Spectral Graph Pruning Against Over-Squashing and Over-Smoothing

Adarsh Jamadandi [*] [1] [2] **Celia Rubio-Madrigal** [*] [2] **Rebekka Burkholz** [*] [2]

Abstract

This report analyzes the paper addressing critical challenges in Graph Neural Networks (GNNs) through spectral graph pruning. The authors propose a novel spectral approach to mitigate over-smoothing and over-squashing by systematically pruning problematic edges. Key contributions include a theoretically grounded pruning framework and empirical validation across multiple benchmarks.

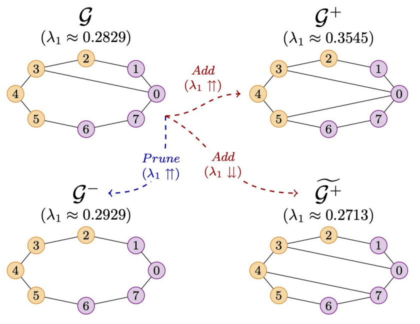
$\mathcal{G}$
$(\lambda_1 \approx 0.2829)$
$\mathcal{G}^+$
$(\lambda_1 \approx 0.3545)$
Add
$(\lambda_1 \upuparrows)$
Prune
$(\lambda_1 \upuparrows)$
Add
$(\lambda_1 \downdownarrows)$
$\mathcal{G}^-$
$(\lambda_1 \approx 0.2929)$
$\widetilde{\mathcal{G}^+}$
$(\lambda_1 \approx 0.2713)$

Figure 1. Braess' paradox example

1. Introduction

Graph Neural Networks (GNNs) face two critical challenges: *over-squashing*, where information propagation is hindered by topological bottlenecks, and *over-smoothing*, where node features become indistinguishable due to excessive aggregation. Traditional solutions often trade off one problem for the other. This paper introduces a unified approach inspired by the Braess paradox, demonstrating that strategic edge deletions can mitigate both issues simultaneously by optimizing the spectral gap of the graph Laplacian. The authors propose efficient algorithms for graph rewiring and connect their findings to the graph lottery ticket hypothesis, showing that sparse subgraphs can achieve competitive performance.

2. Key Contributions

The paper makes three major contributions. First, it challenges the conventional dichotomy between over-squashing and over-smoothing by proving that edge deletions can increase the spectral gap (reducing bottlenecks) while preserving feature distinctiveness. This is formalized through a minimal example leveraging the Braess paradox which is operationalized through Lemma A.1 that provides a sufficient condition for spectral gap increase upon edge deletion.

Second, the authors introduce PROXYDELETE and PROXYADD, greedy algorithms that approximate spectral gap changes using matrix perturbation theory, enabling efficient edge modifications. They proposed two more rigourous and complex algorithms based on Eldan criterion (Eldan et al. (2017)) ELDANADD and ELDANDELETE that avoid spectral gap approximation but fail to scale in large graphs. Third, they bridge graph rewiring with graph lottery tickets, demonstrating that the resulting sparse subgraphs can achieve competitive performance.

3. Related Work

Prior work such as FoSR addresses over-squashing via graph rewiring and others via curvature optimization, while over-smoothing is tackled through regularization or adaptive aggregation. The authors differentiate their approach by showing that **edge deletions—not just additions—can expand the spectral gap**, a counterintuitive result rooted in the Braess paradox. They also extend the lottery ticket hypothesis to GNNs, arguing that spectral pruning identifies ”winning tickets” at initialization, reducing computational costs while maintaining performance. Connections to Eldan's criterion and matrix perturbation theory further distinguish their methodology from existing curvature-based or diffusion-inspired techniques.

4. Methodology

The core insight lies in optimizing the spectral gap λ_1 of the normalized graph Laplacian $\mathcal{L}_G$. A larger λ_1 reduces over-

[*]Equal contribution [1]Universität des Saarlandes, Saarbrücken, Germany [2]CISPA Helmholtz Center for Information Security, Saarbrücken, Germany. Correspondence to: Adarsh Jamadandi <adarsh.jam@gmail.com>, Celia Rubio-Madrigal <celia.rubio-madrigal@cispa.de>, Rebekka Burkholz <burkholz@cispa.de>.

Proceedings of the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

squashing by improving information flow, while selective edge deletions prevent over-smoothing by limiting harmful aggregations.

