.

[43] Khang Nguyen, Hieu Nong, Vinh Nguyen, Nhat Ho, Stanley Osher, and Tan Nguyen. Revisiting over-smoothing and over-squashing using ollivier-ricci curvature, 2023.

[44] Hoang NT and Takanori Maehara. Revisiting graph neural networks: All we have is low-pass filters. *ArXiv*, abs/1905.09550, 2019.

[45] Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2020.

[46] Pál András Papp, Karolis Martinkus, Lukas Faber, and Roger Wattenhofer. DropGNN: Random dropouts increase the expressiveness of graph neural networks. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[47] Oleg Platonov, Denis Kuznedelev, Artem Babenko, and Liudmila Prokhorenkova. Characterizing graph datasets for node classification: Homophily-heterophily dichotomy and beyond. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[48] Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at evaluation of gnns under heterophily: Are we really making progress? In *The Eleventh International Conference on Learning Representations*, 2023.

[49] Patrick Reiser, Marlen Neubert, André Eberhard, Luca Torresi, Chen Zhou, Chen Shao, Houssam Metni, Clint van Hoesel, Henrik Schopmans, Timo Sommer, and Pascal Friederich. Graph neural networks for materials science and chemistry. *Communications Materials*, 3(1):93, 2022.

[50] Justin Salez. Sparse expanders have negative curvature, 2021.

[51] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8459–8468. PMLR, 13–18 Jul 2020.

[52] Ryoma Sato. Training-free graph neural networks and the power of labels as features, 2024.

[53] 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, 2009.

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

[55] Oleksandr Shchur and Stephan Günnemann. Overlapping community detection with graph neural networks. *Deep Learning on Graphs Workshop, KDD*, 2019.

[56] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation, 2019.

[57] Jonathan Shlomi, Peter Battaglia, and Jean-Roch Vlimant. Graph neural networks in particle physics. *Machine Learning: Science and Technology*, 2(2):021001, jan 2021.

[58] John R. Silvester. Determinants of block matrices. *Mathematical Gazette*, 84(501):460–467, November 2000.

[59] Yijun Tian, Shichao Pei, Xiangliang Zhang, Chuxu Zhang, and Nitesh V. Chawla. Knowledge distillation on graphs: A survey, 2023.

[60] Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M. Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations*, 2022.

[61] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. In *ICLR*, 2018.

[62] Hongwei Wang and Jure Leskovec. Unifying graph convolutional neural networks and label propagation, 2020.

[63] Cheng Yang, Yuxin Guo, Yao Xu, Chuan Shi, Jiawei Liu, Chunchen Wang, Xin Li, Ning Guo, and Hongzhi Yin. Learning to distill graph neural networks. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, WSDM '23, page 123–131, New York, NY, USA, 2023. Association for Computing Machinery.

[64] Yiding Yang, Jiayan Qiu, Mingli Song, Dacheng Tao, and Xinchao Wang. Distilling knowledge from graph convolutional networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7072–7081, 2020.

[65] Kaixiong Zhou, Xiao Huang, Daochen Zha, Rui Chen, Li Li, Soo-Hyun Choi, and Xia Hu. Dirichlet energy constrained learning for deep graph neural networks. *Advances in neural information processing systems*, 2021.

Appendix

A Proofs

A.1 Proof of Theorem 1

Proof. We consider an SBM with 2 classes and $\frac{N}{2}$ nodes in each class, with intra-class edge probability p and inter-class edge probability q. Its adjacency matrix A is a random matrix where $A_{ij} = \text{Bernoulli}(p)$ if nodes i, j are in the same cluster, and $A_{ij} = \text{Bernoulli}(q)$ otherwise. For a large N, the adjacency matrix A can be approximated by its expected value, which is a block matrix: $\tilde{A} = \begin{pmatrix} P & Q \\ Q & P \end{pmatrix}$, where $P = p \cdot \mathbb{1}_{\frac{N}{2}} + (1-p) I_{\frac{N}{2}}$, where all values are p except the diagonal with ones, and $Q = q \cdot \mathbb{1}_{\frac{N}{2}}$. Thus, the expected degree matrix $\tilde{D}$ is diagonal with entries $\tilde{D}_{ii} = 1 - p + \frac{N}{2}(p+q)$.

To find the second largest eigenvalue λ_2, we need to spectrally analyze the (expected) normalized Laplacian of $\tilde{A}$; that is, $\mathcal{L} = I - \tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2}$. We have that $\tilde{D}^{-1/2} = \left(\frac{1}{1-p+\frac{N}{2}(p+q)} \right)^{1/2} I_N$, so

$$\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2} = \left(\frac{1}{1 - p + \frac{N}{2}(p+q)} \right) \tilde{A} \coloneqq \tilde{d} \tilde{A}$$

We need to find λ such that $\det(\mathcal{L} - \lambda I) = 0$. We have

$$\mathcal{L} - \lambda I = I - \tilde{d}\tilde{A} - \lambda I = -\tilde{d}\tilde{A} - (\lambda - 1) I = \begin{pmatrix} -\tilde{d}P - (\lambda-1)I & -\tilde{d}Q \\ -\tilde{d}Q & -\tilde{d}P - (\lambda-1)I \end{pmatrix}$$

