More Expressive GNNs: Mitigating Oversquashing

S Deepak Narayanan, 22nd March 2022

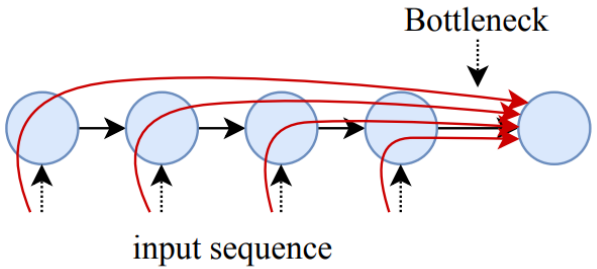
Mentor: Kenza Amara

Overview

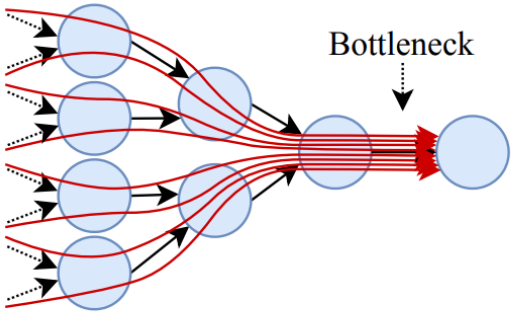
- Introduction to Oversquashing
  - Oversquashing v/s Oversmoothing
  - An example problem and a simple rewiring solution
- More Solutions to Alleviate Oversquashing
  - Geometric GCN
  - Rewiring with Positional Encodings
- A curvature perspective on Oversquashing

What is Oversquashing?

- Aggregation in multi-hop GNNs involve large neighborhoods
- GNNs “compress” this information into a fixed-length vector
- Bottleneck → Loss of Information
- Consequence: Difficult to learn long range information



(a) The bottleneck of RNN seq2seq models



(b) The bottleneck of graph neural networks

[1] Alon, Uri, and Eran Yahav. "On the bottleneck of graph neural networks and its practical implications." arXiv preprint arXiv:2006.05205 (2020).

Picture taken from [1]

Oversmoothing v/s Oversquashing

Oversmoothing

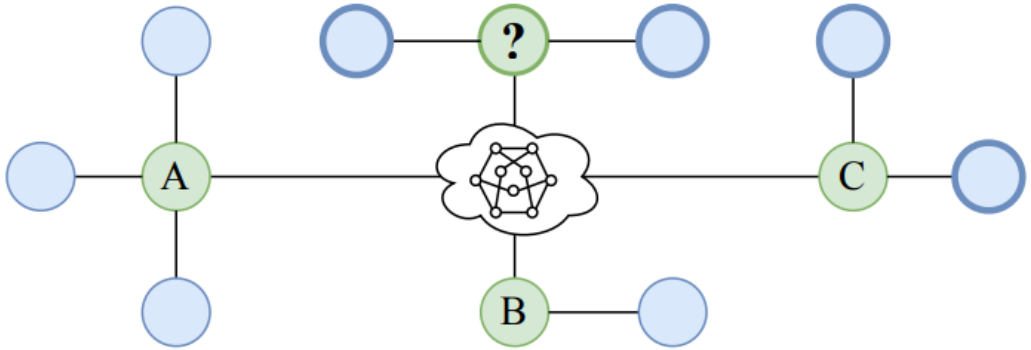
- Refers to all node embeddings converging to similar vectors
- Found to occur with increasing number of layers
- Common in short-range tasks

Oversquashing

- Bottleneck caused due to information compression
- Issue more related to graph topology than # of GNN layers
- More relevant to long-range tasks

Demonstrating Oversquashing

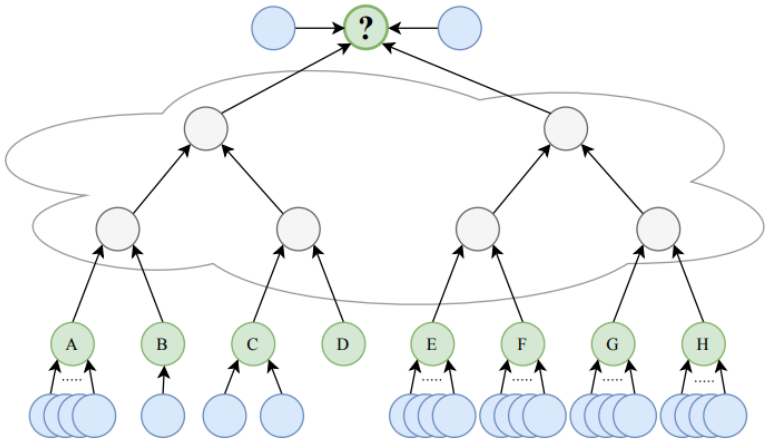
- The Neighbors Match Problem
- Multiple Graphs; Labels are a function of the number of immediate blue neighbors
- A single layer GNN can count, but cannot infer the label!



Picture taken from [1]

Tree Neighbors Match

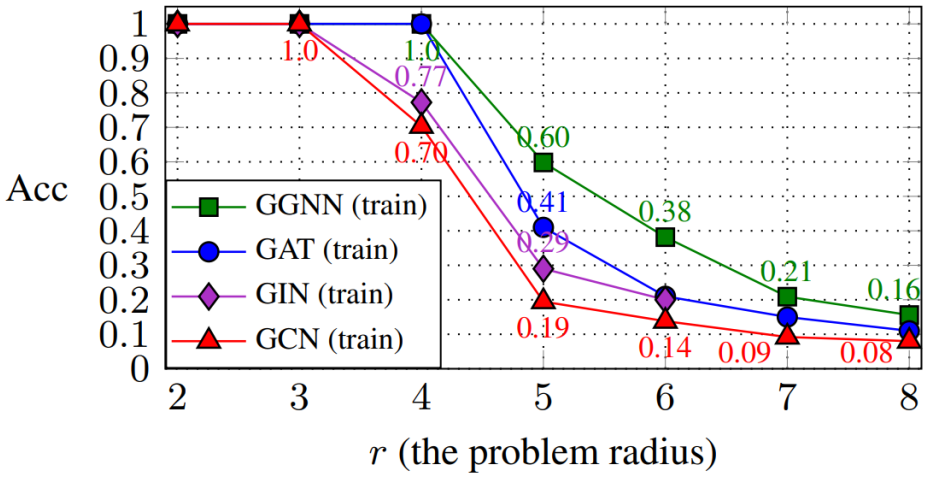
- Suppose the cloud is a binary tree
- We can control the "problem radius" (r) – minimum number of GNN layers needed to propagate sufficient information
- This example has Problem Radius 3



