European Research Council
Established by the European Commission

UNIVERSITY OF OXFORD

Engineering and Physical Sciences Research Council

EPSRC Centre for Doctoral Training in Autonomous Intelligent Machines & Systems

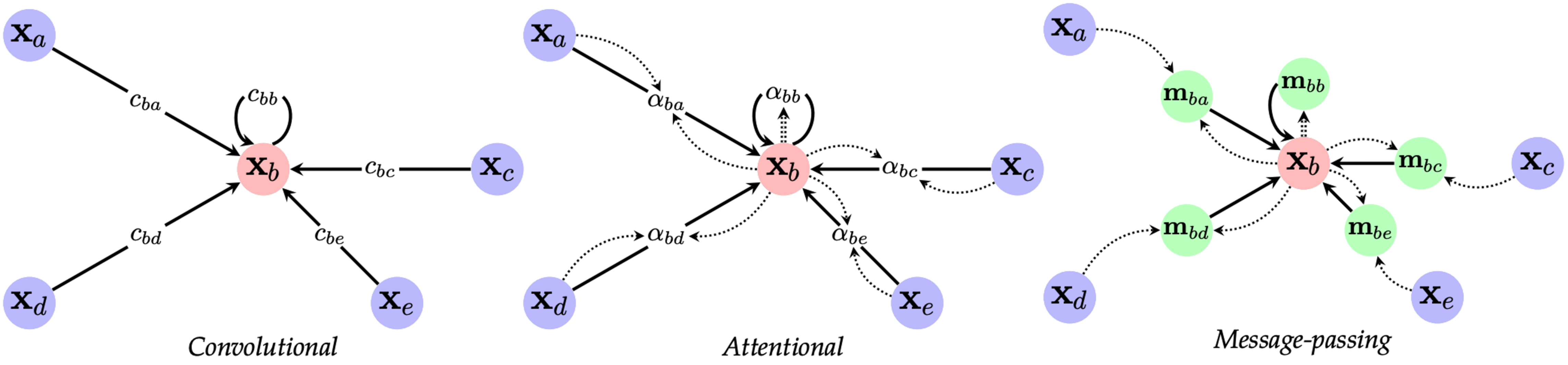
DRew: Dynamically Rewired Message Passing with Delay

Benjamin Gutteridge,
Xiaowen Dong, Michael Bronstein,
Francesco Di Giovanni

Overview

- Background: MPNNs and long-range interactions
- Contributions:
  - Dynamically Rewired Message Passing
  - DRew + **Delay**
- Why DRew works
- Experimental results

Message-Passing Neural Networks



- Message passing: aggregation and update steps
- Occurs over **1-hop neighbourhood**
- Several variations, but most graph neural networks are MPNNs

Figure credit: Bronstein et al 2021. Geometric Deep Learning Grids, Groups, Graphs, Geodesics, and Gauges