$(-\tilde{d}P - (\lambda-1)I)$ and $-\tilde{d}Q$ commute, so by [58], the determinant of that matrix is

$$\det\left(\left(-\tilde{d}P - (\lambda-1)I\right)^2 - \left(-\tilde{d}Q\right)^2 \right) = \det\left(\tilde{d}(Q-P) - (\lambda-1)I\right) \det\left(-\tilde{d}(P+Q) - (\lambda-1)I\right)$$

We have that $Q - P = (q-p) \cdot \mathbb{1}_{\frac{N}{2}} + (1-p) I_{\frac{N}{2}}$, which has eigenvalues $(1-p)$ and $((q-p)\frac{N}{2} + (1-p))$, so we finally get the required eigenvalue

$$\lambda_2 = \tilde{d}((q-p)\frac{N}{2} + (1-p)) + 1 = \frac{(q-p)\frac{N}{2} + (1-p)}{(q+p)\frac{N}{2} + (1-p)} + 1$$

For $N > 2$ and $p, q \in (0,1)$: $\frac{\partial}{\partial p}\left(\frac{(q-p)\frac{N}{2} + (1-p)}{(q+p)\frac{N}{2} + (1-p)} + 1 \right) = -\frac{2N(Nq+2)}{((N-2)p + Nq + 2)^2} < 0$, while $\frac{\partial}{\partial q}\left(\frac{\left((p-q)\frac{N}{2} + (1-p)\right)}{\left((p+q)\frac{N}{2} + (1-p)\right)} + 1 \right) = \frac{2N^2 p}{((N-2)p + Nq + 2)^2} > 0$. This proves that λ_2 increases when p decreases, and when q increases. So a higher spectral gap is related to a lower community structure. □

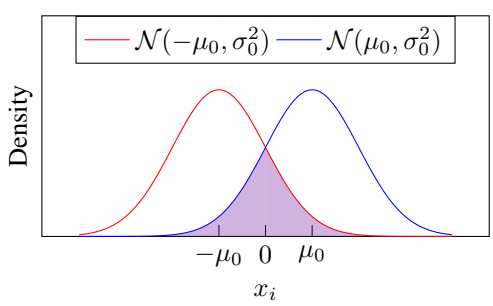
A.2 Proof of Theorem 2

Proof. We consider an SBM with 2 classes and $\frac{N}{2}$ nodes in each class, with intra-class edge probability p and inter-class edge probability q. Each node $i \in \{0, \dots, N-1\}$ has one feature, x_i, and a label ℓ_i which corresponds to the block it belongs to: $\ell_i = 1 \Leftrightarrow i \geq \frac{N}{2}$. The task is, therefore, to predict each node's community association. In this case, the alignment of communities and labels is perfect.

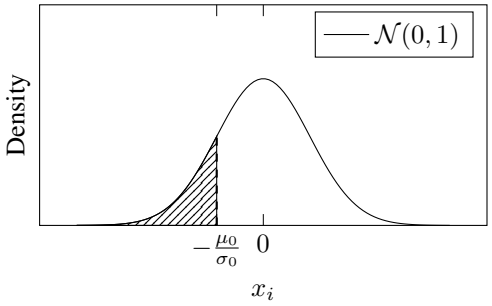
Each feature x_i is aligned with its label following a normal distribution: class-0 node features follow $\mathcal{N}(-\mu_0, \sigma_0^2)$, while class-1 node features follow $\mathcal{N}(\mu_0, \sigma_0^2)$, as shown in Figure 7(a). A perfect classifier f without any knowledge of the graph structure builds a decision boundary at $x = 0$. The expected number of misclassified nodes is $\frac{N}{2}$ times the intersection area of both distributions —because they are normalized from a population of $\frac{N}{2}$ each. Such area is $2 \cdot \Phi(\frac{-\mu_0}{\sigma_0})$, where Φ is the cumulative distribution function of the standard normal distribution (see Figure 7(b)). Therefore,

the proportion of misclassifications is $e(f) = \frac{N}{2} \cdot 2 \cdot \Phi(\frac{-\mu_0}{\sigma_0}) \cdot \frac{1}{N} = \Phi(\frac{-\mu_0}{\sigma_0})$. As Φ is a cumulative function, it is monotonically increasing with respect to its argument.

The classification error of f can be reduced by performing a step of message passing on the graph, which utilizes the community information to further separate the two classes. We shall consider a single round of sum aggregation as an example.



(a) The distribution of features from both clusters before training. The area in purple corresponds to nodes wrongly classified by the decision boundary $x = 0$.



(b) The cumulative distribution function at $x = -1$ of $\mathcal{N}(\mu_0, \sigma_0^2)$ is equal to the cumulative distribution function at $x = -\frac{\mu_0}{\sigma_0}$ of the standard normal distribution $\mathcal{N}(0, 1)$, which is $\Phi(\frac{-\mu_0}{\sigma_0})$. The purple area of Figure 7(a) is two times this quantity.