Picture taken from [1]

Tree Neighbors Match - Results

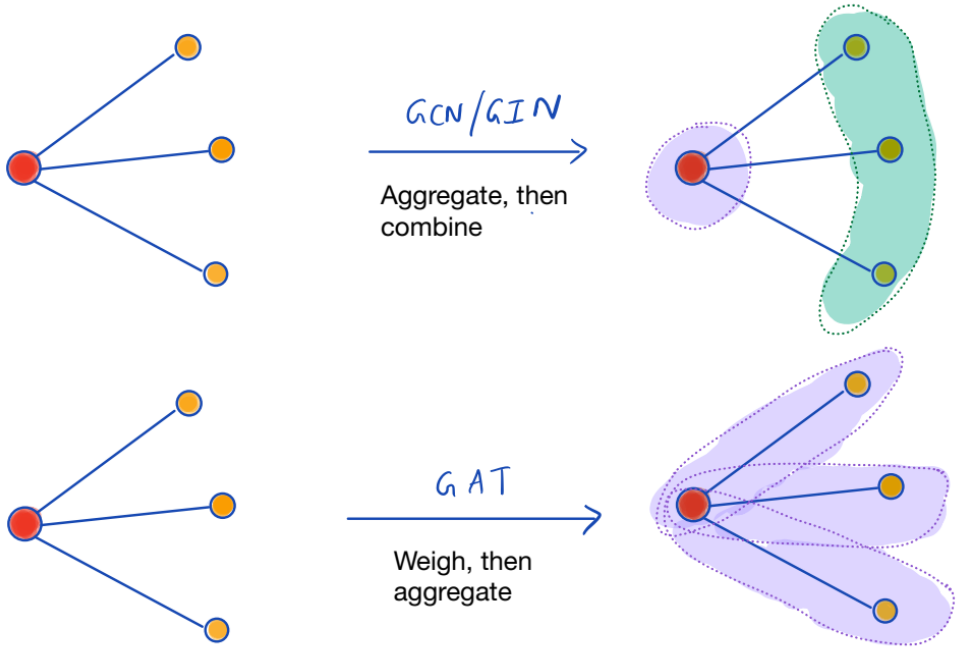
- Take GNNs of $k = r + 1$ layers
- Even $r = 4$ causes over-squashing
- Extent of Oversquashing depends on the aggregator



Picture taken from [1]

Tree Neighbors Match - Analysis

GIN and GCN suffer from oversquashing “before” GAT!