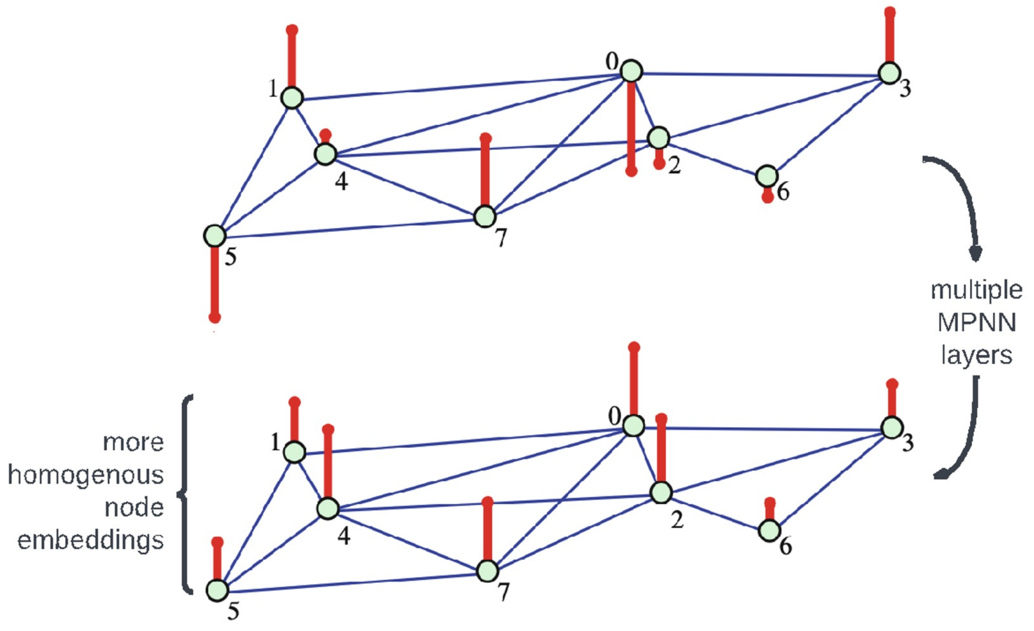
Challenges with MPNNs

- **Long-range dependency**
  - When the output of a MPNN depends on distant nodes interacting with each other

Necessitates more MPNNs layers, leading to:

- Oversmoothing
  - increasing network depth leading to homogeneous node representations and thus poor performance

- Oversquashing
  - “Lack of sensitivity of the output of an MPNN at node p to the input features at an k-hop-distant node s”



↑ Oversmoothing
↓ Oversquashing

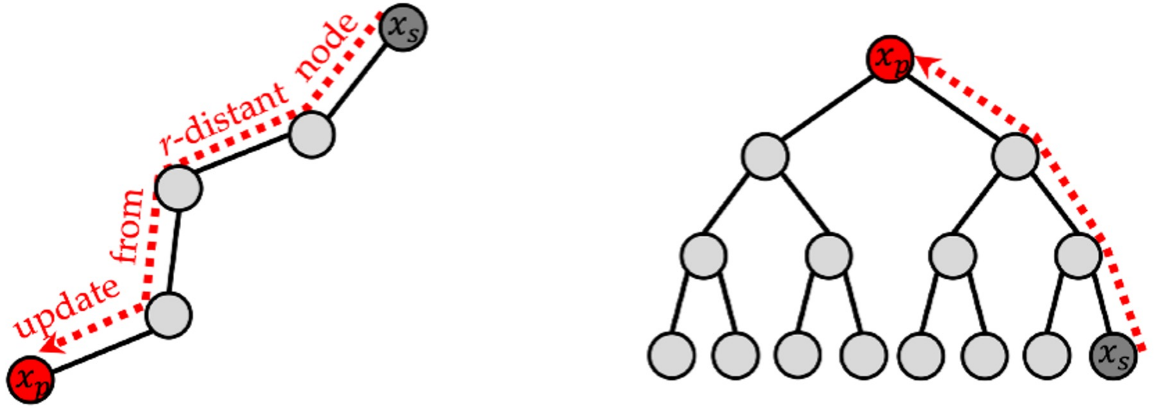


Figure credits: Topping et al 2022. Over-squashing, Bottlenecks, and Graph Ricci curvature (bottom). Stanković, Ljubiša, and Ervin Sejdić, eds. 2019. Vertex-frequency analysis of graph signals (top).

Long-range interactions

- Various domains use global graph information or rely on distant node interactions
- Many large graphs likely exhibit a degree of long-range dependence
- Several recent works looking at long-range interactions, as well as a set of benchmark datasets

  - Wu, Zhanghao, et al. "Representing long-range context for graph neural networks with global attention." (NIPS 2021)
  - Dwivedi, Vijay Prakash, et al. "Long range graph benchmark." (NIPS 2022)
  - Di Giovanni, Francesco, et al. "On over-squashing in message passing neural networks: The impact of width, depth, and topology." (ICML 2023)
  - Ma, Liheng, et al. "Graph Inductive Biases in Transformers without Message Passing." (ICML 2023).

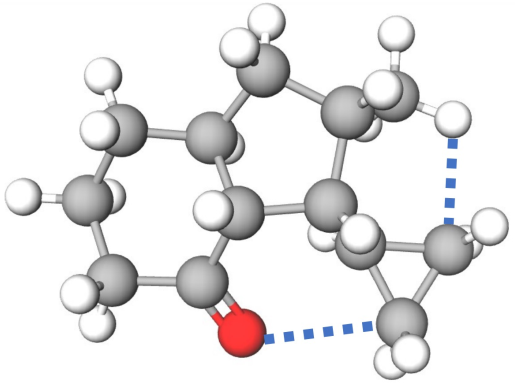


Figure 1: Molecule with LRIs (dotted lines showing 3D atomic contact) that are not trivially captured by the graph structure.