4.1. Eldan's Algorithm

ELDANDELETE and ELDANADD rigorously identify edge deletions/additions that maximize λ_1 using Lemma 3.1 from Eldan et al. (2017). For an edge (u, v), the spectral impact is evaluated via the criterion $g(u, v, \mathcal{L}_G)$ given by :

$$-2(1-\lambda_1)\left(\frac{\sqrt{d_u+1}-\sqrt{d_u}}{\sqrt{d_u+1}}f_u^2+\frac{\sqrt{d_v+1}-\sqrt{d_v}}{\sqrt{d_v+1}}f_v^2\right) \\ -\mathcal{P}_f^2\lambda_1+\frac{2f_uf_v}{\sqrt{d_u+1}\sqrt{d_v+1}},$$

where $\mathcal{P}_f = \langle f, \hat{f}_0 \rangle$ projects the eigenvector f onto the top eigenvector $\hat{f}_0$ of the modified graph. If $g(u, v, \mathcal{L}_G) > 0$, deleting (u, v) increases λ_1. While theoretically sound, computing $\mathcal{P}_f$ and eigenvector updates for each candidate edge requires $O(|\mathcal{V}|)$ operations per iteration, making it infeasible for large graphs.

4.2. Proxy Methods

To address scalability, PROXYDELETE and PROXYADD approximate λ_1 changes using first-order matrix perturbation theory:

$$\hat{\lambda}_1 \approx \lambda_1 + \Delta w_{uv}\left[(f_u - f_v)^2 - \lambda_1(f_u^2 + f_v^2)\right],$$

where $\Delta w_{uv} = -1$ (deletion) or $+1$ (addition). This bypasses full eigendecompositions by leveraging the current eigenvector f and eigenvalue λ_1. Edges are ranked in $O(|\mathcal{E}|)$ time per batch, with periodic power iteration updates to f and λ_1. The proxy achieves 85–90% of Eldan's spectral gap improvements (Table 16) while reducing time complexity from $O(N|\mathcal{E}||\mathcal{V}|)$ to $O(N|\mathcal{E}|)$, enabling applications to graphs with 10k+ nodes.

5. Experiments

The authors' experiments substantiate the proposed methodology by employing a node-level task across three distinct types of graph datasets: classic homophilic (e.g., Cora, Citeseer), long-range (e.g., PascalVOC-SP, Peptides, Roman-Empire), and heterophilic datasets (e.g., Amazon ratings, Texas). The study aims to evaluate the effectiveness of their algorithms in comparison with GCN and GAT baseline models, as well as other algorithms from the literature, such as FoSR, which will be examined in detail in the subsequent section of the report.

Table 1. Results on Long Range Graph Benchmark datasets.

Method	PascalVOC-SP (Test F1 ↑)	Peptides-Func (Test AP ↑)	Peptides-Struct (Test MAE ↓)
Baseline-GCN	0.1268±0.0060	0.5930±0.0023	0.3496±0.0013
DRew+GCN	0.1848±0.0107	**0.6996±0.0076**	0.2781±0.0028
FoSR+GCN	0.2157±0.0057	0.6526±0.0014	0.2499±0.0006
ProxyAdd+GCN	**0.2213±0.0011**	0.6789±0.0002	**0.2465±0.0004**
ProxyDelete+GCN	0.2170±0.0015	0.6908±0.0007	**0.2470±0.0080**

Table 2. Node classification on Amazon-Ratings (heterophilic).

Method	#EdgesAdded	Accuracy	#EdgesDeleted	Accuracy	Layers
GCN	-	47.20±0.33	-	47.20±0.33	10
GCN+FoSR	25	49.68±0.73	-	-	10
GCN+Eldan	25	48.71±0.99	100	**50.15±0.50**	10
GCN+ProxyGap	10	49.72±0.41	50	**49.75±0.46**	10
GAT	-	47.43±0.44	-	47.43±0.44	10
GAT+FoSR	25	51.36±0.62	-	-	10
GAT+Eldan	25	51.68±0.60	50	**51.80±0.27**	10
GAT+ProxyGap	20	49.06±0.92	100	**51.72±0.30**	10
GCN	-	47.32±0.59	-	47.32±0.59	20
GCN+FoSR	100	49.57±0.39	-	-	20
GCN+Eldan	50	**49.66±0.31**	20	48.32±0.76	20
GCN+ProxyGap	50	49.48±0.59	500	**49.58±0.59**	20
GAT	-	47.31±0.46	-	47.31±0.46	20
GAT+FoSR	100	51.31±0.44	-	-	20
GAT+Eldan	20	51.40±0.36	20	**51.64±0.44**	20
GAT+ProxyGap	50	47.53±0.90	20	**51.69±0.46**	20

