Extended Abstract Track

RelWire: Metric Based Graph Rewiring

Editors: List of editors' names

Abstract

Oversquashing is a major hurdle to the application of geometric deep learning and graph neural networks to real applications. Recent work has found connections between oversquashing and commute times, effective resistance, and the eigengap of the underlying graph. Graph rewiring is the most promising technique to alleviate this issue. Some prior work adds edges locally to highly negatively curved subgraphs. These local changes, however, have a small effect on global statistics such as commute times and the eigengap. Other prior work uses the spectrum of the graph Laplacian to target rewiring to increase the eigengap. These approaches, however, make large structural and topological changes to the underlying graph. We use ideas from geometric group theory to present RelWire, a rewiring technique based on the geometry of the graph. We derive topological connections for RelWire. We then rewire different real world molecule datasets and show that RelWire is Pareto optimal: it has the best balance between improvement in eigengap and commute times and minimizing changes in the topology of the underlying graph.

1. Introduction

Graph neural networks (GNNs) are a promising generalization of (deep) neural networks based upon the premise that many real world data sets exhibit graphical structure. That is, data points are related to one another in complex ways that are best captured by relations on a graph, with vertices being the data points and edges the relations between those points. Hence, GNNs are methods for aggregating information across the relation graph and then propagating those updates. All of these methods, however, seem to suffer from two structural problems: oversquashing and over-smoothing.

Oversquashing has been a challenging problem to define. The problem was initially observed in [1], where the authors setup the neighbors match problem whose task is to match a target subgraph with a template subgraph. They noticed that the feature vectors cannot store enough information, a problem which they called oversquashing. However, since then oversquashing has been defined in a different manner. Influenced by [38], who defined the *influence* of a node on another as the magnitude of the relative derivative, the perspective on oversquashing has shifted. Since then a variety of papers [8, 14, 26, 34] have defined oversquashing to be the problem of having nodes having small influence on each other (i.e., small Jacobian), including theoretical connections between oversquashing and structural properties like commute time [14] and eigengaps.

As a result of these deficiencies, many recent works have introduced the notion of "rewiring" a graph, adding edges or relations amongst the data points so as to improve the performance of GNNs. These methods leverage notions from spectral graph theory (e.g., effective resistance, spectral or eigengap), discrete graph geometry (e.g., graph curvature), and random walks (via commute time). All of these methods make large structural changes to the underlying graph, some more effective than others, which are aimed at affecting one or more of the above quantities.

In this paper, we present a new graph rewiring regime by importing techniques from coarse geometry. We use the geometry of the underlying graph to define two relations between the graph's edges, one which roughly encodes local negative curvature and the other flat curvature. Our rewiring algorithm, called RelWire, utilizes this notion of curvature to rewire the underlying graph by targeting certain structural features while minimally affecting the topology of the underlying graph. We develop a framework for analyzing this balancing act of effective rewiring with minimal topological disturbance.

Contributions The main contributions of this paper are as follows:

- We define a new topological distance called the rank distance that can be used to measure the structural changes to a graph after rewiring.
- We present a new method for rewiring graphs RelWire. Our method uses new ideas and concepts that have not been applied to the field of geometric deep learning before. Specifically, it introduces a global notion of curvature.
- We present topological differences between RelWire and prior rewiring techniques.
- We extensively test on real data to show that RelWire is Pareto optimal for the graph statistics. That is, it performs the best at improving eigengap and commute times, while simultaneously preserving the graph topology. We also show that it helps for graph regression.

1.1. Background and Problem Setup

Prior work has shown that the norm of Jacobian (which controls oversquashing) can be bounded by various different graph properties such as the spectral gap, the commute times, the curvature, and the effect resistance of the graph. However, the initial motivation of GNNs was that the structure of the graph had important information. The process of graph rewiring changes this structure. Hence we are interested in quantifying this change and keeping it to a minimum.

In this paper, we are interested in the problem of graph rewiring. That is given a graph G, we want to add k edges to the graph to improve various graph statistics mentioned above while preserving as much structural information. To do this, in the following section we develop the relevant background. We start by setting up notation for the paper. Throughout the paper $G = (V, E)$ will refer to a graph on vertex set V with edges E. We shall have that G has n nodes and m edges. Let A denote the adjacency matrix of the graph and let D denote the degree matrix of the graph.

2. Capturing topological distortion: Distances from persistent homology

In this paper, we consider two notions of distances between graphs using topological information. The first is based on comparing 1-dimensional information, which is already quite powerful in the context of graphs. The second is based on techniques from topological data analysis, which takes into account higher dimensional features of the graphs. This latter machinery is called *persistence homology* (see e.g., [30]), as it attempts to capture "persistent" homological features as one takes larger samples of the space.

In this paper, we consider the following two notions of topological distancces. More details can be found in Appendix D. The *persistence distance* between two persistence diagrams is the L_2 norm of the difference between their respective Betti curves. Given two graphs G, G', we will call the persistence distance between their respective Vietoris-Rips filtrations $\{VR_r(G)\}_{r\in\mathbb{R}_+}$ and $\{VR_r(G')\}_{r\in\mathbb{R}_+}$ the *Betti distance* between G, G'.

We use the Betti and rank distances to measure of how much a given rewiring procedure changes the topology of the graph. *For GNNs, the structure of the base graph encodes crucial information. Hence we can use these two distances to measure how much a given graph rewiring procedure preserves the underlying graph structure.* We explore the topological differences between our method and prior work in Appendix F.

3. RelWire: relations on graphs

We draw inspiration from the notion of Hierarchical Hyperbolic Spaces (HHSes) from coarse geometry. More details on the connection can be found in Appendix E. In our setting, the ambient space $X = G$ is a simplicial graph and this philosophy becomes quite simple: the spaces in the hierarchy are the edges, and a projection of a vertex of G to an edge E is a some collection of its endpoints. Specifically, given a vertex $v \in G^{(0)}$ and an edge E of G, the *projection* $\pi_E(v) \subset E^{(0)}$ of v to E is the endpoint of E which is closest in G to v. When both endpoints are equidistant to v, then we set $\pi_E(v) = E^{(0)}$ to be both endpoints. With these projections defined, we define our (simplified) relations.

The first relation, called *orthogonal*, encodes flat curvature. In the setting of an HHS X, when two hyperbolic spaces U, V in the hierarchy are orthogonal, the product map $\pi_U \times \pi_V : X \to U \times V$ is surjective, and there is a coarsely isometrically embedded flat subspace of X (see e.g. Subsection 5B of [5]). For instance, $\mathbb{R}^2$ is an HHS where the hyperbolic spaces are the coordinate axes (i.e., copies of $\mathbb{R}$), and the flat subspace corresponding to their product is the whole ambient space $\mathbb{R} \times \mathbb{R} = \mathbb{R}^2$. In our graphical setting, we will say two edges E_1, E_2 are *independent* when $\pi_{E_1} \times \pi_{E_2} : G \to E_1^{(0)} \times E_2^{(0)}$ is surjective. Otherwise, we will say that E_1, E_2 are *transverse*. Roughly, this notion of transversality encodes negative curvature (see e.g. [7]). While the connection between independence/transversality and flat/negative curvature is not exact, the connection is more than moral:

Lemma 1 *Let E_1, E_2 be edges of a simplicial graph G. If for E_1, E_2 we have that they*
1. *are contained in a clique subgraph of G, then E_1, E_2 are independent;*
2. *are separated by vertex v with $\pi_{E_1}(v)$, $\pi_{E_2}(v)$ both singletons, then E_1, E_2 are transverse.*

As every pair of edges in a graph satisfies one of our two relations, they are both capable of capturing local and global properties of the graph. While item (2) of Lemma 15 says that independence is frequently more locally focused, transversality captures negative curvature in a fundamentally different way than existing notions of graph curvature.

