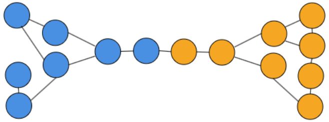
# FoSR: First-order spectral rewiring for addressing oversquashing in GNNs

**Kedar Karhadkar**[*], Pradeep Kr. Banerjee[†], Guido Montúfar[*†]
[*]UCLA, [†]MPI MiS

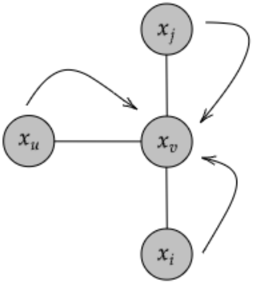
ICLR 2023



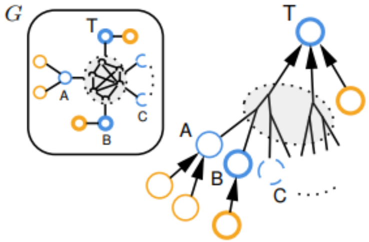
## Graph Neural Networks

- Let $G = (\mathcal{V}, \mathcal{E})$ be a graph.
- Graph neural networks (GNNs) learn node representations for graphs by message passing.
- $x_v^{(k+1)} = f^{(k+1)}(x_v, \{x_u : (u, v) \in \mathcal{E}\})$
- Graph convolutional networks:

$$x_v^{(k+1)} = \sigma\left(\sum_{u \in \mathcal{N}(v) \cup \{v\}} \frac{1}{\sqrt{d_u d_v}} W^{(k)} x_u^{(k)}\right)$$



## GNNs and long-range tasks: NeighborsMatch



- NeighborsMatch task [Alon and Yahav, 2021]: Root node $T$ has some number of orange neighbors. Goal is to identify the leaf node which has the same number of neighbors as $T$.
- Exponentially growing receptive field with a fixed hidden dimension results in *oversquashing*.

## Measuring oversquashing via the Cheeger constant

- How do we measure how extreme the “bottleneck" of a graph is?
- Cheeger constant:

$$h(G) := \min_{\mathcal{S} \subset \mathcal{V} : |\mathcal{S}| \leq n/2} \frac{|\partial \mathcal{S}|}{|\mathcal{S}|},$$

where $\partial S$ consists of the set of edges $(x, y)$ where $x \in S$ and $y \in \mathcal{S}^C$.

## Cheeger cuts

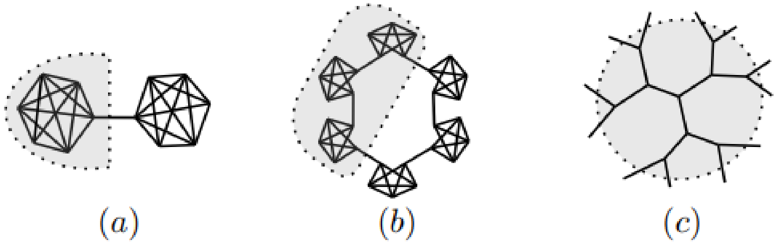


Fig. 2. (a) Dumbbell graph $K_n$–$K_n$, (b) $d$-regular ring-of-cliques comprising of $m$ cliques connected in a ring, (c) Infinite 3-regular tree. The shaded portions specify the Cheeger "cut".

## Measuring oversquashing via the Cheeger constant

- Discrete Cheeger inequality [Cheeger, 1970]:

$$\frac{d - \mu_2}{2} \leq h(G) \leq \sqrt{2d(d - \mu_2)},$$

  where $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n$ are the eigenvalues of $A$, and $d$ is the degree of all nodes of $A$.
- Bounding the Cheeger constant is equivalent to bounding the spectral gap $\lambda_2$ (the second eigenvalue of the Laplacian $L = I - D^{-1/2}AD^{-1/2}$).

## First-order spectral rewiring (FoSR)

- Estimate the change in spectral gap from adding edges.
- Add the edge which maximizes this increase.