In Table 1, the results highlight the effectiveness of various methods on long-range graph benchmarks. Proxy-based approaches achieve the best performance on PascalVOC-SP and Peptides-Struct, suggesting their superiority in capturing long-range dependencies. These results emphasize the advantages of refining graph representations through edge modifications in long-range learning scenarios using the proposed proxy methods.

Table 2 presents node classification results on the heterophilic Amazon-Ratings dataset, showing that the proposed algorithms, particularly Eldan and ProxyGap, significantly outperform baseline GCN and GAT architectures as well as FoSR in both edge deletion and addition scenarios. These findings underscore the effectiveness of these algorithms even on extremely heterophilic graph datasets, which are traditionally considered a limitation of graph neural networks.

6. Our experiments

To evaluate the performance of the rewiring methods, we used datasets from different frameworks, following the approach of our reference paper. Below is a summary of the datasets used along with their defining properties.

Table 3. Experiment datasets characteristics

Dataset	ELI	Adj Hom	Type
Photo	0.67	0.78	Homophilic
COCO-SP	0.25	0.53	LRG
Tolokers	0.01	0.09	Heterophilic

We compare ProxyDelete and ProxyAdd with a baseline-GCN without any rewiring and the FoSR approach from (Karhadkar et al., 2022). Moreover, we use also in our COCO-SP experiment an algorithm designed to delete, then add edges using the same proxy of the spectral gap. Below are the results of our 3 experiments.

Table 4. Homophilic graph results (Photo)

	ValAcc	TestAcc	Initial Gap	Final Gap
Baseline-GCN	72.93±1.22	72.46±1.18	0.00146	0.00146
ProxyDelete+GCN	**73.56±1.22**	**72.82±1.23**	0.00146	0.00147
FoSR+GCN	73.02±1.39	73.65±1.13	0.00146	0.00148

Table 5. Heterophilic graph results (Tolokers)

	ValAcc	TestAcc	Initial Gap	Final Gap
Baseline-GCN	**78.36±0.21**	78.23±0.2	0.06744	0.06744
ProxyDelete+GCN	78.34±0.18	**78.29±0.25**	0.06744	0.05573

Table 6. Performance comparison on COCO-SP (long-range)

	ValAcc	TestAcc	Initial Gap	Final Gap
Baseline-GCN	**69.16±1.53**	**68.27±1.22**	0.00495	0.00495
ProxyDelete+GCN	68.29±1.75	68.04±1.32	0.00495	0.00505
ProxyAdd+GCN	67.88±1.82	67.86±1.46	0.00495	**0.01518**
FoSR+GCN	68.39±1.55	67.46±1.15	0.00495	0.00833
DeleteAdd+GCN	66.96±1.64	66.62±1.42	0.00495	0.01402

As observed, in the case of the long-range dataset COCO-SP, the baseline GCN without any rewiring outperforms all other models. However, in the two other experiments—on the heterophilic Toloker dataset and the homophilic Photo dataset—ProxyDelete, which is the main contribution of our reference paper, outperforms the other methods. Nonetheless, we observe a decrease in the spectral gap in our heterophilic experiment, indicating that the proxy used has certain limitations. Additionally, we were unable to apply all methods to all experiments due to the computational cost of the algorithms, particularly ProxyAdd. This is expected, as the space of potential edges that can be added is significantly larger than the set of existing edges that can be deleted.

7. FoSR: First-order Spectral Rewiring for addressing Over-squashing in GNNs

7.1. Motivation

We compare the previous approach with another spectral rewiring method, (Karhadkar et al., 2022), which we also used in our experiments. In this paper, the authors leverage edge additions to tackle over-squashing and employ relational GNNs (Battaglia et al., 2018) to reduce over-smoothing. This choice was motivated by the similar approach to spectral rewiring that both papers use. To reduce computational costs, instead of computing the spectral gap of the new graphs at each iteration, they use proxies to choose which edge to add (or delete, in the case of ProxyDelete) in both approaches. Moreover, as in (Jamadandi et al., 2024), FoSR addresses the issue of over-smoothing resulting from spectral rewiring.