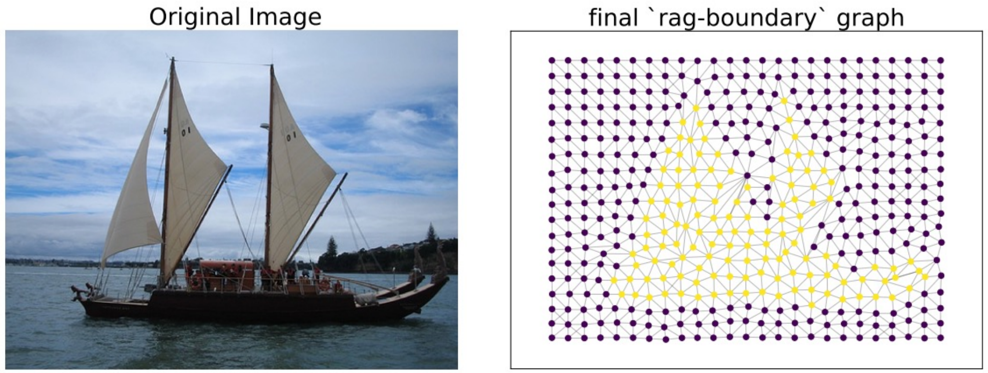
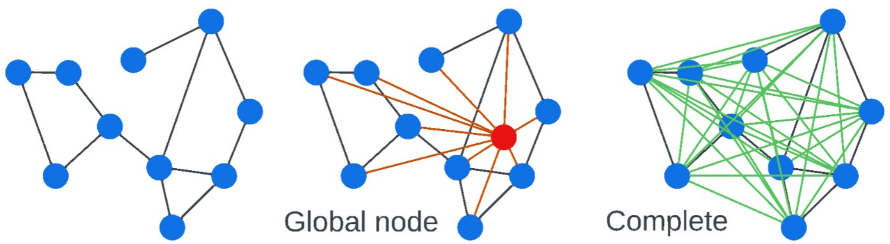


Figure credits: Dwivedi et al 2022, Long Range Graph Benchmark

Graph Rewiring



Static graph rewiring

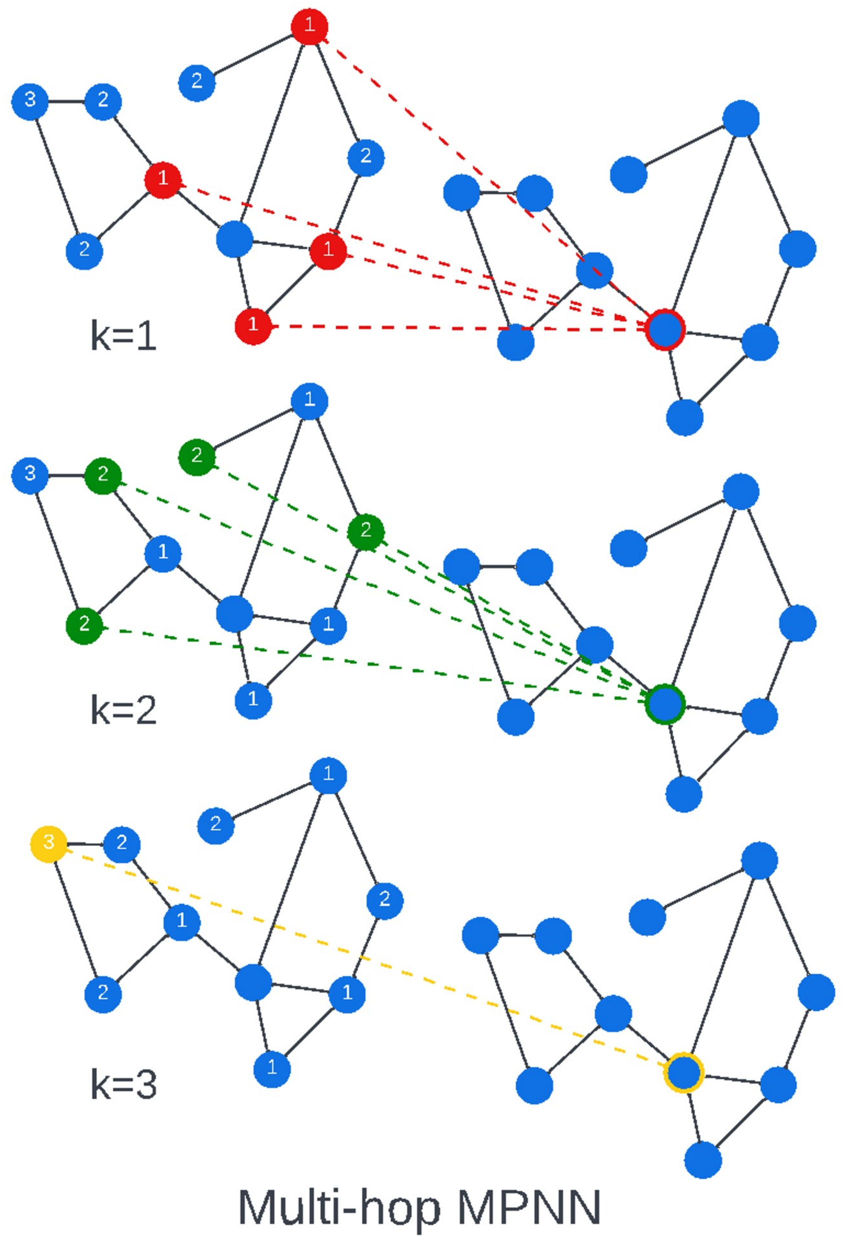
- Graph topology itself is altered to make it 'friendlier'
- E.g.
    - Dropping or adding nodes or edges (DropEdge, DropGNN)
    - Global nodes/fully adjacent layers
    - Rewiring according to a spectral/connectivity measure (SDRF, DIGL)
    - Positional encoding