GAT can potentially *ignore* half the information

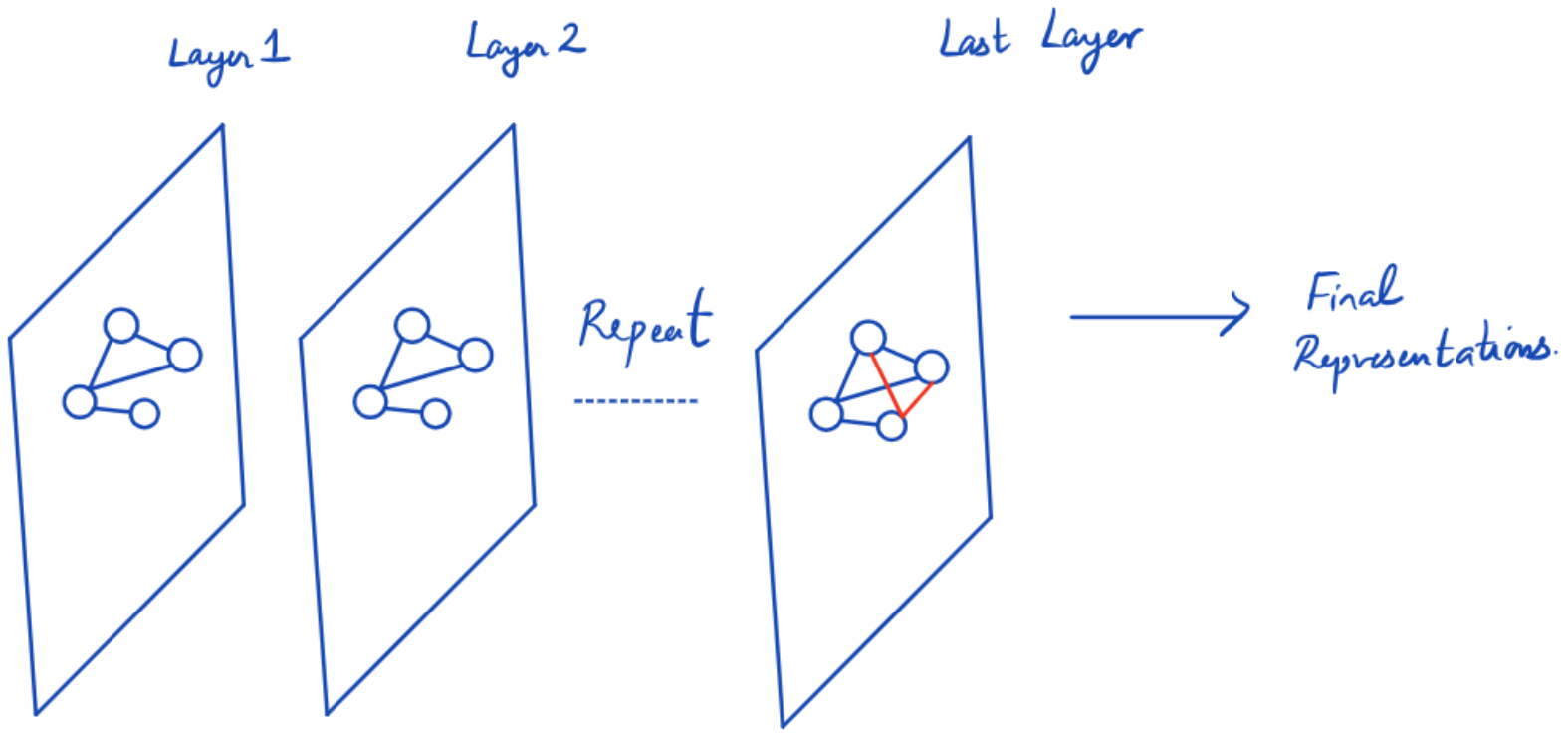
Rewiring: Solutions Outlook

Oversquashing Consequences

→ Capturing long-range dependencies are hard

- Rewiring involves modifying the underlying graph structure
- Modification: Changes the graph connectivity by addition or removal of edges to ease information flow
- Solution 1: A Fully Adjacent Layer

Solution 1: A Fully Adjacent (Last) Layer



Fully Adjacent (FA) Layer – Results

Datasets: Quantum Chemistry, Biological, Computer Programs

All of these datasets contain long-range problems

	R-GIN	
Property	base†	+FA
mu	2.64±0.11	**2.54**±0.09
alpha	4.67±0.52	**2.28**±0.04
HOMO	1.42±0.01	**1.26**±0.02
LUMO	1.50±0.09	**1.34**±0.04
gap	2.27±0.09	**1.96**±0.04
R2	15.63±1.40	**12.61**±0.37
ZPVE	12.93±1.81	**5.03**±0.36
U0	5.88±1.01	**2.21**±0.12
U	18.71±23.36	**2.32**±0.18
H	5.62±0.81	**2.26**±0.19
G	5.38±0.75	**2.04**±0.24
Cv	3.53±0.37	**1.86**±0.03
Omega	1.05±0.11	**0.80**±0.04
Relative:		-39.54%

		NCI1	ENZYMES
No Struct†		69.8±2.2	65.2±6.4
DiffPool	base†	76.9±1.9	59.5±5.6
	+FA	**77.6**±1.3	**65.7**±4.8
GraphSAGE	base†	76.0±1.8	58.2±6.0
	+FA	**77.7**±1.8	**60.8**±4.5
DGCNN	base†	76.4±1.7	38.9±5.7
	+FA	**76.8**±1.5	**42.8**±5.3
GIN	base†	80.0±1.4	59.6±4.5
	+FA	**81.5**±1.2	**67.7**±5.3

Table 2: Average accuracy (30 runs±stdev) on the biological datasets. † – previously reported by Errica et al. (2020).

		SeenProj	UnseenProj
GGNN†	base†	85.7±0.5	**79.3**±1.2
	+FA	**86.3**±0.7	79.1±1.1
R-GCN	base†	88.3±0.4	82.9±0.8
	+FA	**88.4**±0.7	**83.8**±1.0
R-GIN	base†	87.1±0.1	81.1±0.9
	+FA	**87.5**±0.7	**81.7**±1.2
GNN-MLP	base†	86.9±0.3	**81.4**±0.7
	+FA	**87.3**±0.2	81.2±0.5
R-GAT	base†	86.9±0.7	81.2±0.9
	+FA	**87.9**±1.0	**82.0**±1.9

Table 3: Average accuracy (5 runs±stdev) on VARMISUSE. † – previously reported by Brockschmidt (2020).

Results taken from [1]

Fully Adjacent Layer - Analysis

- Impressive performance gains over SOTA
- Why not make all layers FA?
- Graph topology
  - Provides Relevant Inductive Bias → Regularization Effect
  - Empirically 1500% higher error with all layers FA!
- Pros: Simple, Easy to Implement ☺
- Cons: Computationally expensive for large graphs ☹

FA layer eases information flow and relieves bottleneck while retaining graph topology from previous layers

Results taken from [1]

Rewiring: Solutions Outlook

Oversquashing Consequences

→ Capturing long-range dependencies are hard

- Rewiring involves modifying the underlying graph structure
- Modification: Changes the graph connectivity by addition or removal of edges to ease information flow
- Solution 1: A Fully Adjacent Layer
- Solution 2: Geometric GCN

Geometric GCN

- Neighborhoods are defined by the underlying graph
- Q. Can we bring together nodes that are far apart but structurally similar, and involve them in aggregation?
- Such nodes are especially important in disassortative graphs (biological networks)
- More concretely, can we design modified neighborhoods over which aggregation can capture long-range dependencies?

[2] Pei H, Wei B, Chang KC, Lei Y, Yang B. Geom-gcn: Geometric graph convolutional networks. arXiv preprint arXiv:2002.05287. 2020 Feb 13.