**Theorem**

*For symmetric matrices $M \in \mathbb{R}^{n \times n}$ with distinct eigenvalues, the i-th largest eigenvalue $\lambda_i$ satisfies*

$$\nabla_M \lambda_i(M) = x_i x_i^T,$$

*where $x_i$ denotes the (normalized) eigenvector for the i-th largest eigenvalue of $M$.*

## First-order Spectral Rewiring

- This gives us the first-order approximation

$$\lambda_2(M + \delta M) \approx \lambda_2(M) + \mathrm{Trace}(\nabla \lambda_2^T (\delta M)) = \lambda_2(M) + x_2^T (\delta M) x_2.$$

**Theorem**

*The first-order change in $\lambda_2 = \lambda_2(D^{-1/2} A D^{-1/2})$ from adding the edge $(u, v)$ is*

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

*where $x$ denotes the second eigenvector of $D^{-1/2} A D^{-1/2}$, and $x_u$ denotes the $u$-th entry of $x$.*

## First-order spectral rewiring

- FoSR operates by minimizing the dominant term of (2), given by

$$\frac{2x_u x_v}{(\sqrt{1+d_u})(\sqrt{1+d_v})}.$$

- This approximation is most accurate when $d_u$ and $d_v$ are large and comparable in size.
- Alternate between choosing an edge to add which minimizes the above expression, and updating our estimate of $x$ via power iteration.

## FoSR Algorithm

**Algorithm 1 FoSR: First-order Spectral Rewiring**

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

## Relational rewiring

- Adding too many edges can result in *oversmoothing*, resulting in all nodes collapsing to similar features [Li et al., 2018], [Oono and Suzuki, 2020].
- Oversmoothing is caused by a high spectral gap. Comparing to oversquashing, which is caused by a low spectral gap, we see a tradeoff.
- Is there a way to prevent both simultaneously?

## Relational Rewiring

- We can form a generalization for GNNs using *relational rewiring* [Brockschmidt, 2020].
- Idea: let $\mathcal{R}$ be a finite set of *relation types*. We assign each edge a relation type $r \in \mathcal{R}$. In our GNN, we have different learned mappings for each relation type.

$$h_v^{(k+1)} = \phi_k \left( h_v^{(k)}, \textstyle\sum_{r \in \mathcal{R}} \sum_{u \in \mathcal{N}_r(v)} \psi_{k,r}(h_u^{(k)}, h_v^{(k)}) \right),$$

- In other words, we use separate weights for the original graph edges and for the rewired edges.

## R-GNNs prevent oversmoothing

- R-GNNs prevent oversmoothing by de-weighting the rewired edges and increasing the weights of self-loops.
- Given a scalar field $f \in \mathbb{R}^n$, its *Dirichlet energy* with respect to $\mathcal{G}$ is defined as

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

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

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

# R-GNNs prevent oversmoothing

- 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 $\varphi$ with respect to $\mathcal{G}$ as

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

**Theorem**

*Let $\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}_1)$ be a graph and $\mathcal{G}_2 = (\mathcal{V}, \mathcal{E}_1 \cup \mathcal{E}_2)$ be a rewiring of $\mathcal{G}_1$. Consider an R-GCN layer $\varphi$, with relations $r_1 = \mathcal{E}_1$, $r_2 = \mathcal{E}_2$. Then for any $\lambda \in [0, \lambda_2(L(\mathcal{G}_2))]$, there exist values of $\Theta, \Theta_1, \Theta_2$ for which $\varphi$ smooths with rate $RS_{\mathcal{G}_2}(\varphi) = \lambda$ with respect to $\mathcal{G}_2$.*

- This theorem says that R-GNNs can flexibly learn the optimal amount of smoothing, especially with rewired edges.