3.1. RelWire

We present our new rewiring technique RelWire. The pseudocode for the method can be seen in Algorithm 2. The basic idea is to eliminate negative curvature by adding edges, so the main task is to identify pairs of vertices which belong to the most transverse edge pairs, weighted by their distance in the graph. We begin by determining for each pair of edges if they are independent or transverse. Then for each pair of nodes u, a, we consider all neighboring edges of the form (u, v) and (a, b) for all $v \in \mathcal{N}(u)$ and $b \in \mathcal{N}(a)$. Then we define $r(u,a) = d(u,a)\sum_{\substack{v\in\mathcal{N}(u)\\ b\in\mathcal{N}(a)}} \mathbb{1}\{(u,v) \not\perp (a,b)\}$. We then connect the k node pairs that are not adjacent in the graph that have the highest r values, where k is our rewiring parameter. That is, for a pair of nodes, we count the number of transverse edge pairs that the two nodes are in. We then weight this count by the distance between the two nodes, and connect the pairs of nodes with the highest weighted r value.

As discussed above, tranversality captures some notion of negative curvature, at both the local and global scale of the graph. Hence connecting a highly transverse distant pair morally helps remove negative curvature at a global level.

Algorithm 1 RelWire

```
1: function RelWire($G$ - Graph, $k$ - number of edges added)
2:     Compute shortest distance $d(u,v)$ between all pairs of nodes $u, v$.
3:     Compute $T : E \times E \to \{0,1\}$ such that $T(e_1,e_2) = 1$ if and only if $e_1 \not\perp e_2$.
4:     Compute $r(u,a) = d(u,a)\sum_{v\in\mathcal{N}(u)}\sum_{b\in\mathcal{N}(a)} T\big((u,v),(a,b)\big)$.
5:     Connect the $k$ non-adjacent node pairs with largest $r$ value.
6:     return Rewired Graph
7: end function
```

4. Experiments

To validate our method on real data[1], we took ten different datasets with roughly $\sim 27,000$ graphs for rewiring (see the Appendix B for more details). We rewired these datasets by adding three edges using RelWire, FOSR, GTR, and SDRF. We then computed the spectral gap for each of the graphs in the dataset and took the average spectral gap for each dataset. Similarly, we computed the average commute times for each of the graphs and then averaged that as well. Note that for three of the datasets (Lipo, Tox21, and Enzymes) we had disconnected graphs post rewiring, hence we did not compute the commute times for these datasets. Finally, we computed the Betti and rank distances. Table 2 shows the spectral gap and the Betti distance, while Table 4 displays the commute times. This is also visualized in Figure 1.

Here we can see there is a give-and-take between the eigengap and commute times with the Betti distance. In particular, GTR greatly decreases the eigengap and commute times at the expense of transforming the topology, as measured by large Betti distance. On the other hand, SDRF relatively preserves the topology as well as the eigengap and commute times. Hence if we are to reduce oversquashing while preserving the graph, we must find a balance. In this regard, we see that RelWire is Pareto optimal, in that we have the second best eigengap, commute times, and Betti distance.

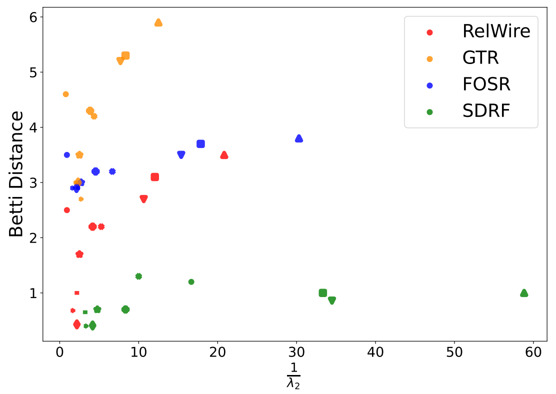


Figure 1: Betti distance versus $1/\lambda_2$. The closer the points are to the origin the better.

1. All code can be found anonymized at Github

5. Conclusion

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. URL: https://openreview.net/forum?id=i80OPhOCVH2 (cit. on p. 1).

[2] Adrian Arnaiz-Rodriguez et al. "DiffWire: Inductive Graph Rewiring via the Lovasz Bound". In: *The First Learning on Graphs Conference*. 2022. URL: https://openreview.net/pdf?id=IXvfIex0mX6f (cit. on p. 11).

[3] Yunsheng Bai et al. "Simgnn: A neural network approach to fast graph similarity computation". In: *Proceedings of the twelfth ACM international conference on web search and data mining*. 2019, pp. 384–392 (cit. on p. 9).

[4] Ulrich Bauer and Michael Lesnick. "Induced Matchings of Barcodes and the Algebraic Stability of Persistence". In: *Proceedings of the Thirtieth Annual Symposium on Computational Geometry*. SOCG'14. Kyoto, Japan: Association for Computing Machinery, 2014, pp. 355–364. ISBN: 9781450325943. DOI: 10.1145/2582112.2582168. URL: https://doi.org/10.1145/2582112.2582168 (cit. on p. 13).

[5] Jason Behrstock, Mark Hagen, and Alessandro Sisto. "Hierarchically hyperbolic spaces II: Combination theorems and the distance formula". In: *Pacific Journal of Mathematics* 299.2 (2019), pp. 257–338 (cit. on pp. 3, 15).

[6] Jason Behrstock, Mark Hagen, and Alessandro Sisto. "Hierarchically hyperbolic spaces, I: Curve complexes for cubical groups". In: *Geometry & Topology* 21.3 (2017), pp. 1731–1804 (cit. on p. 14).

[7] Mladen Bestvina, Ken Bromberg, and Koji Fujiwara. "Constructing group actions on quasi-trees and applications to mapping class groups". In: *Publications mathématiques de l'IHÉS* 122.1 (2015), pp. 1–64 (cit. on pp. 3, 15).

[8] Mitchell Black et al. "Understanding oversquashing in gnns through the lens of effective resistance". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 2528–2547 (cit. on pp. 1, 9).

[9] Karsten M. Borgwardt et al. "Protein Function Prediction via Graph Kernels". In: 21.1 (Jan. 2005), pp. 47–56. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bti1007. URL: https://doi.org/10.1093/bioinformatics/bti1007 (cit. on p. 9).

[10] Jeffrey Brock. "The Weil-Petersson metric and volumes of 3-dimensional hyperbolic convex cores". In: *Journal of the American Mathematical Society* 16.3 (2003), pp. 495–535 (cit. on p. 14).