Geometric GCN – Key Ideas

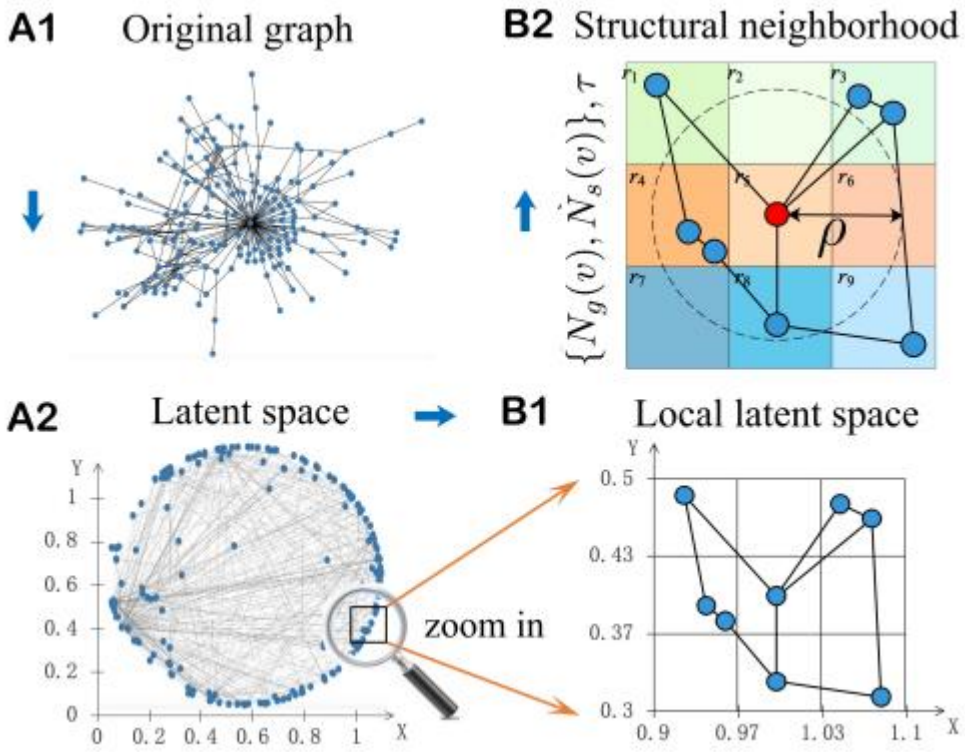
- Construct a latent space where structurally similar nodes appear together
- Exploit the underlying geometry of the latent space to define new neighborhoods for aggregation

Geometric GCN – Modules and Components

- Node Embedding Module
  - Maps the nodes to a latent continuous space
- Structural Neighborhood:
  - Graph defined neighborhood and Latent space neighborhood
- Bi-Level Aggregation: This module has two levels of aggregation
  - Low-Level Aggregation: Nodes from the same neighborhood and same geometric relationship are aggregated into a virtual node
  - High-Level Aggregation: Features are aggregated from virtual nodes to generate final representations

Geometric GCN – Modules

- Node Embedding Module f: For Graph $G = (V, E)$, define $f: v \rightarrow z_v,\ z_v \in \mathbb{R}^d,\ v \in V$
- Neighborhood of a node : $N(v) = (\{N_g(v), N_s(v)\}, \tau)$.Here, $N_g(v), N_s(v)$ are respectively the graph neighborhood and the latent space neighborhood
- $N_s(v) = \{u \,|u \in V, d(z_u, z_v) < \rho\}$, where $d(z_u, z_v)$ is the distance metric in the latent space
- τ is a relational operator; τ: $(z_u, z_v) \rightarrow r \in R$, where R is the set of geometric relations; E.g.: Direction w.r.t target node



Picture Taken from [2]

Geometric GCN – Aggregation

- Low Level Aggregation:

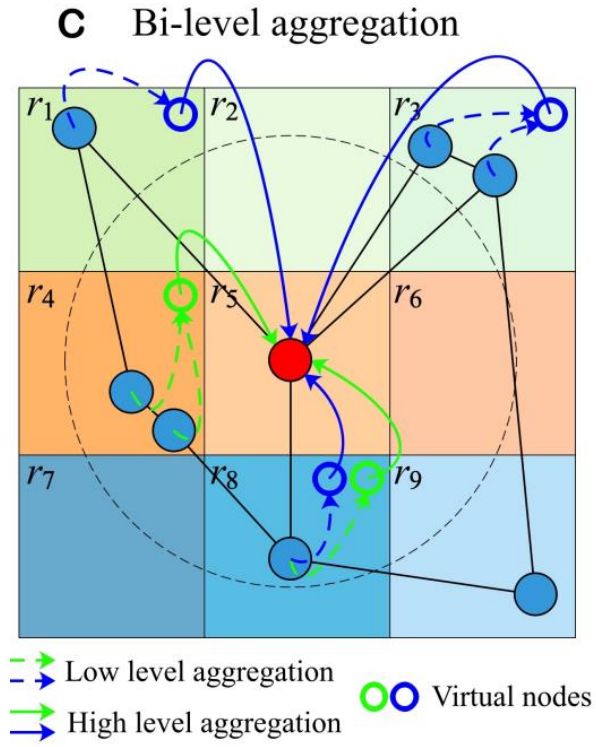
$$e_{i,r}^{v} = p(\{h_u | u \in N_i(v), \tau(z_v, z_u) = r\})$$
$$\forall i \in \{g, s\}, \forall\, r \in R$$

- High Level Aggregation:

$$m_v = q(e_{i,r}^{v}, (i, r)) \forall i \in \{g, s\}\, \forall\, r \in R$$

- Representation for layer L:

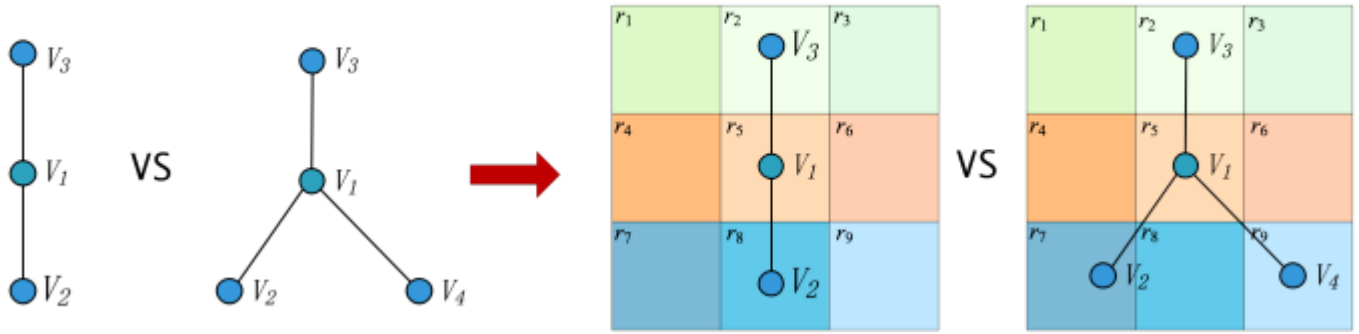
$$h_v^L = \sigma(m_v^L; W^L)$$



Picture Taken from [2]

Advantages of a geometric neighborhood

- Geometric neighborhood informs every edge differently
- Can distinguish different graphs, even with same aggregator!