## Experiments

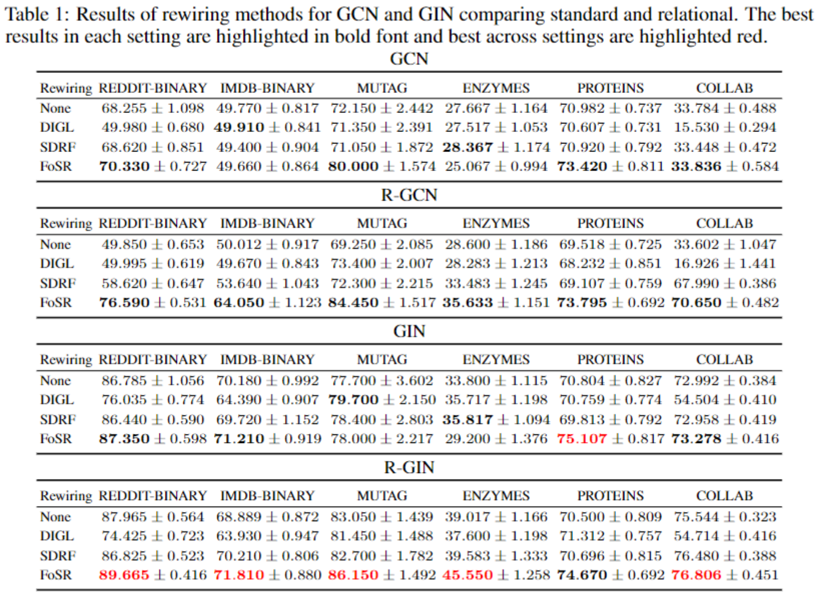


Table 1: Results of rewiring methods for GCN and GIN comparing standard and relational. The best results in each setting are highlighted in bold font and best across settings are highlighted red.

GCN

| Rewiring | REDDIT-BINARY | IMDB-BINARY | MUTAG | ENZYMES | PROTEINS | COLLAB |
|---|---|---|---|---|---|---|
| None | 68.255 ± 1.098 | 49.770 ± 0.817 | 72.150 ± 2.442 | 27.667 ± 1.164 | 70.982 ± 0.737 | 33.784 ± 0.488 |
| DIGL | 49.980 ± 0.680 | **49.910** ± 0.841 | 71.350 ± 2.391 | 27.517 ± 1.053 | 70.607 ± 0.731 | 15.530 ± 0.294 |
| SDRF | 68.620 ± 0.851 | 49.400 ± 0.904 | 71.050 ± 1.872 | **28.367** ± 1.174 | 70.920 ± 0.792 | 33.448 ± 0.472 |
| FoSR | **70.330** ± 0.727 | 49.660 ± 0.864 | **80.000** ± 1.574 | 25.067 ± 0.994 | **73.420** ± 0.811 | **33.836** ± 0.584 |

R-GCN

| Rewiring | REDDIT-BINARY | IMDB-BINARY | MUTAG | ENZYMES | PROTEINS | COLLAB |
|---|---|---|---|---|---|---|
| None | 49.850 ± 0.653 | 50.012 ± 0.917 | 69.250 ± 2.085 | 28.600 ± 1.186 | 69.518 ± 0.725 | 33.602 ± 1.047 |
| DIGL | 49.995 ± 0.619 | 49.670 ± 0.843 | 73.400 ± 2.007 | 28.283 ± 1.213 | 68.232 ± 0.851 | 16.926 ± 1.441 |
| SDRF | 58.620 ± 0.647 | 53.640 ± 1.043 | 72.300 ± 2.215 | 33.483 ± 1.245 | 69.107 ± 0.759 | 67.990 ± 0.386 |
| FoSR | **76.590** ± 0.531 | **64.050** ± 1.123 | **84.450** ± 1.517 | **35.633** ± 1.151 | **73.795** ± 0.692 | **70.650** ± 0.482 |

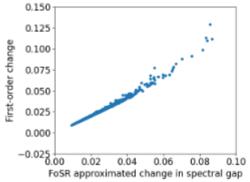
GIN

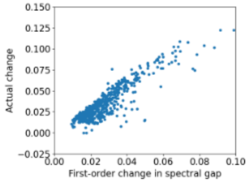
| Rewiring | REDDIT-BINARY | IMDB-BINARY | MUTAG | ENZYMES | PROTEINS | COLLAB |
|---|---|---|---|---|---|---|
| None | 86.785 ± 1.056 | 70.180 ± 0.992 | 77.700 ± 3.602 | 33.800 ± 1.115 | 70.804 ± 0.827 | 72.992 ± 0.384 |
| DIGL | 76.035 ± 0.774 | 64.390 ± 0.907 | **79.700** ± 2.150 | 35.717 ± 1.198 | 70.759 ± 0.774 | 54.504 ± 0.410 |
| SDRF | 86.440 ± 0.590 | 69.720 ± 1.152 | 78.400 ± 2.803 | **35.817** ± 1.094 | 69.813 ± 0.792 | 72.958 ± 0.419 |
| FoSR | **87.350** ± 0.598 | **71.210** ± 0.919 | 78.000 ± 2.217 | 29.200 ± 1.376 | **75.107** ± 0.817 | **73.278** ± 0.416 |