[11] Jeffrey F Brock, Richard D Canary, and Yair N Minsky. "The classification of Kleinian surface groups, II: The ending lamination conjecture". In: *Annals of Mathematics* (2012), pp. 1–149 (cit. on p. 14).

[12] Asim Kumar Debnath et al. "Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity." In: *Journal of medicinal chemistry* 34 2 (1991), pp. 786–97. URL: https://api.semanticscholar.org/CorpusID:19990980 (cit. on p. 9).

[13] Francesco Di Giovanni et al. "How does over-squashing affect the power of GNNs?" In: *arXiv preprint arXiv:2306.03589* (2023) (cit. on p. 16).

[14] Francesco Di Giovanni et al. "On Over-Squashing in Message Passing Neural Networks: The Impact of Width, Depth, and Topology". In: *Proceedings of the 40th International Conference on Machine Learning*. Ed. by Andreas Krause et al. Vol. 202. Proceedings of Machine Learning Research. PMLR, 23–29 Jul 2023, pp. 7865–7885 (cit. on pp. 1, 16).

[15] Matthew Gentry Durham. "The augmented marking complex of a surface". In: *Journal of the London Mathematical Society* 94.3 (2016), pp. 933–969 (cit. on p. 14).

[16] Matthias Fey and Jan E. Lenssen. "Fast Graph Representation Learning with PyTorch Geometric". In: *ICLR Workshop on Representation Learning on Graphs and Manifolds*. 2019 (cit. on p. 11).

[17] Rafael Gómez-Bombarelli et al. "Automatic chemical design using a data-driven continuous representation of molecules". In: *ACS central science* 4.2 (2018), pp. 268–276 (cit. on p. 9).

[18] Mikhael Gromov. "Hyperbolic groups". In: *Essays in group theory*. Springer, 1987, pp. 75–263 (cit. on p. 14).

[19] Max Horn et al. "Topological Graph Neural Networks". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=oxxUMeFwEHd (cit. on p. 13).

[20] 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. URL: https://openreview.net/forum?id=3YjQfCLdrzz (cit. on pp. 9, 16).

[21] Michael Lesnick. "The theory of the interleaving distance on multidimensional persistence modules". In: *Foundations of Computational Mathematics* 15.3 (2015), pp. 613–650 (cit. on p. 13).

[22] Sunhyuk Lim, Facundo Memoli, and Osman Berat Okutan. "Vietoris-rips persistent homology, injective metric spaces, and the filling radius". In: *arXiv preprint arXiv:2001.07588* (2020) (cit. on p. 13).

[23] Howard A Masur and Yair N Minsky. "Geometry of the complex of curves II: Hierarchical structure". In: *Geometric and Functional Analysis* 10.4 (2000), pp. 902–974 (cit. on p. 14).

[24] Yair Minsky. "The classification of Kleinian surface groups, I: Models and bounds". In: *Annals of Mathematics* (2010), pp. 1–107 (cit. on p. 14).

[25] Christopher Morris et al. "Tudataset: A collection of benchmark datasets for learning with graphs". In: *arXiv preprint arXiv:2007.08663* (2020) (cit. on p. 9).

[26] Khang Nguyen et al. "Revisiting Over-smoothing and Over-squashing Using Ollivier-Ricci Curvature". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 25956–25979 (cit. on pp. 1, 8).

[27] Yann Ollivier. "Ricci curvature of Markov chains on metric spaces". In: *Journal of Functional Analysis* 256.3 (2009), pp. 810–864 (cit. on p. 8).

[28] Hongbin Pei et al. "Geom-GCN: Geometric Graph Convolutional Networks". In: *ArXiv* abs/2002.05287 (2020). URL: https://api.semanticscholar.org/CorpusID:210843644 (cit. on p. 18).

[29] Kasra Rafi. "Hyperbolicity in Teichmüller space". In: *Geometry & Topology* 18.5 (2014), pp. 3025–3053 (cit. on p. 14).

[30] Bastian Rieck et al. "Neural Persistence: A Complexity Measure for Deep Neural Networks Using Algebraic Topology". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=ByxkijC5FQ (cit. on pp. 2, 12, 13).

[31] Yunsheng Shi et al. "Masked label prediction: Unified message passing model for semi-supervised classification". In: *arXiv preprint arXiv:2009.03509* (2020) (cit. on p. 10).

[32] Rishi Sonthalia and Anna Gilbert. "Tree! i am no tree! i am a low dimensional hyperbolic embedding". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 845–856 (cit. on p. 8).

[33] T. Sterling and John J. Irwin. "ZINC 15 – Ligand Discovery for Everyone". In: *Journal of Chemical Information and Modeling* 55 (2015), pp. 2324–2337. URL: https://api.semanticscholar.org/CorpusID:327319 (cit. on p. 9).

[34] Jake Topping et al. "Understanding over-squashing and bottlenecks on graphs via curvature". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=7UmjRGzp-A (cit. on pp. 1, 8, 9, 16).

[35] Domenico Tortorella and Alessio Micheli. "Is Rewiring Actually Helpful in Graph Neural Networks?" In: *arXiv preprint arXiv:2305.19717* (2023) (cit. on p. 10).

[36] Nicolas Garcia Trillos and Melanie Weber. "Continuum Limits of Ollivier's Ricci Curvature on data clouds: pointwise consistency and global lower bounds". In: *arXiv preprint arXiv:2307.02378* (2023) (cit. on p. 8).

[37] Zhenqin Wu et al. "MoleculeNet: a benchmark for molecular machine learning". In: *Chemical science* 9.2 (2018), pp. 513–530 (cit. on p. 9).

[38] Keyulu Xu et al. "Representation Learning on Graphs with Jumping Knowledge Networks". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, Oct. 2018, pp. 5453–5462 (cit. on p. 1).

Appendix A. Prior

A.1. Graph Properties: Eigengap, commute time, and curvature.

In this section, we detail connections between oversquashing and different statistics.

Definition 2 (Combinatorial Laplacian):
Suppose A is the adjacency matrix for a graph G and D is the degree matrix, the *Combinatorial Laplacian* is $L(G) := D - A$.

Definition 3 (Eigengap):
For a connected graph G, the *eigengap* or spectral gap $\lambda_2(G)$ is the second largest eigenvalue of the Combinatorial Laplacian $L(G)$.

Definition 4 (Cheeger Constant):
Given a graph $G = (V, E)$ a *cut* $cut(C_1, C_2)$ is a disjoint partition of the nodes V into C_1 and C_2. The *size* of a cut is $|cut(C_1, C_2)| := |\{(u, v) \in E, u \in C_1, v \in C_2\}|$. The *Cheeger constant* h_G is defined as $\min_{(C_1,C_2)} \frac{|cut(C_1,C_2)|}{|C_1||C_2|}$.

Thus, we see that the Cheeger constant tells how connected the graph is, giving it a clear relation to oversquashing. However, the Cheeger constant is difficult to compute, but can be well approximated by the eigengap.

$$\frac{\lambda_2(G)}{2} \leq h_G \leq \sqrt{2\lambda_2(G)} \text{ and } 2h_G \leq \lambda_2(G) \leq \frac{h_G^2}{2}.$$

Thus, having a large $\lambda_2(G)$ results in a large Cheeger constant. Thus, the graph is better connected.