Picture Taken from [2]

Geometric GCN – Empirical Results

- GeomGCN with GCN aggregator outperforms GAT and GCN significantly; Huge performance gains in disassortative graphs
- Design of Latent Space method is critical for performance

Table 3: Mean Classification Accuracy (Percent)

Dataset	Cora	Cite.	Pubm.	Cham.	Squi.	Actor	Corn.	Texa.	Wisc.
GCN	85.77	73.68	88.13	28.18	23.96	26.86	52.70	52.16	45.88
GAT	**86.37**	74.32	87.62	42.93	30.03	28.45	54.32	58.38	49.41
Geom-GCN-I	85.19	**77.99**	**90.05**	60.31	33.32	29.09	56.76	57.58	58.24
Geom-GCN-P	84.93	75.14	88.09	**60.90**	**38.14**	**31.63**	**60.81**	**67.57**	**64.12**
Geom-GCN-S	85.27	74.71	84.75	59.96	36.24	30.30	55.68	59.73	56.67

Results taken from [2]

Geometric GCN – Summary and Analysis

- Alleviates Oversquashing by rewiring the graph based on a structural neighborhood ☺
- Requires manual design of geometric relations ☹
  - E.g. Direction from Target Node
- Performance heavily depends on the embedding module
  - Current work uses methods such as Struct2Vec, Poincare, and Iso-map, all of which have strong inductive biases

Rewiring: Solutions Outlook

Oversquashing Consequences

→ Capturing long-range dependencies are hard

- Rewiring involves modifying the underlying graph structure
- Modification: Changes the graph connectivity by addition or removal of edges to ease information flow
- Solution 1: A Fully Adjacent Layer
- Solution 2: Geometric GCN
- Solution 3: Rewiring with Positional Encodings

Rewiring with Positional Encodings - Preliminaries

- Receptive field of a node is it's immediate neighbors from which it aggregates information
- To reduce oversquashing, increase receptive field to the entire graph
  - Issues: Poor performance and introduces computational load
- Trade some compute for increasing receptive fields, while reducing oversquashing

[3] Brüel-Gabrielsson, Rickard, Mikhail Yurochkin, and Justin Solomon. "Rewiring with Positional Encodings for Graph Neural Networks." arXiv preprint arXiv:2201.12674 (2022).

Positional Encodings in GNNs

- Additional information about graph topology provided either as node/edge attribute for GNNs
- Examples: Laplacian Spectra, Node Degrees, Shortest Path Lengths
- Increase GNN expressivity

Rewiring with Positional Encodings: Key Ideas

- Increase receptive field to a k-hop neighborhood where $k \ll D$ and D is the diameter of the graph → Ease information flow
- Introduce positional encodings → More expressivity and preserve graph topology
- Introduce Virtual Nodes → Ease information flow

Increasing Receptive Fields: Idea I

Expanded Receptive Field

- Given a graph $G = (V, E, f_v, f_e)$, with node attributes f_v and edge attributes f_e, add edges between all nodes within k-hops of each other to create $G' = \ (V, E', f_v', f_e')$
- Set constant feature $C_e \forall\ e \in E' \backslash \mathrm{E}$
- Virtual Node: Add a virtual node v_{CLS} to V, and add an edge between v_{CLS} and every other node
- Set $f_v'(v_{CLS}) = C_v$ for some constant C_v

Introducing Positional Encoding: Aims

- Lossless encodings – Be able to recover the original graph
- Discriminative Power: Encodings should improve the discriminative power measured in terms of the 1-WL Test
- Context Range: Global or Local Information

Positional Encoding: Idea II - Options

- Shortest Path Encodings (SPE):
  - Edge positional encoding denoting the shortest path distance between two nodes in the graph
  - Lossless encoding as we can recover G from G' when this encoding is 1
  - Expanded receptive fields + SPE is more powerful than the 1-WL Test
- Spectral Embeddings:
  - Node positional encoding consisting eigenvectors of the Laplacian
  - Not necessarily lossless, but works well in practice
  - Contains Global Information about the entire graph

Positional Encoding: Idea II - Options

- Powers of the Adjacency Matrix:
  - Edge positional encoding generalizing SPE
  - Captures number of paths between a pair of nodes
  - Lossless encoding, since we can recover G from G' using the first power of A

Positional Encoding: Options - Summary

- Shortest Path Encodings – Edge-based, Lossless, Expressive, Local Context
- Spectral Embeddings – Node-based, Lossless*, Global Context
- Powers of the Adjacency Matrix – Edge-based, Lossless, Generalizes SPE, Expressive, Controllable Context

Empirical Results – Benchmark Datasets

- Strong empirical performance with fewer parameters
- Empirically required receptive field has size $k \ll D$
- Encoding primarily depends on the dataset

Table 2. Benchmarking. Higher is better for all but for ZINC where lower is better. All results can be found in (Dwivedi et al., 2020; Corso et al., 2020; Bouritsas et al., 2020) and the leaderboard at this link. The benchmarks and corresponding leaderboard have 100K and 500K parameter entries, with many models only appearing with a subset of datasets or number of parameters; dashes indicate that the result for corresponding model, number of parameters, and dataset was not found in their paper or on the leaderboard.