7.2. Summary of the paper

In (Karhadkar et al., 2022), the idea is to add edges in order to address over-squashing. However, to reduce the oversmoothing that this may cause, they propose a relational GNNs framework in order to attribute different weights to the added edges. Since we use a GCN in our experiments, in this framework the layers of a R-GCN is given by:

$$h_v^{(k+1)} = \sigma\Bigg(W^{(k)}h_v^{(k)} + \sum_{(u,v)\in E_1} \frac{1}{c_{u,v}} W_1^{(k)} h_u^{(k)} + \sum_{(u,v)\in E_2} \frac{1}{c_{u,v}} W_2^{(k)} h_u^{(k)}\Bigg)$$

where $G = (V, E_1)$ is the original graph before rewiring, and $G' = (V, E_1 \cup E_2)$ is the rewired graph, where additional edges are added. In addition, $h_v^{(k)}$ denotes the hidden representation of node v at layer k, and $W_1^{(k)}$, $W_2^{(k)}$ are the weight matrices associated with edges of type 1 (initial edges) and type 2 (added edges), respectively.

Theorem 3 in (Karhadkar et al., 2022) proves that adding separate weights, as described above, allows the network greater flexibility in learning an appropriate rate of smoothing by leveraging the Dirichlet energy, which quantifies how non-smooth a graph is. In the appendix, we provide the necessary definitions to understand the rate of smoothing and the Dirichlet energy, along with Theorem 3.

As for their rewiring algorithm, FoSR adds edges in the input graph relying on the following proposition in order to compute a first-order approximation of the spectral gap.

Proposition 7.1. *The first-order change in* $\lambda_2 =$

$\lambda_2(D^{-1/2}AD^{-1/2})$ *from adding the edge* (u, v) *is*

$$\frac{2x_u x_v}{(\sqrt{1+d_u})(\sqrt{1+d_v})} + 2\lambda_2 x_u^2 \left(\frac{\sqrt{d_u}}{\sqrt{1+d_u}} - 1 \right) + 2\lambda_2 x_v^2 \left(\frac{\sqrt{d_v}}{\sqrt{1+d_v}} - 1 \right),$$

where λ_2 *is as denoted in (Karhadkar et al., 2022) the spectral gap of the graph, and* x *denotes the second eigenvector (*λ_2*'s eigenvector) of* $D^{-1/2}AD^{-1/2}$*, and* x_u *denotes the* u*-th entry of* x*.*

For computational reasons, the authors choose to minimize only the first term of the above expression to determine which edge to add at each iteration. In summary, FoSR computes an approximation of the second eigenvector and chooses the edge (u, v) that minimizes this first term. The approximation of the second eigenvector is obtained by repeatedly applying the following map:

$$x \mapsto D^{-1/2}AD^{-1/2}x - \frac{\langle x, \sqrt{d} \rangle}{2m}\sqrt{d}.$$

where $\sqrt{d}$ denote the entry-wise square root of d, i.e., with components $\sqrt{d}_i$. Recall that $\sqrt{d}$ is the first eigenvector of $D^{-1/2}AD^{-1/2}$.. It is also important to note that x is normalized after applying this map. The algorithm of FoSR is given in the appendix.

7.3. Methods Comparison

Both methods propose using proxies to optimize the spectral gap and determine which edges to add or delete. However, the two papers present different contributions aimed at alleviating over-smoothing. (Jamadandi et al., 2024) leverages Braess' paradox to show that deleting edges can simultaneously alleviate both over-smoothing and over-squashing. On the other hand, (Karhadkar et al., 2022) proposes using different weights in a relational GNN framework for the added edges to mitigate the over-smoothing they may cause. Moreover (Jamadandi et al., 2024) propose two methods for adding/deleting edges using two different criterions: one using the proxy of the spectral gap and the other using Eldan's criterion.

However, in the experiments, only the first framework was available in the code. Therefore, we compare the performance of both approaches: Proxy methods and FoSR. As mentioned earlier, we were only able to evaluate these models on the COCO-SP and Photo datasets. FoSR outperforms both methods in the COCO-SP node classification task, while ProxyDelete achieves the best performance in the Photo experiment.

For long-range datasets, the approximation used in FoSR can be more efficient. However, this is not always the case, as in the experiments presented in the reference paper, Proxy methods outperform the FoSR approach. In the Photo experiment, the proxy used in ProxyDelete appeared to be more suitable, as it led to a smaller increase in the spectral gap.