Definition 5 (Commute Times):
Let A be the adjacency matrix of a graph G. Consider a random walk on G with transition probabilities given by $P = D^{-1}A$.

1. The *hitting time* $H(i, j)$ is the expected time for a random walk starting at node i to hit node j.
2. The *commute time* is $CT(i, j) = H(i, j) + H(j, i)$.

As discussed in [34], the Ricci curvature is a natural method for measuring information dispersion on a manifold. For instance, two geodesics starting nearby with the same velocity will converge on a sphere (positive curvature), will remain parallel in Euclidean space (zero curvature), or will diverge in Hyperbolic space (negative curvature). This divergence in Hyperbolic space can be used to show that Hyperbolic space is good for representing hierarchical information [32].

Similar to the Ricci curvature on manifolds, Ricci curvature has also been defined for graphs [27] and has been used for rewiring [34]. Recently it has been shown by [36] that the discrete notion of the Ollivier-Ricci curvature defined on metric graphs (i.e. k-nearest neighbor graphs of data on a manifold $\mathcal{M}$) converges pointwise to the Ricci curvature on the manifold $\mathcal{M}$. Thus the information divergence interpretation of the curvature applies to the graphs as well, indicating that graphs with negative curvature are detrimental to oversquashing. This has been formalized by recent work such as [26, Theorem 4.5], where the authors show that negative curvature results in sharply decaying importance of distant nodes. Thus, increasing the curvature of the graphs helps address oversquashing.

A.2. Prior Rewiring Works: SDRF, FOSR, and GTR

We review the existing rewiring methods against which we compare our own method RelWire.

SDRF As we have seen, the curvature of a graph is related to oversquashing. Thus, [34] design a method to increase the curvature of negatively curved areas. However, the Ollivier Ricci curvature is difficult to calculate, hence they approximate it using a notion called Balanced Forman Curvature $Ric(i,j)$. In [34] show that if $Ric(i,j) > k$ for all edges then we have that $\frac{k}{2} \leq h_G \leq \frac{\lambda_2}{2}$. Thus, showing the connection between the curvature and other quantities such as the eigengap and the Cheeger constant. They then create a method that finds the most negatively curved edge and then add the edge that increases the curvature of this edge the most.

FOSR In [20], they showed that if f is the second eigenvector for the normalized Laplacian $(I - D^{-1/2}AD^{-1/2}))$ then adding an edge i, increases the second eigenvalue by

$$\frac{2f_i f_j}{\sqrt{1+d_i}\sqrt{1+d_j}} + 2\lambda_2 \left[f_i^2 \left(\frac{\sqrt{d_i}}{\sqrt{1+d_i}} - 1 \right) + f_j^2 \left(\frac{\sqrt{d_j}}{\sqrt{1+d_j}} - 1 \right) \right].$$

They use this method to design an algorithm FOSR, that maximizes the first order term.

GTR Another notion of relevance is the total resistance of a graph, G. Let L be the Combinatorial Laplacian of a graph. Then the resistance $R(i,j)$ between nodes i and j is given by $R(i,j) = (1_i - 1_j)^T L^\dagger (1_i - 1_j)$. Here 1_i is the indicator vector for the ith node and $L^\dagger$ is the pseudoinverse of the Combinatorial Laplacian. The total resistance R_{tot} is $R_{tot} = \sum_{i,j} R(i,j)$. Then the biharmonic distance $B(i,j)$ between nodes i and j is given by $B(i,j) = \sqrt{(1_i - 1_j)^T (L^\dagger)^2 (1_i - 1_j)}$.

[8] show that the increase in the total resistance of adding an edge (i,j) is given by $\frac{B(i,j)^2}{1+R(i,j)}$. Hence they design a method GTRthat maximizes this quantity. This quantity is related to the eigengap as well [8], where the maximum resistance between any two pairs of nodes R_{max} is bounded by

$$\frac{1}{n\lambda_2} \leq R_{max} \leq \frac{1}{\lambda_2}.$$

Appendix B. Datasets

Specifically, we look at The ZINC dataset [17, 33] which consists of 10,000 molecular graphs. From [37], we look at ESOL which is the water solubility of 1,128 compounds, BACE which 1,522 compounds representing the inhibitors or human β-secretase 1, Lipophilicity which is 4,200 drug compounds, and Tox21 measure the toxicity of 7831 compounds. From the TUDataset [25], we use MUTAG [12] with consists of 188 nitroaromatic compounds and ENZYMES [9] which is a dataset of 600 protein tertiary structures obtained from the BRENDA enzyme database. Finally, we use three datasets from [3] consisting of 1520 graphs in total. The statistics for the datasets can be seen in Table 1.

	Zinc	ESOL	BACE	Lipo	Tox21	Mutag	Enzymes	AIDS	Alkane	Linux
Nodes	23.2	13.3	34.1	27	18.6	17.9	32.6	8.9	8.9	7.6
Edges	49.8	27.4	73.7	59	38.6	39.6	124.3	17.6	15.8	13.9

Table 1: Table showing the average sizes of the graphs in each dataset.

Appendix C. Experiments

C.1. Graph Rewiring

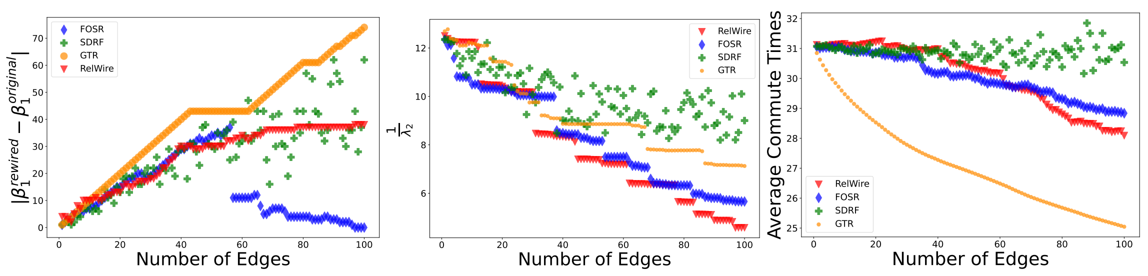
Dataset	Eigengap				Betti Distance			
	RelWire	FOSR	GTR	SDRF	RelWire	FOSR	GTR	SDRF
Zinc	0.094	0.065	0.13	0.029	2.7	3.5	5.2	0.9
ESOL	0.4	0.37	0.4	0.21	1.7	3	3.5	0.7
BACE	0.048	0.033	0.08	0.017	3.5	3.8	5.9	1
Lipophilicity	0.083	0.056	0.12	0.03	3.1	3.7	5.3	1
Tox21	0.24	0.22	0.26	0.12	2.2	3.2	4.3	0.7
Mutag	0.18	0.15	0.23	0.1	2.2	3.2	4.2	1.3
Enzymes	1.1	1.1	1.3	0.06	2.5	3.5	4.6	1.2
AIDS700nef	0.46	0.44	0.49	0.31	1.0	2.9	3	0.65
Alkane	0.46	0.47	0.43	0.24	0.43	2.9	3.0	0.41
Linux	0.6	0.62	0.57	0.3	0.68	2.9	2.7	0.4

Table 2: Table showing the eigengap λ_2 and the Betti distance for the rewired graphs.