Any node has an expected $\mathcal{E}_p = p \cdot \left(\frac{N}{2} - 1\right)$ intra-class neighbors, plus itself, and an expected $\mathcal{E}_q = q \cdot \frac{N}{2}$ inter-class neighbors. In the proof A.3 we compute the same quantity with neighbour distributions instead, for a more fine-grained approximation. The hidden state of a class-1 node i after a step of sum aggregation is therefore the sum of $\mathcal{E}_p + 1$ random variables $\sim \mathcal{N}(\mu_0, \sigma_0^2)$ and $\mathcal{E}_q$ random variables $\sim \mathcal{N}(-\mu_0, \sigma_0^2)$. This follows another normal distribution with mean $\mu_1 \coloneqq \mu_0 \cdot (1 + \mathcal{E}_p - \mathcal{E}_q)$ and variance $\sigma_1^2 \coloneqq \sigma_0^2 \cdot (1 + \mathcal{E}_p + \mathcal{E}_q)$. Conversely, the hidden state of a class-0 node i follows a normal distribution of mean $-\mu_1$ and the same variance σ_1^2. The decision boundary of a perfect classifier is still at $x = 0$, but the average proportion of misclassified nodes is now $\Phi(\frac{-\mu_1}{\sigma_1})$, which depends on p and q. Specifically, it tends to be monotonically decreasing with respect to p; this means that the higher the community structure, the more accurate the classifier can be, because there is more information to utilize.

Let us take $\mu_0 = 1$ and $\sigma_0 = 1$ to simplify the calculations. We need to check that $\frac{\partial}{\partial p}\left(\frac{-\mu_1}{\sigma_1}\right) < 0$. For $N > 2$ and $p, q \in (0, 1)$:

$$\mu_1 = 1 \cdot (1 + p\left(\frac{N}{2} - 1\right) - q \cdot \frac{N}{2}) = 1 - p + \frac{N}{2} \cdot (p - q)$$

$$\sigma_1^2 = 1 \cdot (1 + p\left(\frac{N}{2} - 1\right) + q \cdot \frac{N}{2}) = 1 - p + \frac{N}{2} \cdot (p + q)$$

$$-\frac{\mu_1}{\sigma_1} = -\frac{1 - p + \frac{N}{2} \cdot (p - q)}{\sqrt{1 - p + \frac{N}{2} \cdot (p + q)}}$$

$$\frac{\partial}{\partial p}\left(\frac{-\mu_1}{\sigma_1}\right) = -\frac{(N-2)((N-2)p + 3Nq + 2)}{2\sqrt{2}\left((N-2)p + Nq + 2\right)^{\frac{3}{2}}} < 0$$

$$\iff (N-2)((N-2)p + 3Nq + 2) > 0$$

On the other side, $\frac{\partial}{\partial q}\left(\frac{-\mu_1}{\sigma_1}\right) > 0$.

$$\frac{\partial}{\partial q}\left(\frac{-\mu_1}{\sigma_1}\right) = \frac{N(6 + 3(N-2)p + Nq)}{2\sqrt{2}(2 + (N-2)p + Nq)^{\frac{3}{2}}} > 0 \iff N(6 + 3(N-2)p + Nq) > 0$$

This proves that, by reducing the community structure (either by decreasing p or increasing q), then the quantity $\frac{-\mu_1}{\sigma_1}$ increases, so the expected proportion of misclassified nodes $e(f) = \Phi\left(\frac{-\mu_1}{\sigma_1}\right)$ also increases. In consequence, it harms the performance of classifier f.

The graph's information provides a better separation between the two classes if the intra-class edge probability is high enough. From this we can conclude that reducing the intra-class edge probability is not a good strategy to improve the classification performance for any model on the graph. □

A.3 Proof of Theorem 3

Proof. We consider another SBM with 2 classes and $\frac{N}{2}$ nodes in each class, with intra-class edge probability p and inter-class edge probability q. Each node $i \in \{0, \dots, N-1\}$ has again one feature, x_i, aligned with its class label ℓ_i following a normal distribution: $\mathcal{N}(-\mu_0, \sigma_0^2)$ for class 0 and $\mathcal{N}(\mu_0, \sigma_0^2)$ for class 1. However, now ℓ_i corresponds to its community with a fixed probability ψ —recovering Theorem 2 when $\psi = 1$.

What is the probability of any node i such that, after a round of sum aggregation, its modified representation x'_i is now misclassified (M)? As the two classes are symmetric:

$$\begin{aligned} P(M) &= P(M, L_0) + P(M, L_1) \\ &= P(L_0)P(M|L_0) + P(L_1)P(M|L_1) \\ &= \frac{1}{2}P(M|L_0) + \frac{1}{2}P(M|L_1) \\ &= P(M|L_0) \end{aligned}$$

Then the question becomes the following: what is the probability of a node with label L_0 being misclassified? It depends whether it belongs to community C_0 or C_1.

$$\begin{aligned} P(M|L_0) &= P(M, C_0|L_0) + P(M, C_1|L_0) \\ &= P(C_0|L_0)P(M|L_0, C_0) + P(C_1|L_0)P(M|L_0, C_1) \\ &= \psi P(M|L_0, C_0) + (1-\psi)P(M|L_0, C_1) \end{aligned}$$

$P(M|L_0, C_0) = P(X'_{(L_0,C_0)} > 0)$. We now need to calculate what is the predicted label of a (L_0, C_0) node after a sum aggregation round. For this we need the distribution of its neighbours. We consider the node to have a self loop, as it uses its own feature too.