We could not test FoSR on the heterophilic dataset due to the same computational limitations as ProxyAdd, since the space of possible edges to add is significantly larger than that of edges that can be deleted. As a result, ProxyDelete appears to be more computationally efficient than these approaches. This highlights the key contribution of our reference paper: leveraging edge deletions to increase the spectral gap while reducing the computational cost of rewiring. However, the effectiveness of this approach remains dependent on the structure of the graph.

8. Conclusion

Our reference paper leverages not only edge addition but also edge deletion to simultaneously address both over-squashing and over-smoothing, rather than treating them as opposing problems, as often proposed in the literature. It introduces four algorithms based on a proxy of the new spectral gap and Eldan's criterion to determine which edges to add or delete.

We compare this approach with FoSR, which also optimizes the spectral gap but selects edges to add using a first-order approximation. The proxy-based methods demonstrate competitive performance compared to FoSR and other baseline methods in both the paper's experiments and ours. However, one could explore combining both edge additions and deletions simultaneously using a unified criterion, whether it be Eldan's, the proxies, or even the first-order approximation used in FoSR.

References

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Jamadandi, A., Rubio-Madrigal, C., and Burkholz, R. Spectral graph pruning against over-squashing and over-smoothing. *arXiv preprint arXiv:2404.04612*, 2024.

Karhadkar, K., Banerjee, P. K., and Montúfar, G. Fosr: First-order spectral rewiring for addressing oversquashing in gnns. *arXiv preprint arXiv:2210.11790*, 2022.

A. Appendix: Braess Paradox

Lemma A.1. *[2]: Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a finite graph, with f denoting the eigenvector and $\lambda_1(\mathcal{L}_\mathcal{G})$ the eigenvalue corresponding to the spectral gap. Let $\{u, v\} \notin \mathcal{V}$ be two vertices that are not connected by an edge. Denote $\hat{\mathcal{G}} = (\mathcal{V}, \hat{\mathcal{E}})$, the new graph obtained after adding an edge between $\{u, v\}$, i.e., $\hat{\mathcal{E}} := \mathcal{E} \cup \{u, v\}$. Denote with $\mathcal{P}_f := \langle f, \hat{f}_0 \rangle$ the projection of f onto the top eigenvector of $\hat{\mathcal{G}}$. Define*

$$g\left(u, v, \mathcal{L}_\mathcal{G}\right) := -\mathcal{P}_f^2 \lambda_1(\mathcal{L}_\mathcal{G}) - 2(1 - \lambda_1(\mathcal{L}_\mathcal{G})) \left(\frac{\sqrt{d_u + 1} - \sqrt{d_u}}{\sqrt{d_u + 1}} f_u^2 + \frac{\sqrt{d_v + 1} - \sqrt{d_v}}{\sqrt{d_v + 1}} f_v^2 \right) + \frac{2 f_u f_v}{\sqrt{d_u + 1}\sqrt{d_v + 1}}.$$

If $g\left(u, v, \mathcal{L}_\mathcal{G}\right) > 0$, then $\lambda_1(\mathcal{L}_\mathcal{G}) > \lambda_1(\mathcal{L}_{\hat{\mathcal{G}}})$.

B. Algorithms

Here we include the corresponding algorithms: PROXYDELETE, PROXYADD, ELDANADD (3), and ELDANDELETE .

Algorithm 1 Proxy Spectral Gap based Greedy Graph Sparsification (PROXYDELETE)

```
Require: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, num. edges to prune $N$, spectral gap $\lambda_1(\mathcal{L}_\mathcal{G})$, second eigenvector $f$.
  repeat
    for $(u, v) \in \mathcal{E}$ do
      Consider $\hat{\mathcal{G}} = \mathcal{G} \setminus (u, v)$.
      Calculate proxy value for the spectral gap of $\hat{\mathcal{G}}$:
      $\lambda_1(\mathcal{L}_{\hat{\mathcal{G}}}) \approx \lambda_1(\mathcal{L}_\mathcal{G}) - ((f_u - f_v)^2 - \lambda_1(\mathcal{L}_\mathcal{G}) \cdot (f_u^2 + f_v^2))$
    end for
    Find the edge that maximizes the proxy: $(u^-, v^-) = \arg\max_{(u,v) \in \mathcal{E}} \lambda_1(\mathcal{L}_{\hat{\mathcal{G}}})$.
    Update graph edges: $\mathcal{E} = \mathcal{E} \setminus (u^-, v^-)$.
    Update degrees: $d_{u^-} = d_{u^-} - 1, d_{v^-} = d_{v^-} - 1$
    Update eigenvectors and eigenvalues of $\mathcal{G}$ accordingly.
  until $N$ edges deleted.
  Output : Sparse graph $\hat{\mathcal{G}} = (\mathcal{V}, \hat{\mathcal{E}})$.
```