C.2. Graph Regression

We also do preliminary experiments to show that RelWire helps improve the performance for graph regression. Four out of the ten data are for graph regression. For each dataset we split into train, test, and validation sets. We train 5 two layer Transformer Convolution networks [31]. We then pick the network with the best validation accuracy and report the test accuracies. These can be see in Table 3. As we see from the Table, we don't have any consistent trends. However, we do see that RelWire does perform well on average. This lack of trends is further supported by [35], where they do node classification tests on different data sets. This suggests that more work needs to be done to understand when graph rewiring is helpful.

Data Split For Zinc we used a random split of 8000 training datapoints, 1000 validation data points and 1000 test datapoints. For BACE we use 1100 training, 200 validation and 222 test datapoints. For Lipo we used 3600 training data points and 300 for validation and test east. Finally, for ESOL we used 700 graphs as the training data and 100 each for validation and test.



(a) Rank Betti Curve (b) $1/\lambda_2$ (c) Commute Time

Figure 2: Graph properties for Cornell for the different rewiring methods.

	Original	RelWire	SDRF	FOSR	GTR
Zinc	185.092	185.089	185.091	185.091	185.090
Lipo	0.7067	0.6662	0.6325	0.7543	1.1388
BACE	0.5142	0.5556	0.5724	0.6284	0.6174
ESOL	0.8605	0.7631	1.4253	0.6936	0.6012

Table 3: Test mean squared error for the model with the best validation mean squared error over five trials.

Optimization For all methods we used Adam optimizer with the default parameters. We also used cosine annealing as the learning rate decay.

For Lipo, we used a batch size of 40 and trained for 100 epochs. For Zinc we used a batchsize of 80 and also trained for a 100 epochs. For ESOL and BACE, we used a bigger batchsize of 100 but trained for 1000 epochs.

Computing Test Error We trained each model five times. We then picked the trial with the smallest validation accuracy and then reported the corresponding test accuracy.

Computer Resource All datasets were accessed using Pytorch Geometric [16]. The models were all trained on Google Colab using a V100 GPU and pytorch geometric.

For rewiring, for used the official implementations of FOSR and GTR. For SDRF, we used the implementation from the LOG conference tutorial on graph rewiring [2].

C.3. WebKd Experiments

In Figure 2, we show the graphs for Cornell.

C.4. Proof

Lemma 6 *Let E_1, E_2 be edges of a simplicial graph G. If for E_1, E_2 we have that they*
1. are contained in a clique subgraph of G, then E_1, E_2 are independent;
2. are separated by vertex v with $\pi_{E_1}(v)$, $\pi_{E_2}(v)$ both singletons, then E_1, E_2 are transverse.

Proof We start by proving (1). For this let $E_i = (u, v)$ and $E_2 = (a, b)$. Then since this is a clique with all edge weights equal to 1. We have that

$$\pi_{E_1}(a) = \{u, v\} \text{ and } \pi_{E_2}(u) = \{a, b\}.$$

Thus, we have independence.

For (2), we note that since v separates tthe graph into at least two components G_1, G_2 such that $E_1 \in G_1$ and $E_2 \in G_2$. Then we see that for all $x \in G_1$, we have that

$$\pi_{E_2}(x) = \pi_{E_2}(v).$$

Similarly for all $x \in G_2$, we have that

$$\pi_{E_1}(x) = \pi_{E_1}(v).$$

Then since $\pi_{E_1}(v), \pi_{E_2}(v)$ are singletons, we see that we cannot get all four projection pairs. Thus, the edges are transverse. ■

Appendix D. Capturing topological distortion: Distances from persistent homology

In this paper, we consider two notions of distances between graphs using topological information. The first is based on comparing 1-dimensional information, which is already quite powerful in the context of graphs. The second is based on techniques from topological data analysis, which takes into account higher dimensional features of the graphs. This latter machinery is called *persistence homology* (see e.g., [30]), as it attempts to capture "persistent" homological features as one takes larger samples of the space.

Definition 7 (Simplicial Complexes):
A *k-simplex* C is the convex hull of $k+1$ affinely independent vectors. A *simplicial complex* $\mathcal{K}$ is a collection of simplices such that for every $C \in \mathcal{C}$ every face of C is in $\mathcal{K}$ and for every $C_1, C_2 \in \mathcal{K}$, if $C_1 \cap C_2$ is not empty then $C_1 \cap C_2$ is a face of both. The *d-skeleton* of $\mathcal{K}$, denoted $\mathcal{K}^{(d)}$, is the simplicial subcomplex of $\mathcal{K}$ consisting of simplices of dimension at most d.

These topological calculations involve the integral *d^{th}-homology group*. The integral *d^{th}-homology group* of a topological space X, denoted $H_d(X;\mathbb{Z})$, is an abelian group which encodes certain d-dimensional topological features up to a natural topological equivalence. The *rank* of $H_d(X;\mathbb{Z})$—namely the number of its $\mathbb{Z}$-factors—encodes the number of d-dimensional "holes", and is called the d^{th} Betti number β_d. Notably, in dimensions 0 and 1, these numbers have concrete meanings: β_0 encodes the number of connected components of X, and β_1 encodes the number of loops on X (up to homotopy).

In what follows, we will want to consider the homology of simplicial complexes obtained by iteratively adding higher dimensional simplices, with our starting point being a graph. The following example explains how this process can change the homology:

Example 1 (Changing Betti numbers) *Consider a unit 3-cube C as a simplicial complex, and its various skeleta. The 1-skeleton $C^{(1)}$ of C is the graph consisting of the edges of the cube. It is connected, so $\beta_0(C^{(1)}) = 1$, while and $\beta_1(C^{(1)}) = 5$ because the graph can be homotoped to a wedge of five circles. The 2-skeleton $C^{(2)}$ of C is the outside of the cube, which is homotopic to the 2-sphere. Since $C^{(2)}$ has one connected component, we have $\beta_0(C^{(1)}) = 1$. Every loop in contractible in $C^{(2)}$, which means that both*

$\beta_1(C^{(2)}) = 0$ and that $\beta_2(C^{(2)}) = 1$, as the outside of the sphere is not nullhomologous. The 3-skeleton $C^{(3)} = C$ is connected and contractible (it is homotopic to the 3-sphere), so $\beta_0(C) = 1, \beta_1(C) = 0, \beta_2(C) = 0$, and $\beta_3(C) = 1$. Hence adding higher dimensional cells can dramatically change Betti numbers.

Our next goal is to define our two distances. For this, we need a notion of a sequence of simplicial complexes, called a *filtration*, as well as a notion of how topological features can appear and vanish along the filtration, which is called *persistence*; see [4, 21, 22] for more details.

Definition 8 (Filtration):
A filtration of simplicial complexes is a collections of nested simplicial complexes $G_0 \subset G_1 \subset \cdots$. The complex G_k is called the k^{th} level of the filtration.

Definition 9 (Persistence):
Given a filtration $G_0 \subset G_1 \subset \cdots \subset G_k$, we can compute the homology groups for each G_i. Then for any *feature* (homology class), we can compute the first level k at which the feature appears, called the *birth* of the feature and the level at which the feature appears, called the *death*. This collection of birth and death tuples is known as the *persistence diagram*.