Datasets:	PATTERN	CLUSTER	MNIST	CIFAR10	ZINC
task:	node class.	node class.	graph class.	graph class.	graph reg.
# graphs:	14000	12000	70000	60000	12000
Avg # nodes:	117.47	117.20	70.57	117.63	23.16
Avg # edges:	4749.15	4301.72	564.53	941.07	49.83
MoNet(100K)	85.482±0.037	58.064±0.131	90.805±0.032	54.655±0.518	0.397±0.010
GAT(100K)	75.824±1.823	57.732±0.323	95.535±0.205	64.223±0.455	0.475±0.007
PNA(100K)	-	-	97.940±0.120	70.350±0.630	0.188±0.004
PNA(500K)	-	-	-	-	**0.142±0.010**
DGN(100K)	-	-	-	72.838±0.417	-
GSN(100K)	-	-	-	-	**0.140±0.006**
GSN(500K)	-	-	-	-	**0.101±0.010**
GatedGCN(100K)	84.480±0.122	60.404±0.419	97.340±0.143	67.312±0.311	0.328±0.003
GatedGCN-PE/E(500K)	86.363±0.127	74.088±0.344	-	-	0.214±0.006
Ours (100K)	**86.757±0.031**	**77.575±0.149**	**98.743±0.062**	**73.808±0.193**	**0.143±0.006**

Results taken from [3]

Empirical Results – Tree Neighbors Match

- In the figure, r is the receptive field, and r_p is the problem radius
- With $r = 1$, and v_{CLS}, similar performances of $r = 2$, and $r = 3$

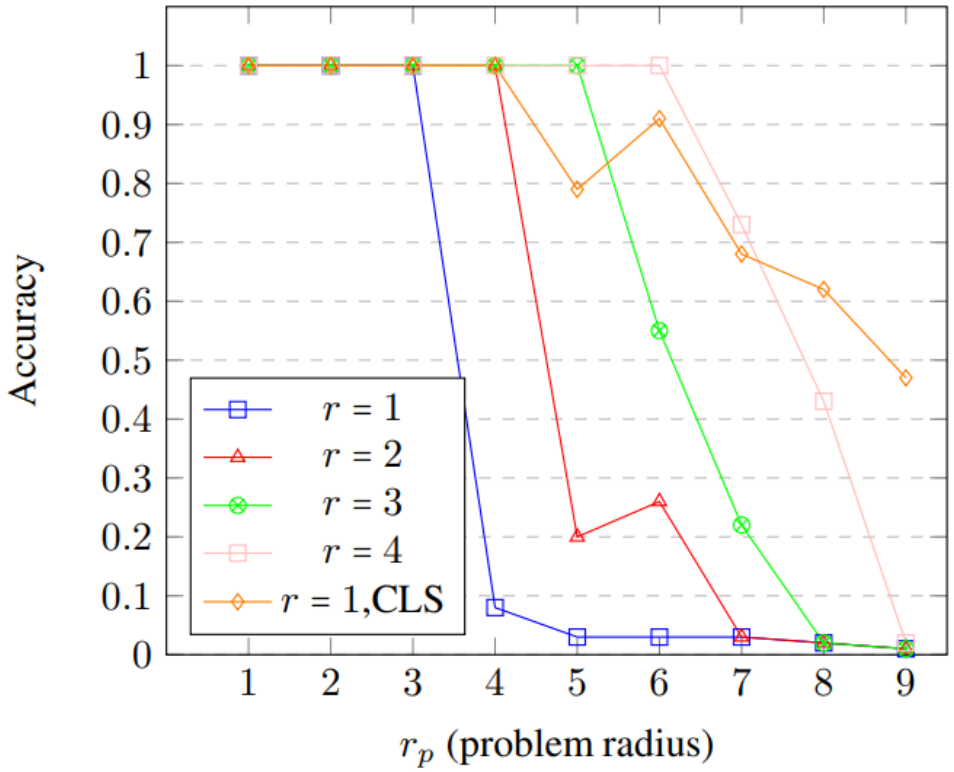


Figure 1. NeighborsMatch (Alon and Yahav, 2021). Benchmarking the extent of over-squashing via the problem radius r_p.

Results taken from [3]

Rewiring with Positional Encodings - Summary

- Reduces Oversquashing by increasing receptive fields ☺
- Provides neat incorporation of positional encodings that can theoretically make the GNN more expressive ☺
- Need to manually tune over receptive fields sizes ☹
- Positional Encodings are application specific in practice ☹

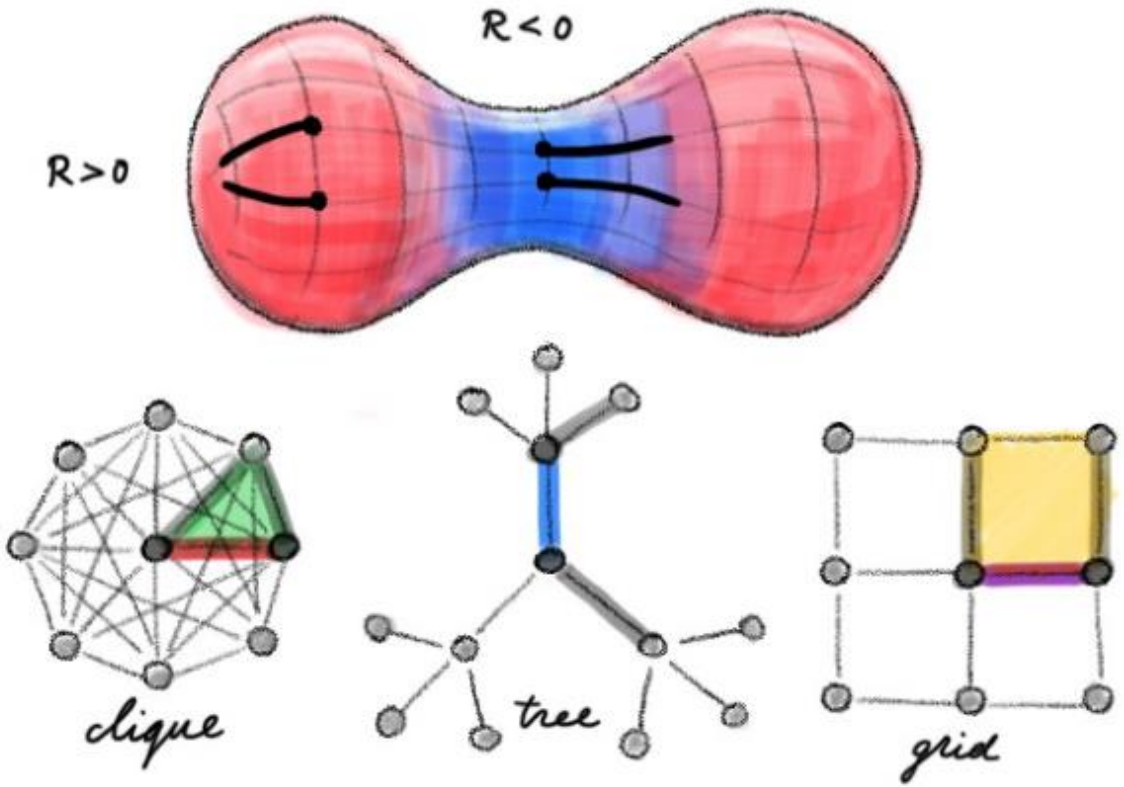
Rewiring: Solutions Outlook

Oversquashing Consequences

→ Capturing long-range dependencies are hard

- Rewiring involves modifying the underlying graph structure
- Modification: Changes the graph *connectivity* by addition or removal of edges to *ease* information flow
- Solution 1: A Fully Adjacent Layer
- Solution 2: Geometric GCN
- Solution 3: Rewiring with Positional Encodings
- A Curvature Perspective of Oversquashing