Algorithm 2 Proxy Spectral Gap based Greedy Graph Addition (PROXYADD)

```
Require: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, num. edges to add $N$, spectral gap $\lambda_1(\mathcal{L}_\mathcal{G})$, second eigenvector $f$ of $\mathcal{G}$.
  repeat
    for $(u, v) \in \bar{\mathcal{E}}$ do
      Consider $\hat{\mathcal{G}} = \mathcal{G} \cup (u, v)$.
      Calculate proxy value for the spectral gap of $\hat{\mathcal{G}}$ :
      $\lambda_1(\mathcal{L}_{\hat{\mathcal{G}}}) \approx \lambda_1(\mathcal{L}_\mathcal{G}) + ((f_u - f_v)^2 - \lambda_1(\mathcal{L}_\mathcal{G}) \cdot (f_u^2 + f_v^2))$
    end for
    Find the edge that maximizes the proxy: $(u^+, v^+) = \arg\max_{(u,v) \in \bar{\mathcal{E}}} \lambda_1(\mathcal{L}_{\hat{\mathcal{G}}})$.
    Update graph edges: $\mathcal{E} = \mathcal{E} \cup (u^+, v^+)$.
    Update degrees: $d_{u^+} = d_{u^+} + 1, d_{v^+} = d_{v^+} + 1$
    Update eigenvectors and eigenvalues of $\mathcal{G}$ accordingly.
  until $N$ edges added.
  Output : Denser graph $\hat{\mathcal{G}} = (\mathcal{V}, \hat{\mathcal{E}})$.
```

Algorithm 3 Eldan based Greedy Graph Addition (ELDANADD)

```
Require: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, num. edges to add $N$, spectral gap $\lambda_1(\mathcal{L}_{\mathcal{G}})$, top eigenvector $f$ of $\mathcal{G}$.
  repeat
    for $edges(u, v) \in \bar{\mathcal{E}}$ do
      Consider $\hat{\mathcal{G}} = \mathcal{G} \cup (u, v)$.
      Compute projection $\mathcal{P}_f^2 = \langle f, \hat{f}_0 \rangle$.
      Compute Eldan's criterion $g(u, v, \mathcal{L}_{\mathcal{G}})$.
    end for
    Find the edge that minimizes the criterion: $(u^+, v^+) = \arg\max_{(u,v) \in \bar{\mathcal{E}}} -g(u, v, \mathcal{L}_{\mathcal{G}})$.
    $\mathcal{E} = \mathcal{E} \cup (u^+, v^+)$.
    Update degrees $d_{u^+} = d_{u^+} + 1, d_{v^+} = d_{v^+} + 1$
    Update eigenvectors and eigenvalues of $\mathcal{G}$ accordingly.
  until $N$ edges added.
  Output : Denser graph $\hat{\mathcal{G}} = (\mathcal{V}, \hat{\mathcal{E}})$.
```