In this paper, we will care about two different filtrations where the base complex is a graph. The first is the filtration defined by the subsequent adding of edges by a graph rewiring procedure, in which every level G_k is a graph. The second is a standard filtration known as the Vietoris-Rips filtration. We will use these filtrations to define distances, which the second type of distance being similar to ones used in prior work such as [19, 30].

Definition 10 (Vietoris-Rips filtration):
Let $X = \{x_1, \dots, x_n\}$ be a collection of data points and d a metric on X. Then for any $r \in \mathbb{R}_+$, the *Vietoris-Rips simplicial complex* $VR_r(X)$ is defined by

$$VR_r(X) = \{[x_{i_1}, \dots x_{i_k}] : \forall j, \ell,\ d(x_{i_j}, x_{i_\ell}) \leq r\}.$$

- We call $\{VR_r(X)\}_{r\in\mathbb{R}_+}$ the *Vietoris-Rips Filtration.*

The idea behind the Vietoris-Rips filtration is that it is a way to transform a metric space to a filtration of simplicial complexes, which in the context of a graph involves introducing higher dimensional topological features which are derived from the geometry of the graph. Using a persistence diagram, we can generalize Betti numbers to a more expressive quantity known as the Betti curve. We can use this to define a distance between persistence diagrams.

Definition 11 (Betti Curve):
Let P be a persistence diagram. The *Betti curve* $\beta : \mathbb{R} \to \mathbb{N}$ is a function, where $\beta(r)$ is the number of features that $b \leq r < d$, where b and d are the birth and death.

Definition 12 (Persistence Distance):
The *persistence distance* between two persistence diagrams is the L_2 norm of the difference between their respective Betti curves.

Definition 13 (Betti distance):
Given two graphs G, G', we will call the persistence distance between their respective Vietoris-Rips filtrations $\{VR_r(G)\}_{r\in\mathbb{R}_+}$ and $\{VR_r(G')\}_{r\in\mathbb{R}_+}$ the *Betti distance* between G, G'.

In practice, we will use the Betti distance to compute the higher dimensional "topological distortion" from a base graph G and some other graph G' built from G by adding edges via a rewiring process.

Our second notion of distance measures 1-dimensional topological distortion. A *graph filtration* $G_0 \subset G_1 \subset \cdots$ has G_i a simplicial graph for each i. Graph filtrations naturally arise in the iterative graph rewiring procedures considered in this paper. Since simplicial graphs have no homology beyond dimension 1 and all edges have length 1, the only relevant features of a graph filtration are loops and each birth and death happens at integer time values. Hence their Betti curves are step functions and we obtain:

Lemma 14 (Rank distance) *If $G_0 \subset G_1 \subset \cdots$ and $G'_0 \subset G'_1 \subset \cdots$ are two graph filtrations, then their persistence distance equals*

$$\mathrm{average_i}|H_1(G_i;\mathbb{Z}) - H_1(G'_i;\mathbb{Z})|.$$

- *Hence we call the persistence distance between a pair of graph filtrations the* rank distance*.*

Upshot of topological discussion: In this paper, we use the Betti and rank distances to measure of how much a given rewiring procedure changes the topology of the graph. *For GNNs, the structure of the base graph encodes crucial information. Hence we can use these two distances to measure how much a given graph rewiring procedure preserves the underlying graph structure.*

Appendix E. RelWire: relations on graphs

In this section, we detail our new algorithm, which utilized ideas imported from geometric group theory. We explain the background and motivation first, before describing our algorithm in detail.

E.1. Coarse geometry and relations on graphs

Geometric group theory is interested in the geometry of infinite groups and the metric spaces on which they act. Gromov's work on hyperbolic and CAT(0) spaces [18] introduced various coarse notions of curvature to the area, which had transformative implications in the study of low-dimensional hyperbolic manifolds via their fundamental groups.

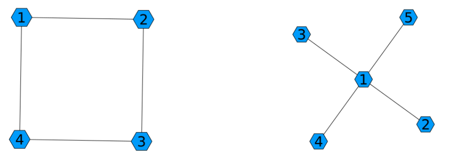
Hierarchical hyperbolicity [6] is an axiomatic framework for studying hybrid spaces which exhibit aspects of coarse negative, flat, and positive curvature. This hierarchical approach builds on work in several areas of low dimensional topology, including mapping class groups ([23], Teichmüller spaces ([10, 15, 29]), and hyperbolic 3-manifolds ([11, 24]). These *hierarchically hyperbolic spaces* (HHSes) are coarsely built out of hyperbolic spaces, which are combined in both negative and flat curvature ways based on various *relations* between the hyperbolic spaces. In this paper, we will apply a simplified version of this hierarchical framework to study curvature properties of graphs. In particular, we will use the geometry of a fixed graph to induce two (mutually exclusive) types of relations among its edges. In the general setting, one starts with an ambient geodesic metric space X with a finite collection of Gromov hyperbolic geodesic metric spaces $\mathcal{V}$. We note that HHSes were developed to study infinite groups where $\mathcal{V}$ is infinite, but we can and will assume $\mathcal{V}$ is finite for simplicity of this discussion. To each space $V \in \mathcal{V}$, there is an associated projection map $\pi_V : X \to V$. The guiding philosophy is that these projections behave like closest-point projections to convex subspaces, and that they collectively coarsely encode most of the geometry of X.

In our setting, the ambient space $X = G$ is a simplicial graph and this philosophy becomes quite simple: the spaces in the hierarchy are the edges, and a projection of a vertex of G to an edge E is a some collection of its endpoints. Specifically, given a vertex $v \in G^{(0)}$ and an edge E of G, the *projection* $\pi_E(v) \subset E^{(0)}$ of v to E is the endpoint of E which is closest in G to v. When both endpoints are equidistant to v, then we set $\pi_E(v) = E^{(0)}$ to be both endpoints. With these projections defined, we define our (simplified) relations.

The first relation, called *orthogonal*, encodes flat curvature. In the setting of an HHS X, when two hyperbolic spaces U, V in the hierarchy are orthogonal, the product map $\pi_U \times \pi_V : X \to U \times V$ is surjective, and there is a coarsely isometrically embedded flat subspace of X (see e.g. Subsection 5B of [5]). For instance, $\mathbb{R}^2$ is an HHS where the hyperbolic spaces are the coordinate axes (i.e., copies of $\mathbb{R}$), and the flat subspace corresponding to their product is the whole ambient space $\mathbb{R} \times \mathbb{R} = \mathbb{R}^2$.

In our graphical setting, we will say two edges E_1, E_2 are *independent* when $\pi_{E_1} \times \pi_{E_2} : G \to E_1^{(0)} \times E_2^{(0)}$ is surjective. Otherwise, we will say that E_1, E_2 are *transverse*. Roughly, this notion of transversality encodes negative curvature (see e.g. [7]).

While the connection between independence/transversality and flat/negative curvature is not exact, the connection is more than moral:



(a) 4 cycle (b) Cross

Figure 3: Here are two simple graphs in which: (a) all adjacent edges are independent and (b) all edges are transverse.