- The number of nodes (L_0, C_0) (that are not node i) follows a binomial distribution $N_0 \sim B(\frac{N}{2} - 1, \psi)$. However, for easiness of proof we will approximate it by a normal distribution, which is appropriate for N large enough: $N_0 \sim \mathcal{N}((\frac{N}{2} - 1)\psi, (\frac{N}{2} - 1)\psi(1-\psi))$. The amount of them connected to node i follows a conditional binomial distribution $H_{00} \sim B(n_0, p) \mid N_0 = n_0$, which we again approximate by $H_{00} \sim \mathcal{N}(n_0 p, n_0 p(1-p)) \mid N_0 = n_0$.

- The number of nodes (L_0, C_1) that are connected to node i follows $H_{01} \sim B(\frac{N}{2} - 1 - n_0, q) \mid N_0 = n_0$, approximated by $H_{01} \sim \mathcal{N}((\frac{N}{2} - 1 - n_0)q, (\frac{N}{2} - 1 - n_0)q(1-q)) \mid N_0 = n_0$.

- Since H_{00} and H_{01} are conditionally independent given $N_0 = n_0$, their sum $H_0 = H_{00} + H_{01}$ also follows a normal distribution with parameters given by the sum of their means and variances. Thus, the number of total L_0 nodes connected to node i (except itself) follows $H_0 \sim \mathcal{N}(n_0 p + (\frac{N}{2} - 1 - n_0)q, n_0 p(1-p) + (\frac{N}{2} - 1 - n_0)q(1-q)) \mid N_0 = n_0$. We are going to get rid of the dependency of N_0 by estimating it by a normal distribution with the

mean and variance of the marginal distribution of H_0:

$$\begin{aligned}\mathbb{E}[H_0] &= \mathbb{E}[\mathbb{E}[H_0|N_0]] = \mathbb{E}[N_0]p + \left(\frac{N}{2} - 1 - \mathbb{E}[N_0]\right) q \\ &= \left(\frac{N}{2} - 1\right)\psi p + \left(\frac{N}{2} - 1 - \left(\frac{N}{2} - 1\right)\psi\right) q \\ &= \left(\frac{N}{2} - 1\right)(p\psi + q(1-\psi))\end{aligned}$$

$$\begin{aligned}\text{Var}[H_0] &= \mathbb{E}[\text{Var}(H_0|N_0)] + \text{Var}(\mathbb{E}[H_0|N_0]) \\ &= \mathbb{E}[N_0]p(1-p) + \left(\frac{N}{2} - 1 - \mathbb{E}[N_0]\right) q(1-q) \\ &+ \text{Var}\left(N_0(p-q) + \left(\frac{N}{2} - 1\right) q\right) \\ &= \left(\frac{N}{2} - 1\right)\psi p(1-p) + \left(\frac{N}{2} - 1 - \left(\frac{N}{2} - 1\right)\psi\right) q(1-q) \\ &+ (p-q)^2\left(\frac{N}{2} - 1\right)\psi(1-\psi) \\ &= \left(\frac{N}{2} - 1\right)(\psi p(1-p) + (1-\psi)q(1-q) + (p-q)^2\psi(1-\psi))\end{aligned}$$

- The number of nodes (L_1, C_1) follows $N_1 \sim B(\frac{N}{2}, \psi)$, approximated by $N_1 \sim \mathcal{N}(\frac{N}{2}\psi, \frac{N}{2}\psi(1-\psi))$. The amount of them connected to node i follows $H_{11} \sim B(n_1, q) \mid N_1 = n_1$, approximated by $H_{11} \sim \mathcal{N}(n_1 q, n_1 q(1-q)) \mid N_1 = n_1$.
- The number of nodes (L_1, C_0) that are connected to node i follows $H_{10} \sim B(\frac{N}{2} - n_1, p) \mid N_1 = n_1$, approximated by $H_{10} \sim \mathcal{N}((\frac{N}{2} - n_1)p, (\frac{N}{2} - n_1)p(1-p)) \mid N_1 = n_1$.
- Similarly to L_0, the number of total L_1 nodes connected to node i follows $H_1 \sim \mathcal{N}(n_1 q + (\frac{N}{2} - n_1)p, n_1 q(1-q) + (\frac{N}{2} - n_1)p(1-p)) \mid N_1 = n_1$. We will estimate it by a normal distribution with its mean and variance:

$$\begin{aligned}\mathbb{E}[H_1] &= \mathbb{E}[\mathbb{E}[H_1|N_1]] = \mathbb{E}[N_1]q + (\frac{N}{2} - \mathbb{E}[N_1])p \\ &= \frac{N}{2}\psi q + \left(\frac{N}{2} - \frac{N}{2}\psi\right) p \\ &= \frac{N}{2}(p(1-\psi) + q\psi)\end{aligned}$$

$$\begin{aligned}\text{Var}[H_1] &= \mathbb{E}[\text{Var}(H_1|N_1)] + \text{Var}(\mathbb{E}[H_1|N_1]) \\ &= \mathbb{E}[N_1]q(1-q) + \left(\frac{N}{2} - \mathbb{E}[N_1]\right) p(1-p) + \text{Var}\left(N_1(q-p) + \frac{N}{2}p\right) \\ &= \frac{N}{2}\psi q(1-q) + \left(\frac{N}{2} - \frac{N}{2}\psi\right) p(1-p) + (p-q)^2\frac{N}{2}\psi(1-\psi) \\ &= \frac{N}{2}(\psi q(1-q) + (1-\psi)p(1-p) + (p-q)^2\psi(1-\psi))\end{aligned}$$