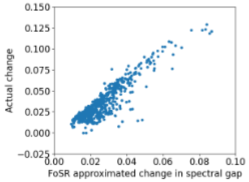
R-GIN

| Rewiring | REDDIT-BINARY | IMDB-BINARY | MUTAG | ENZYMES | PROTEINS | COLLAB |
|---|---|---|---|---|---|---|
| None | 87.965 ± 0.564 | 68.889 ± 0.872 | 83.050 ± 1.439 | 39.017 ± 1.166 | 70.500 ± 0.809 | 75.544 ± 0.323 |
| DIGL | 74.425 ± 0.723 | 63.930 ± 0.947 | 81.450 ± 1.488 | 37.600 ± 1.198 | 71.312 ± 0.757 | 54.714 ± 0.416 |
| SDRF | 86.825 ± 0.523 | 70.210 ± 0.806 | 82.700 ± 1.782 | 39.583 ± 1.333 | 70.696 ± 0.815 | 76.480 ± 0.388 |
| FoSR | **89.665** ± 0.416 | **71.810** ± 0.880 | **86.150** ± 1.492 | **45.550** ± 1.258 | **74.670** ± 0.692 | **76.806** ± 0.451 |

# Experiments

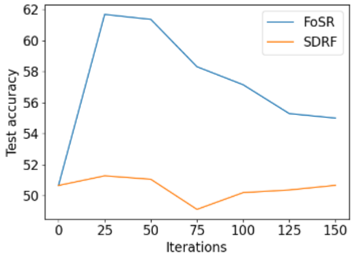


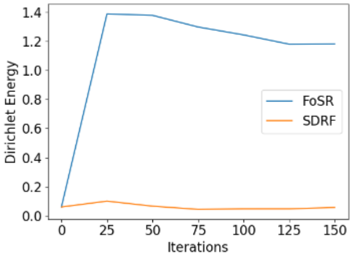




- The FoSR algorithm makes two approximations: one to get the first-order change, and another to select the dominant term. We record all three here for the ENZYMES dataset.

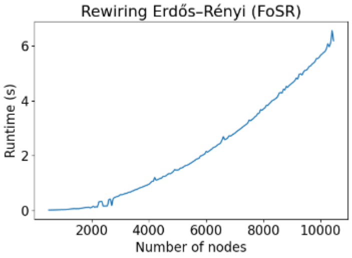
## Experiments

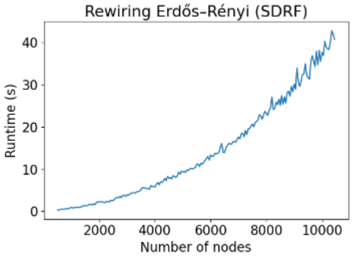




- Tradeoff between oversquashing and oversmoothing; the GNN smooths more as more rewiring edges are added.
- Dirichlet energy closely matches test accuracy in its peak.

# Experiments





- Rewiring for Erdös-Reny graphs with edge probability $p = \frac{5 \log n}{n}$.
- Computational complexity for adding a single edge with FoSR is $O(m)$ typically and $O(n^2)$ in the worst case. For SDRF, it is $O(md^3)$, where $d$ is the maximal node degree.

Alon, U. and Yahav, E. (2021).
On the bottleneck of graph neural networks and its practical implications.
In *International Conference on Learning Representations*.

Brockschmidt, M. (2020).
GNN-film: Graph neural networks with feature-wise linear modulation.
In *International Conference on Machine Learning*, pages 1144–1152. PMLR.

Cheeger, J. (1970).
A lower bound for the smallest eigenvalue of the Laplacian.
*Problems in Analysis*, 625(195-199):110.

Li, Q., Han, Z., and Wu, X.-M. (2018).
Deeper insights into graph convolutional networks for semi-supervised learning.
In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Oono, K. and Suzuki, T. (2020).
Graph neural networks exponentially lose expressive power for node classification.
In *International Conference on Learning Representations*.