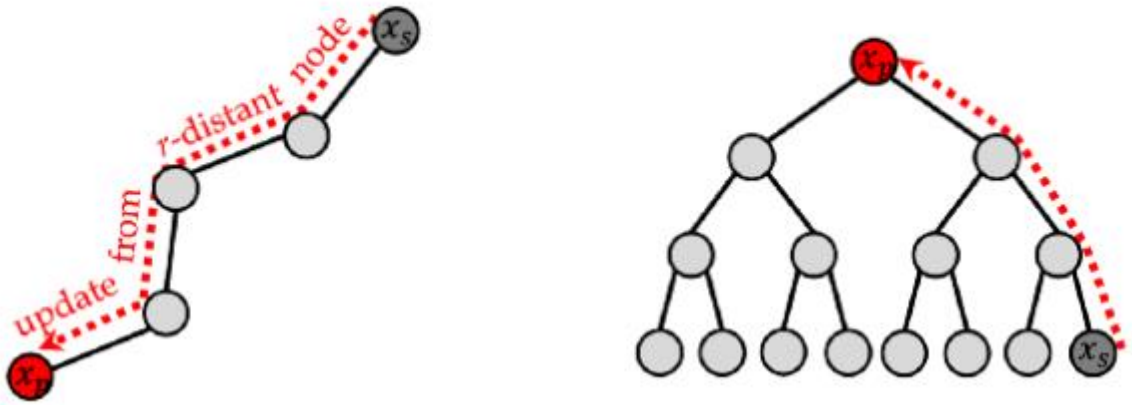
A curvature perspective of Oversquashing



Picture Taken from [5]

[4] Topping, Jake, et al. "Understanding over-squashing and bottlenecks on graphs via curvature." arXiv preprint arXiv:2111.14522 (2021).
[5] Bronstein, M. (2021, November 30). *Over-squashing, bottlenecks, and graph Ricci curvature*. Medium. Retrieved March 12, 2022, from https://towardsdatascience.com/over-squashing-bottlenecks-and-graph-ricci-curvature-c238b7169e16

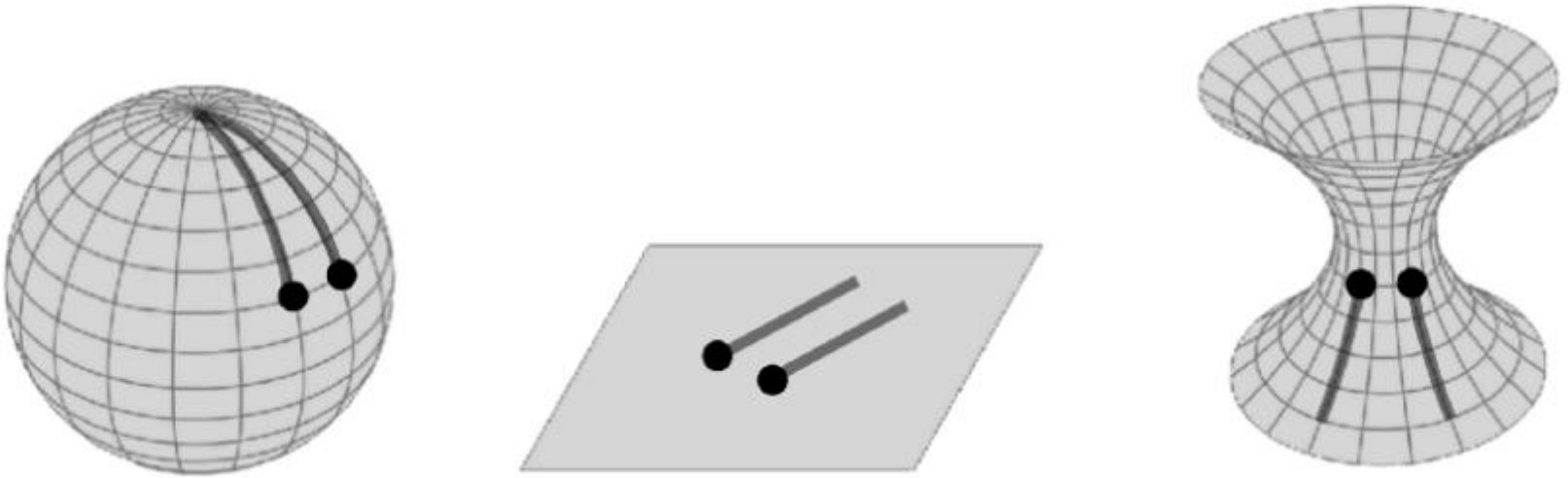
Curvature – Sensitivity



- Sensitivity $\left|\frac{\partial h_u}{\partial x_s}\right|$ can quantify how much a node representation is affected by other node representations
- Sensitivity is bounded by powers of adjacency matrix → Graph Topology is responsible!
- Smaller value potentially indicates oversquashing

Picture Taken from [5]