The representation of node i after one step of sum aggregation is the summation of $H_0 + 1$ (independent) normal distributions $\sim \mathcal{N}(-\mu_0, \sigma_0^2)$ and H_1 (independent) normal distributions $\sim \mathcal{N}(\mu_0, \sigma_0^2)$. Therefore:

$$X'_{(L_0,C_0)} \sim \mathcal{N}(-\mu_0(1 + h_0 - h_1),\ \sigma_0^2(1 + h_0 + h_1)) \mid H_0 = h_0,\ H_1 = h_1$$

Again calculating its mean and variance:

$$\begin{aligned}\mathbb{E}[X'_{(L_0,C_0)}] &= \mathbb{E}[\mathbb{E}[X'_{(L_0,C_0)}|H_0,H_1]] = -\mu_0(1+\mathbb{E}[H_0]-\mathbb{E}[H_1]) \\ &= -\mu_0\left(1+\left(\frac{N}{2}-1\right)(p\psi+q(1-\psi))-\frac{N}{2}(p(1-\psi)+q\psi)\right)\end{aligned}$$

$$\begin{aligned}\text{Var}[X'_{(L_0,C_0)}] &= \mathbb{E}[\text{Var}(X'_{(L_0,C_0)}|H_0,H_1)] + \text{Var}(\mathbb{E}[X'_{(L_0,C_0)}|H_0,H_1]) \\ &= \sigma_0^2(1+\mathbb{E}[H_0]+\mathbb{E}[H_1]) + \mu_0^2(\text{Var}(H_0)+\text{Var}(H_1)) \\ &= \sigma_0^2\left(1+\left(\frac{N}{2}-1\right)(p\psi+q(1-\psi))+\frac{N}{2}(p(1-\psi)+q\psi)\right) \\ &+ \mu_0^2\left(\left(\frac{N}{2}-1\right)(\psi p(1-p)+(1-\psi)q(1-q)+(p-q)^2\psi(1-\psi))\right. \\ &\left.+\frac{N}{2}(\psi q(1-q)+(1-\psi)p(1-p)+(p-q)^2\psi(1-\psi))\right)\end{aligned}$$

For a more clear analysis of this formula, we take $\mu_0 = 1, \sigma_0 = 1$ and N large enough:

$$\mathbb{E}[X'_{(L_0,C_0)}] \approx -\frac{N}{2}(p\psi+q(1-\psi)-p(1-\psi)-q\psi) = -\frac{N}{2}(2\psi-1)(p-q)$$

$$\begin{aligned}\text{Var}[X'_{(L_0,C_0)}] &\approx \frac{N}{2}\Big(p\psi+q(1-\psi)+p(1-\psi)+q\psi+2(p-q)^2\psi(1-\psi) \\ &+\psi p(1-p)+(1-\psi)q(1-q)+\psi q(1-q)+(1-\psi)p(1-p)\Big) \\ &= \frac{N}{2}(p+q+p(1-p)+q(1-q)+2(p-q)^2\psi(1-\psi))\end{aligned}$$

Finally, we have $P(X'_{(L_0,C_0)} > 0) \approx$

$$\Phi\left(\frac{-\frac{N}{2}(2\psi-1)(p-q)}{\sqrt{\frac{N}{2}(p+q+p(1-p)+q(1-q)+2(p-q)^2\psi(1-\psi))}}\right)$$

For $P(M|L_0,C_1) = P(X'_{(L_0,C_1)} > 0)$, the calculation of the predicted label of a (L_0,C_1) node follows exactly the same steps, but exchanging p and q, as the probabilities for nodes to be connected to node i are now exactly of the opposite community. So we have $P(X'_{(L_0,C_1)} > 0) \approx$

$$\Phi\left(\frac{-\frac{N}{2}(2\psi-1)(q-p)}{\sqrt{\frac{N}{2}(p+q+p(1-p)+q(1-q)+2(p-q)^2\psi(1-\psi))}}\right) = 1 - P(X'_{(L_0,C_0)} > 0)$$

And $P(M) \approx \psi P(X'_{(L_0,C_0)} > 0) + (1-\psi)(1-P(X'_{(L_0,C_0)} > 0)) = (1-\psi) + (2\psi - 1)P(X'_{(L_0,C_0)} > 0)$. □

B Additional results

In Table 5 we compare our results with an additional baseline [5] (DHGR), which uses a feature similarity based rewiring for heterophilic graphs. As there was no code available to reproduce the results, we take the results reported from the paper.