Lemma 15 *Let E_1, E_2 be edges of a simplicial graph G. If for E_1, E_2 we have that they*

1. *are contained in a clique subgraph of G, then E_1, E_2 are independent;*
2. *are separated by vertex v with $\pi_{E_1}(v)$ and $\pi_{E_2}(v)$ both singletons, then E_1, E_2 are transverse.*

Much more is true in practice. For instance, most edges in a given loop will be pairwise independent, while the condition in item (2) of Lemma 15 is fairly generic. More refined relations are capable of exactly encoding, e.g. that independence and being in a loop are equivalent, but these conditions are difficult to state and even slower to implement algorithmically.

Remark 16 (Global vs. local curvature):
As every pair of edges in a graph satisfies one of our two relations, they are both capable of capturing local and global properties of the graph. While item (2) of Lemma 15 says that independence is frequently more locally focused, transversality captures negative curvature in a fundamentally different way than existing notions of graph curvature.

E.2. RelWire

We present our new rewiring technique RelWire. The pseudocode for the method can be seen in Algorithm 2. The basic idea is to eliminate negative curvature by adding edges, so the main task is to identify pairs of vertices which belong to the most transverse edge pairs, weighted by their distance in the graph.

Algorithm 2 RelWire

```
1: function RelWire($G$ - Graph, $k$ - number of edges added)
2:     Compute shortest distance $d(u, v)$ between all pairs of nodes $u, v$.
3:     Compute $T : E \times E \to \{0, 1\}$ such that $T(e_1, e_2) = 1$ if and only if $e_1 \not\perp e_2$.
4:     Compute $r(u, a) = d(u, a) \sum_{v \in \mathcal{N}(u)} \sum_{b \in \mathcal{N}(a)} T\big((u, v), (a, b)\big)$.
5:     Connect the $k$ non-adjacent node pairs with largest $r$ value.
6:     return Rewired Graph
7: end function
```

We begin by determining for each pair of edges if they are independent or transverse. Then for each pair of nodes u, a, we consider all neighboring edges of the form (u, v) and (a, b) for all $v \in \mathcal{N}(u)$ and $b \in \mathcal{N}(a)$. Then we define

$$r(u, a) = d(u, a) \sum_{\substack{v \in \mathcal{N}(u) \\ b \in \mathcal{N}(a)}} \mathbb{1}\{(u, v) \not\perp (a, b)\}.$$

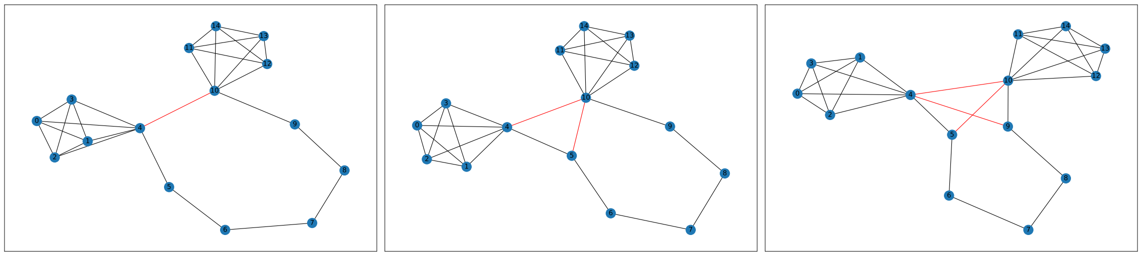
We then connect the k node pairs that are not adjacent in the graph that have the highest r values, where k is our rewiring parameter. That is, for a pair of nodes, we count the number of transverse edge pairs that the two nodes are in. We then weight this count by the distance between the two nodes, and connect the pairs of nodes with the highest weighted r value.

As discussed above, tranversality captures some notion of negative curvature, at both the local and global scale of the graph. Hence connecting a highly transverse distant pair morally helps remove negative curvature at a global level.

Appendix F. Preserving the topology: barbell example

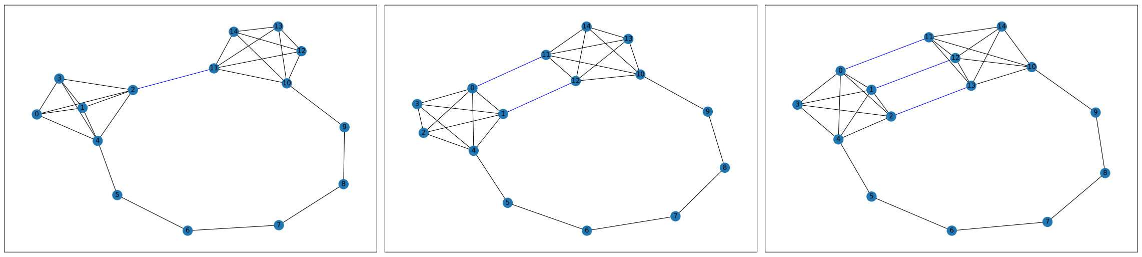
In this section we explore the topological differences between RelWire, FOSR, GTR, and SDRF. We will do this using the standard example of a barbell graph G (Figure 5a) that has been used in prior work such as [13, 14, 20, 34]. We shall use all four methods to add between one and nine edges, and we shall see that the results are quite different. To understand the topological distortion caused by rewiring, we compute both the rank and Betti distances relative to the underlying graph.

Figure 4 shows the rewired graphs for $k = 1, 2, 3$ and Figure 5b shows the ranks for adding up to 9 edges. Here we see that RelWire initially introduces a loop and the rank increases to 1, while successive edges fill in that loop. On the other hand, both FOSR and