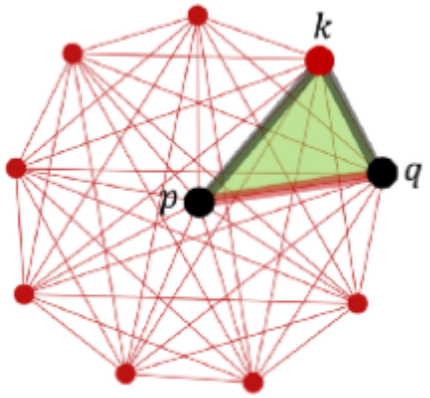
Ricci Curvature of a Manifold



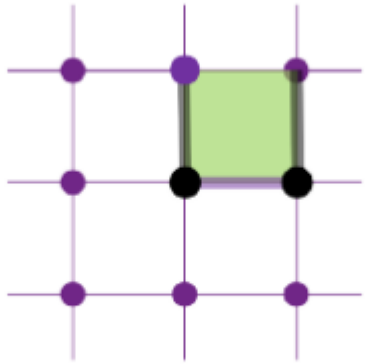
Curvature is characterized by “geodesic dispersion”

Picture Taken from [5]

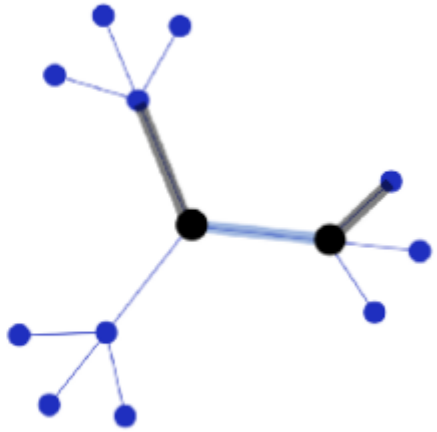
Geodesic Dispersion on Graphs



Clique (>0)



Grid (=0)



Tree (<0)

Picture Taken from [5]

Curvature for Graphs and Take-Away

- Define a version of curvature for edges based on geometric cues
  - For Spherical Geometry, Count Triangles
  - For Euclidean Geometry, Count 4-cycles
  - For Hyperbolic Geometry, Count # of outgoing edges
- Sensitivity is related to graph specific curvature
- Conclusion: Negatively curved edges → Cause oversquashing

Rewiring: Solutions Outlook

Oversquashing Consequences

→ Capturing long-range dependencies are hard

- Rewiring involves modifying the underlying graph structure
- Modification: Changes the graph *connectivity* by addition or removal of edges to *ease* information flow
- Solution 1: A Fully Adjacent Layer
- Solution 2: Geometric GCN
- Solution 3: Rewiring with Positional Encodings
- A Curvature Perspective of Oversquashing
  - Solution 4: Curvature Based Rewiring

A Curvature Based Rewiring Solution

Stochastic Discrete Ricci Flow (SDRF)

- Before training a GNN, rewire the graph as follows
  - Consider a most bottlenecked edge
  - Add an edge that can provide a high improvement to curvature
  - Remove the least bottlenecked edge under certain conditions
  - Repeat
- The above procedure greedily increases curvature and reduces over-squashing by providing better connectivity ☺

Empirical Results

$\mathcal{H}(G)$	Cornell 0.11	Texas 0.06	Wisconsin 0.16	Chameleon 0.25	Squirrel 0.22	Actor 0.24	Cora 0.83	Citeseer 0.71	Pubmed 0.79
None	52.69 ± 0.21	61.19 ± 0.49	54.60 ± 0.86	41.33 ± 0.18	30.32 ± 0.99	23.84 ± 0.43	81.89 ± 0.79	72.31 ± 0.17	78.16 ± 0.23
Undirected	53.20 ± 0.53	63.38 ± 0.87	51.37 ± 1.15	42.02 ± 0.30	35.53 ± 0.78	21.45 ± 0.47	-	-	-
+FA	$\mathbf{58.29 \pm 0.49}$	$\mathbf{64.82 \pm 0.29}$	55.48 ± 0.62	42.67 ± 0.17	36.86 ± 0.44	24.14 ± 0.43	81.65 ± 0.18	70.47 ± 0.18	$\mathbf{79.48 \pm 0.12}$
DIGL (PPR)	58.26 ± 0.50	62.03 ± 0.43	49.53 ± 0.27	42.02 ± 0.13	33.22 ± 0.14	24.77 ± 0.32	$\mathbf{83.21 \pm 0.27}$	$\mathbf{73.29 \pm 0.17}$	78.84 ± 0.08
DIGL + Undirected	$\mathbf{59.54 \pm 0.64}$	63.54 ± 0.38	52.23 ± 0.54	42.68 ± 0.12	32.48 ± 0.23	25.45 ± 0.30	-	-	-
SDRF	54.60 ± 0.39	64.46 ± 0.38	$\mathbf{55.51 \pm 0.27}$	$\mathbf{42.73 \pm 0.15}$	$\mathbf{37.05 \pm 0.17}$	$\mathbf{28.42 \pm 0.75}$	$\mathbf{82.76 \pm 0.23}$	$\mathbf{72.58 \pm 0.20}$	$\mathbf{79.10 \pm 0.11}$
SDRF + Undirected	57.54 ± 0.34	$\mathbf{70.35 \pm 0.60}$	$\mathbf{61.55 \pm 0.86}$	$\mathbf{44.46 \pm 0.17}$	$\mathbf{37.67 \pm 0.23}$	$\mathbf{28.35 \pm 0.06}$	-	-	-

Table 2: Experimental results on common node classification benchmarks. Top two in bold.

Results Taken from [4]

A curvature perspective of Oversquashing - Summary

- Establishes links between geometry and graph topology
- Leads to a very simple algorithm for rewiring
- These links can prove a better understanding using tools from spectral graph theory
- Lots of interesting theoretical directions

Summary: Outlook on Rewiring and Solutions

Oversquashing Consequences

→ Capturing long-range dependencies are hard

- Rewiring involves modifying the underlying graph structure
- Modification: Changes the graph *connectivity* by addition or removal of edges to *ease* information flow
- Solution 1: A Fully Adjacent Layer
- Solution 2: Geometric GCN
- Solution 3: Rewiring with Positional Encodings
- A Curvature Perspective of Oversquashing
  - Solution 4: A Curvature Inspired Rewiring Solution