C Effect on homophily

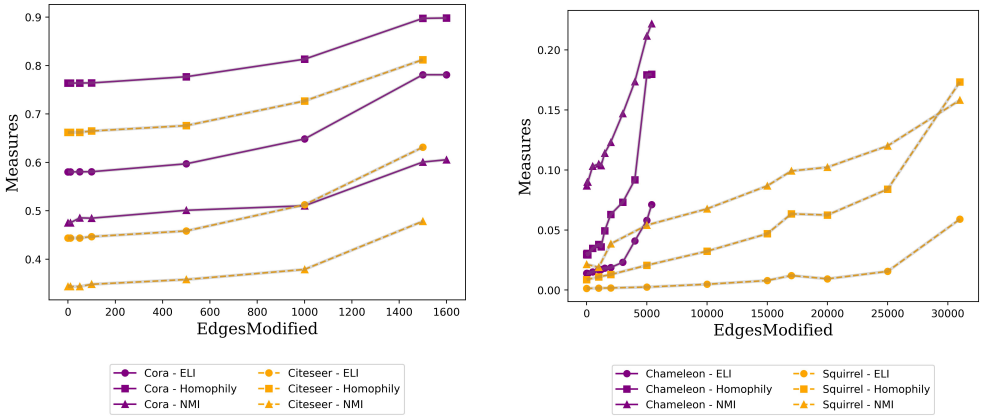
Graph neural networks provably perform better on homophilic graphs and some *good-heterophilic* graphs [39]. We investigate the effect our one-shot rewiring strategy (GCN☼GCN) has on Edge label

Table 4: Node classification on 2 homophilic and 2 heterophilic graphs using Iterative SoLAR.

	GCN✡GCN-Delete				GCN✡GCN-Add			
Dataset	Cora	Citeseer	Chameleon	Squirrel	Cora	Citeseer	Chameleon	Squirrel
Iterations	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy
0	84.13±0.32	72.58±1.85	29.31±2.09	26.44±0.28	84.13±0.32	72.58±1.85	29.31±2.09	26.44±0.28
1	86.45±0.58	77.0±0.31	31.31±1.01	31.31±0.32	**92.26±0.00**	76.82±0.16	31.52±0.17	**34.14±0.31**
2	85.94±0.48	**78.9±0.46**	31.1±0.77	32.06±0.16	88.39±0.00	**79.76±0.79**	**31.79±0.14**	30.97±0.40
3	**89.16±0.95**	77.51±0.57	**35.66±0.35**	**33.20±1.12**	84.26±0.52	77.22±0.65	30.62±0.51	26.97±0.86
4	86.19±1.85	73.86±1.6	34.62±2.39	31.76±0.82	81.94±0.00	78.2±2.15	25.52±1.23	30.14±0.35

	GAT✡GAT-Delete				GAT✡GAT-Add			
0	83.48±1.20	77.55±2.65	35.66±3.48	35.17±1.94	83.48±1.20	77.55±2.65	35.66±3.48	35.17±1.94
1	88.65±0.66	**81.14±0.16**	33.03±1.16	38.28±0.98	**88.26±0.63**	76.33±0.26	27.93±0.22	38.55±1.42
2	88.77±1.12	76.16±2.23	34.41±0.26	**40.34±0.79**	83.87±0.41	77.63±0.83	30.14±0.41	**41.45±1.20**
3	88.65±1.85	76.98±0.76	**36.83±1.67**	35.45±3.03	87.35±0.66	**77.96±0.46**	**34.97±0.83**	40.83±2.53
4	**89.94±1.45**	73.96±2.68	35.03±0.56	39.24±3.18	85.94±0.63	73.14±0.79	26.41±0.60	41.17±1.75

informativeness (ELI) and adjusted homophily score proposed in [47] and report the Normalized Mutual Information between the node ground truth labels and community membership labels after performing modularity maximization [16] on the rewired graph in §C. Evidently, our rewiring strategy improves the homophily score, as well as the edge label informativeness (denoted by ELI), which is also found to have high correlation to GNN performance [47]. We also better align the node ground truth labels to community labels, as we delete inter-community edges (denoted by NMI).



(c) ELI, Homophily, NMI for Cora and Citeseer with GCN✡GCN.

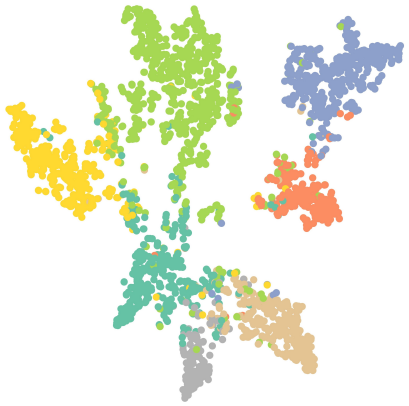
(d) ELI, Homophily, NMI for Chameleon and Squirrel with GCN✡GCN.

Figure 7: The effect of one-shot rewiring on ELI, homophily and NMI on Cora, Citeseer, Chameleon and Squirrel datasets.