Algorithm 4 Eldan based Greedy Graph Sparsification (ELDANDELETE)

```
Require: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, num. edges to prune $N$, spectral gap $\lambda_1(\mathcal{L}_{\mathcal{G}})$, top eigenvector $f$ of $\mathcal{G}$.
  repeat
    for $edges(u, v) \in \mathcal{E}$ do
      Consider $\hat{\mathcal{G}} = \mathcal{G} \setminus (u, v)$.
      {Note that the denser graph is the original $\mathcal{G}$, so we require approximations of $\hat{f}$ and $\lambda_1(\mathcal{L}_{\hat{\mathcal{G}}})$ from the sparser $\hat{\mathcal{G}}$.}
      Estimate eigenvector $\hat{f}$ from $f$ based on the power iteration method.
      Estimate corresponding eigenvalue $\lambda_1(\mathcal{L}_{\hat{\mathcal{G}}})$.
      Compute projection $\mathcal{P}_f^2 = \langle \hat{f}, f_0 \rangle$.
      Compute Eldan's criterion $g(u, v, \mathcal{L}_{\hat{\mathcal{G}}})$.
    end for
    Find the edge that maximizes the criterion: $(u^-, v^-) = \arg\max_{(u,v) \in \mathcal{E}} g(u, v, \mathcal{L}_{\hat{\mathcal{G}}})$
    $\hat{\mathcal{E}} = \hat{\mathcal{E}} \setminus (u^-, v^-)$.
    Update degrees $d_{u^-} = d_{u^-} - 1, d_{v^-} = d_{v^-} - 1$
    Update eigenvectors and eigenvalues of $\mathcal{G}$ accordingly.
  until $N$ edges deleted.
  Output : Sparse graph $\hat{\mathcal{G}} = (\mathcal{V}, \hat{\mathcal{E}})$.
```

C. Appendix: Relational Rewiring against Oversmoothing

Before giving the theorem previously discussed in Section 2, we introduce some notions quantifying smoothing in the graph framework.

Definition C.1. Let G be a connected graph with adjacency matrix A and normalized Laplacian L. For $i \in V$, let d_i denote the degree of node i. Given a scalar field $f \in \mathbb{R}^n$, its Dirichlet energy with respect to G is defined as

$$E(f) := \frac{1}{2} \sum_{i,j} A_{i,j} \left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2 = f^T L f.$$

For a vector field $X \in \mathbb{R}^{n \times p}$, we define

$$E(X) := \frac{1}{2} \sum_{i,j,k} A_{i,j} \left(\frac{X_{i,k}}{\sqrt{d_i}} - \frac{X_{j,k}}{\sqrt{d_j}} \right)^2 = \mathrm{Tr}(X^T L X).$$

Definition C.2. Let $\mathcal{G}$ be a graph and $\varphi : \mathbb{R}^{n\times p} \to \mathbb{R}^{n\times p}$ be a mapping. We define the *rate of smoothing* of φ with respect to $\mathcal{G}$ as

$$RS_{\mathcal{G}}(\varphi) = 1 - \left(\frac{\sup_{X:E(X)\neq 0} E(\varphi(X))/E(X)}{\sup_{X:X\neq 0} \|\varphi(X)\|_F^2/\|X\|_F^2}\right)^{1/2}$$
$$= 1 - \left(\frac{\sup_{E(X)=1} E(\varphi(X))}{\sup_{\|X\|_F=1} \|\varphi(X)\|_F^2}\right)^{1/2}.$$

The following theorem proves that the R-GCN framework defined is flexible in choosing the appropriate rate of smoothing, and thus can overcome over-smoothing.

Theorem C.3. *Let $G_1 = (V, E_1)$ be a graph and $G_2 = (V, E_1 \cup E_2)$ be a rewiring of G_1. Consider an R-GCN layer φ, with relations $r_1 = E_1$, $r_2 = E_2$. Then for any $\lambda \in [0, \lambda_2(L(G_2))]$, there exist values of $\Theta, \Theta_1, \Theta_2$ for which φ smooths with rate $RS_{G_2}(\varphi) = \lambda$ with respect to $\mathcal{G}_2$.*

D. Appendix B: FoSR's algorithm

Algorithm 5 First-order Spectral Rewiring (FoSR)

```
Input: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, iteration count $k$, initial number of power iterations $r$
Output: Rewired graph $\mathcal{G}' = (\mathcal{V}, \mathcal{E}')$
Initialize $x \in \mathbb{R}^n$ arbitrarily
for $i = 1$ to $r$ do
    $x \leftarrow D^{-1/2}AD^{-1/2}x - \frac{\langle x, \sqrt{d}\rangle}{2m}\sqrt{d}$ {Approximate second eigenvector before rewiring}
    $x \leftarrow \frac{x}{\|x\|_2}$
end for
for $i = 1$ to $k$ do
    Add edge $(i, j)$ which minimizes $\frac{x_i x_j}{\sqrt{(1+d_i)(1+d_j)}}$
    $x \leftarrow D^{-1/2}AD^{-1/2}x - \frac{\langle x, \sqrt{d}\rangle}{2m}\sqrt{d}$ {Power iteration to update second eigenvector}
    $x \leftarrow \frac{x}{\|x\|_2}$
end for
```