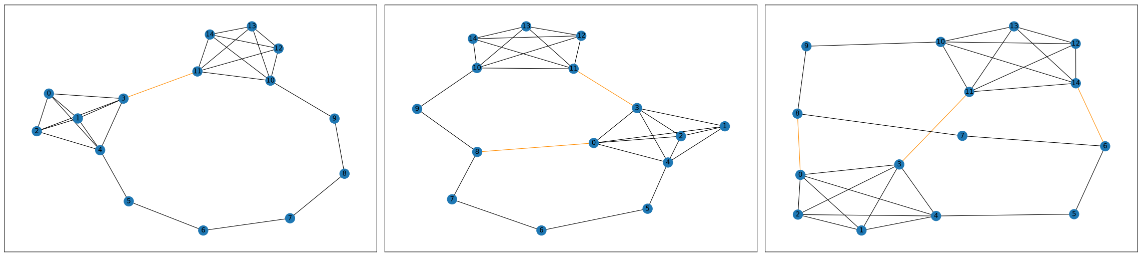
(a) $k = 1$ (b) $k = 2$ (c) $k = 3$

(d) RelWire rewiring of the barbell graph



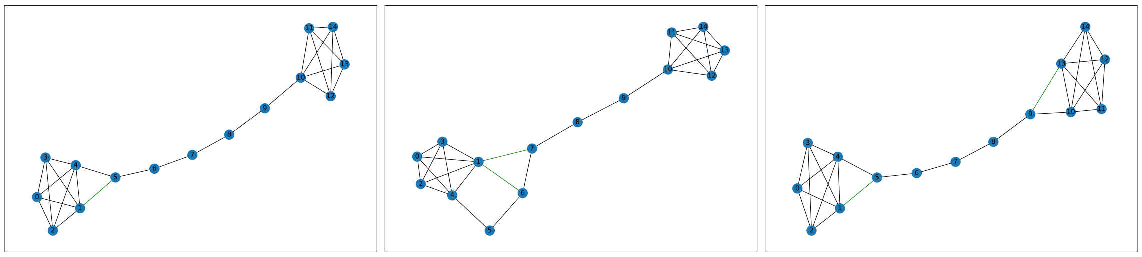
(e) $k = 1$ (f) $k = 2$ (g) $k = 3$

(h) FOSR rewiring of the barbell graph



(i) $k = 1$ (j) $k = 2$ (k) $k = 3$

(l) GTR rewiring of the barbell graph

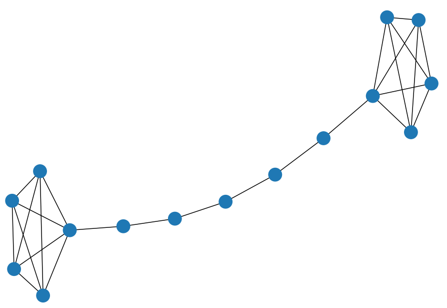


(m) $k = 1$ (n) $k = 2$ (o) $k = 3$

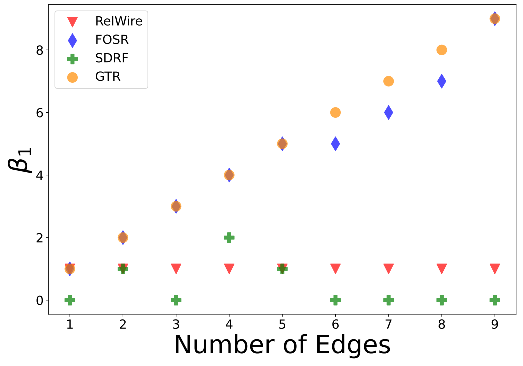
(p) SDRF rewiring of the barbell graph

Figure 4: Rewired barbell graph.

GTR create a loop with each successive edge addition. Finally, it is not clear what SDRF does to the topology and the topological rank is not very stable.

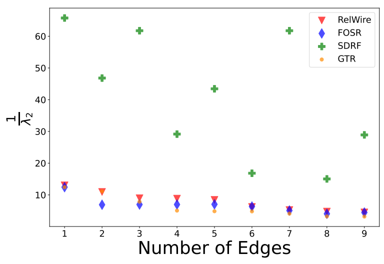


(a) Barbell Graph

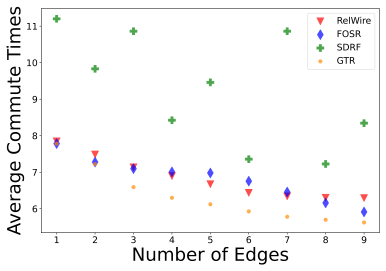


(b) Ranks

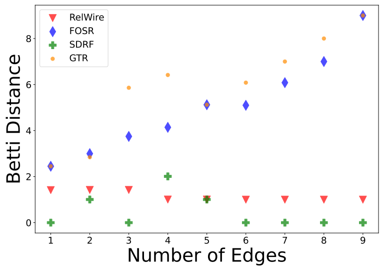
Figure 5: (a) Barbell Graph. (b) The ranks of the first homology group of the rewired graphs.



(a) $\frac{1}{\lambda_2}$



(b) Average Commute Times



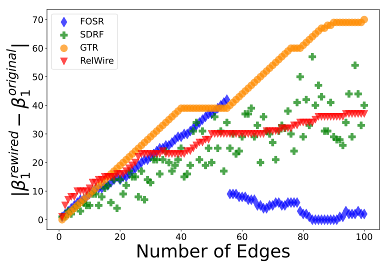
(c) Betti Distance

Figure 6: Figure showing $\frac{1}{\lambda_2}$, the average commute times, as well as the Betti distance between the rewired graph and the original graph for RelWire, SDRF, GTR, and FOSR.

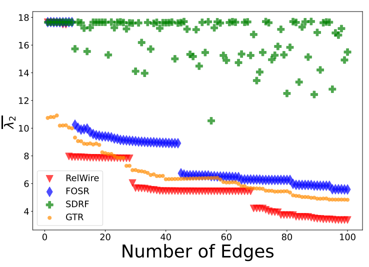
We also analyze how rewiring affects the statistics related to oversquashing, i.e. average commute times and eigengaps of the graph. These quantities, along with Betti distance to the original graph, are plotted in Figure 6. As we can see, RelWire, GTR, and FOSR have the best average commute times and eigengaps, whereas SDRF is ineffective. On the other hand, we see that FOSR and GTR result in large topological changes, whereas RelWire has relatively little effect on the topology.

Appendix G. Betti Curve for Rank Persistence

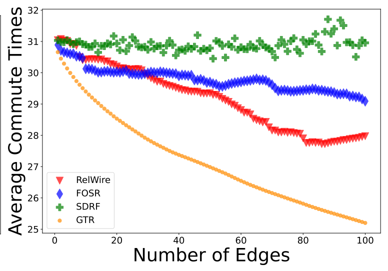
In the previous experiments we only added a fixed number of edges to the graphs. As with the barbell graph example, it is interesting to see how the statistics change as we vary the number of edges added. Hence, we took Texas and Cornell from the WebKb dataset [28] and added up to 100 edges. Figure 7 shows the results for Texas. The one for Cornell can be seen in the Appendix. Here we see that RelWire has the best eigengap, FOSR has the best rank distance, and GTR has the best commute times. There are many other interesting aspects to the curves. The first is the jump discontinuity in the rank of the first homology group from FOSR. This implies that FOSR reaches a critical number of edges after which adding loops is no longer beneficial and starts eliminating loops. On the other, GTR and SDRF seem to always add loops. RelWire on the other hand, seems to always want to eliminate topological loops. The jump discontinuity in the rank of the first homology group for FOSR seems to correlate with the jump discontinuity in the eigengap curve. However, interestingly we see no discontinuity in the commute times curve.



(a) Rank distance



(b) $1/\lambda_2$



(c) Commute Time

Figure 7: Comparing the four rewiring methods after many edge additions: in (a), rank distance; (b) eigengap (λ_2^{-1}), and (c) the average commute times, each as a function of the number of edges added.

Appendix H. Conclusion

In conclusion, we use ideas from geometric group theory to develop a new rewiring technique known as RelWireusing a new curvature-like relation on edges. We also introduce a new topological distance which measures how rewiring changes the structure of a graph. We show that compared to other method RelWireis Pareto optimal in that it makes small topological changes to the graph and makes big changes to statistics connected to oversquashing such as eigengap and commute times. We test RelWireon the downstream task of graph regression and report positive results.

Dataset	RelWire	FOSR	GTR	SDRF
Zinc	9.96	10.9	9.22	12.7
ESOL	6.1	6.3	5.7	7.0
BACE	13.9	15.4	12.7	17.7
Mutag	7.7	8	7.2	8.8
AIDS700nef	4.5	4.4	4.1	4.8
Alkane	4.8	4.6	4.2	5.2
Linux	3.9	3.8	3.6	4.3

Table 4: Average commute times after rewiring.