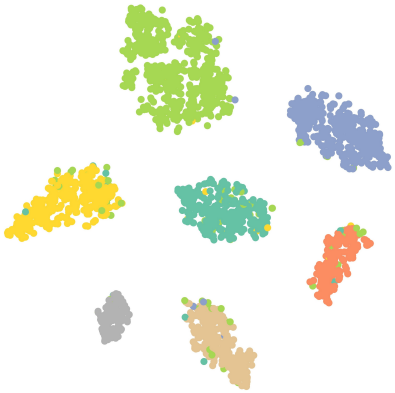
C.1 Iterative SoLAR

As outlined in Figure 1, we can have an iterative rewiring procedure that constantly updates its predictions till certain number of edges are modified. In Table 4, we present results on Cora [40], Citeseer [54], Chameleon and Squirrel [48] using the iterative version of rewiring for combinations GCN✡GCN and GATv2✡GATv2 both deletions and additions. The iteration 0 refers to the baseline, where no rewiring is performed. Since it is more computationally expensive to perform iterative rewiring on several splits of the data, we instead divide the datasets into one 80/20 split for train/test nodes respectively, and report the final test accuracy averaged over 5 random seeds. The hidden dimension used is $\{32,128\}$. The learning rate is $\{0.01, 0.001\}$ with no dropout and weight decay. The hyperparameters obtained from the one-shot pruning were applicable here as well. We can see from the table that we can get significant boost in the accuracy (for instance, Cora 92.26%) by doing iterative rewiring, as the proxy labels get iteratively better. Our experiments indicate that usually 2-3 iterations of edge modifications should suffice to get a good performance. We also visualize a T-SNE plot in Figure 8 of the node embeddings after training on the original graph and the rewired

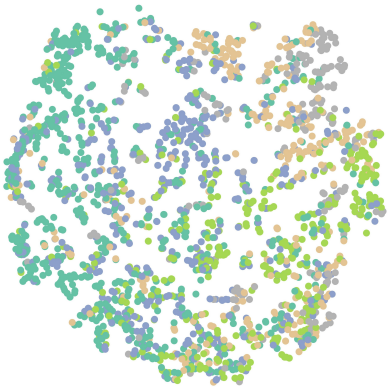
graph (GCN ☼GCN) on Cora and Squirrel datasets. From the figure, we can see that the classes are more separable in the embedding space on the rewired graph, the class separability is more evident in a homophilic graph like Cora (Figure 8(b) than in a heterophilic graph like Squirrel (8(d)), highlighting the fact that GNNs are usually more useful in homophilic settings and if the surrogate model gives noisy labels for rewiring, the performance on the downstream is also affected. These plots further substantiate our claims about task-community alignment and how our rewiring can effect these changes.



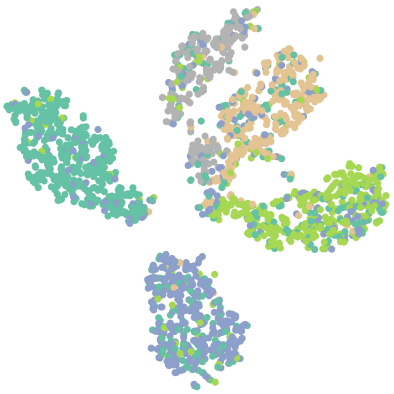
(a) GCN trained on the original Cora graph.



(b) GCN☼GCN trained on Cora with 1500 inter-class edge deletions.



(c) GCN trained on the original Squirrel graph.



(d) GCN☼GCN trained on Squirrel with 20K inter-class edge deletions.

Figure 8: We plot T-SNE for Cora and Squirrel datasets after training a GCN on the original graph and the rewired graph.

D Training details

We use PyTorch-Geometric and DGL library for all our experiments. We use a 2-layered GCN [34] and GATv2 [8] with $\{8, 16\}$ attention heads. For datasets Cora, Citeseer, Pubmed, Cornell, Texas, Wisconsin, Chameleon, Squirrel, Actor, CS, Physics and Photo the final test accuracy is reported averaged over 100 splits, run for 100 epochs. We use the split mechanism introduced in [56]. The weight decay and dropout are set to 0. The hidden dimension sizes we experimented with are {32,128,512} and learning rate {0.01,0.001}. The heterophilic graphs (Cornell, Texas, Wisconsin, Chameleon, Squirrel and Actor) are taken from [48]. For experiments on Roman-empire and Amazon-ratings, we use the code base provided by [48], where the datasets are split into 50/25/25 for train/test/validation respectively. The accuracy is averaged over 10 runs run for 1000 epochs. We use a 5-layered GCN and GATv2 for these experiments, which are further augmented with skip connections, layernorm [3] and batchnorm [30] to facilitate training them better. For the Penn94

Table 5: Node classification on heterophilic graphs using one-shot rewiring.

Method	Cornell	Texas	Wisconsin	Chameleon	Squirrel	Actor
GCN	68.31±8.13	73.47±10.13	66.14±9.23	54.64±6.94	43.25±6.32	28.26±3.22
GAT	86.84±9.78	89.01±10.43	87.56±9.20	61.79±10.20	45.71±5.12	29.41±2.98
GCN+FoSR	71.64±9.80	73.93±10.23	65.85±7.73	54.40±6.58	42.80±6.40	28.66±3.21
GAT+FoSR	76.12±6.51	78.15±7.81	74.08±9.01	46.48 ± 4.97	47.40±7.17	27.45±3.61
GCN+Proxydelmin	81.94±7.96	83.46±10.90	70.63±7.68	53.64±6.00	41.26±5.35	28.58±2.93
GAT+Proxydelmin	85.40±7.64	83.44±9.52	79.78±11.26	66.15±12.01	45.02±5.13	**32.37±4.36**
GCN+DHGR	67.38±5.33	81.78±0.89	76.47±3.62	**70.83±2.03**	**67.15±1.43**	**36.29±0.12**
GAT+DHGR	70.09±6.77	83.78±3.37	73.20±4.89	**72.11±2.87**	**62.37±1.78**	**34.71±0.48**
GCN✡GCN+Delete	68.35±8.54	74.12±9.89	67.85±7.14	57.19 ± 6.45	44.50±6.29	29.25±3.50
GCN✡GCN+Add	69.42±8.93	74.20±10.26	68.51±7.20	56.43 ± 6.16	44.04±6.34	28.16±3.22
GAT✡GAT+Delete	**87.40±9.89**	**90.14±10.64**	**88.32±9.08**	68.89±11.50	49.10±5.59	30.31±4.29
GAT✡GAT+Add	87.12±9.59	87.97±10.95	87.76±9.57	66.35±11.18	46.44±6.00	29.46±4.67

dataset introduced in [38], we use hidden dimension size of 32, learning rate set to 0.01, weight decay $1e-3$ and also batchnorm. All the experiments were done on 2 V100 GPUs. The hyperparameters used for our experiments are provided in the tables below. The runtime is provided in seconds for one-shot rewiring. The statistics for the datasets used is given in Table 6. Our code is available.

Table 6: Statistics of the graphs used. We use the largest connected component for all our experiments.

Dataset	#Nodes	#Edges
Cora	2,708	10,138
Citeseer	3,327	7,358
Pubmed	19,717	88,648
Cornell	183	277
Texas	183	279
Wisconsin	251	450
Chameleon	890	8,854
Squirrel	2,223	57,850
Actor	7,600	26,659
CS	18,333	1,63,788
Physics	34,493	4,95,924
Photo	7,650	2,38,162
Roman-empire	22,662	32,927
Amazon-ratings	24,492	93,050
Penn94	41,554	13,62,229

Table 7: Hyperparameters for GCN✡GCN+Del

Dataset	EdgesDeleted	LR	HiddenDimension	Runtime
Cora	1500	0.01	32	71.43
Citeseer	1500	0.01	32	84.08
Pubmed	10000	0.01	32	90.79
Cornell	100	0.001	128	86.76
Texas	100	0.001	128	73.94
Wisconsin	100	0.001	128	77.23
Chameleon	5400	0.001	128	76.82
Squirrel	310000	0.001	128	78.70
Actor	16000	0.001	128	80.12
CS	22000	0.01	128	200.90
Physics	30000	0.01	128	412.47
Photo	35000	0.01	512	263.11

Table 8: Hyperparameters for GCN✡GCN+Add

Dataset	EdgesAdded	LR	HiddenDimension	Runtime
Cora	6929	0.01	32	89.43
Citeseer	7168	0.01	32	70.88
Pubmed	352	0.01	32	94.07
Cornell	55	0.001	128	88.76
Texas	54	0.001	128	74.33
Wisconsin	41	0.001	128	86.68
Chameleon	4088	0.001	128	70.35
Squirrel	12349	0.001	128	74.85
Actor	12215	0.001	128	78.38
CS	8680	0.01	32	129.36
Physics	45991	0.01	32	351.85
Photo	26846	0.01	32	108.79

Table 9: Hyperparameters for GAT☼GAT+Add

Dataset	EdgesAdded	LR	HiddenDimension	Runtime
Cora	9711	0.001	32	149.73
Citeseer	11996	0.001	32	192.35
Pubmed	17647	0.001	32	594.85
Cornell	37	0.001	32	134.70
Texas	55	0.001	32	123.80
Wisconsin	49	0.001	32	135.06
Chameleon	4167	0.001	32	105.65
Squirrel	20754	0.001	32	313.19
Actor	30251	0.001	32	388.99
CS	27592	0.001	32	2292.85
Physics	46700	0.001	32	1761.90
Photo	27713	0.01	32	456.02

Table 10: Hyperparameters for GAT☼GAT+Del

Dataset	EdgesDeleted	LR	HiddenDimension	Runtime
Cora	1700	0.001	32	105.27
Citeseer	1500	0.001	32	116.28
Pubmed	14126	0.001	32	395.73
Cornell	120	0.001	32	100.89
Texas	120	0.001	32	131.33
Wisconsin	120	0.001	32	131.69
Chameleon	6000	0.001	32	169.09
Squirrel	35000	0.001	32	212.64
Actor	30000	0.001	32	139.54
CS	30000	0.001	32	1579.53
Physics	30000	0.001	32	3766.81
Photo	40264	0.001	32	450.34