Computational graph rewiring

- Rather than changing input graph itself, you change the way you *allow information to propagate* during message passing
- E.g.
    - Multi-hop MPNNs (Shortest Path Network, N-GCN, MixHop, k-hop GNN)
    - Graph Transformers
- ⬆️ This is our focus

Proposal

- Transformers throw away the graph topology by making graphs fully-connected
- Multi-hop MPNNs are similar:
  - They make the computational graph denser
  - They lose the notion of information flow through the graph, i.e. that nodes that are closer should interact earlier
- How can we exploit these inductive biases?



Intuition: Dynamic Rewiring

"…aggregating information over distant nodes that goes beyond the limitations of classical MPNNs, but respects the inductive bias provided by the graph: **nodes that are closer should interact earlier in the architecture.**"

"We argue that it is important not simply *how* two node states interact with each other, but also ***when that happens***."

Background: MPNNs

- MPNN:
  - 1-hop local aggregation
  - update

$$a_i^{(\ell)} = \mathrm{AGG}^{(\ell)}\left(\{h_j^{(\ell)} : j \in \mathcal{N}_1(i)\}\right),$$

$$h_i^{(\ell+1)} = \mathrm{UP}^{(\ell)}\left(h_i^{(\ell)}, a_i^{(\ell)}\right),$$

1-hop neighbourhood

- k-hop neighbourhood:

Shortest path distance

$$\mathcal{N}_k(i) := \{j \in V : d_G(i,j) = k\}.$$

Dynamically Rewired MPNN

Vanilla MPNN:

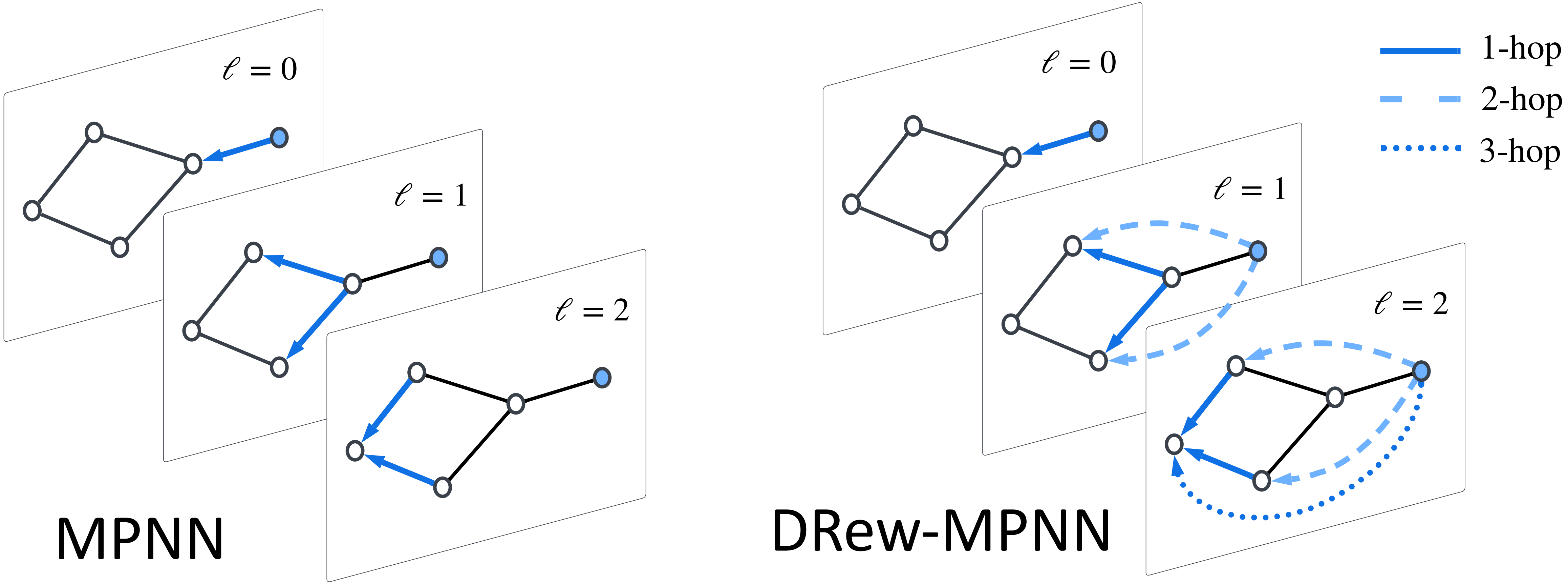
$$a_i^{(\ell)} = \mathrm{AGG}^{(\ell)}\left(\{h_j^{(\ell)} : j \in \mathcal{N}_1(i)\}\right),$$
$$h_i^{(\ell+1)} = \mathrm{UP}^{(\ell)}\left(h_i^{(\ell)}, a_i^{(\ell)}\right),$$

Separate aggregation for each k-hop neighbourhood

$$a_{i,k}^{(\ell)} = \mathrm{AGG}_k^{(\ell)}\left(\{h_j^{(\ell)} : j \in \mathcal{N}_k(i)\}\right), 1 \leq k \leq \ell + 1$$
$$h_i^{(\ell+1)} = \mathrm{UP}^{(\ell)}\left(h_i^{(\ell)}, a_{i,1}^{(\ell)}, \ldots, a_{i,\ell+1}^{(\ell)}\right). \quad (5)$$

Reduces to vanilla MPNN if $\mathrm{AGG}_k = I$ for $k > 1$

$(\ell + 1)$th hop only aggregated from layer ℓ



$$a_i^{(\ell)} = \mathrm{AGG}^{(\ell)}\left(\{h_j^{(\ell)} : j \in \mathcal{N}_1(i)\}\right),$$
$$h_i^{(\ell+1)} = \mathrm{UP}^{(\ell)}\left(h_i^{(\ell)}, a_i^{(\ell)}\right),$$

$$a_{i,k}^{(\ell)} = \mathrm{AGG}_k^{(\ell)}\left(\{h_j^{(\ell)} : j \in \mathcal{N}_k(i)\}\right), 1 \leq k \leq \ell + 1$$
$$h_i^{(\ell+1)} = \mathrm{UP}_k^{(\ell)}\left(h_i^{(\ell)}, a_{i,1}^{(\ell)}, \ldots, a_{i,\ell+1}^{(\ell)}\right). \quad (5)$$

Introducing delay

Currently:

- MPNNs: nodes i, j interact with a constant delay given by their distance – leading to the same lag of information
- DRew: nodes interact only from a certain depth of the architecture, but without any delay

What if we consider the state of j as it was when the information 'left' to flow towards i?

Introducing delay: νDRew

- What if we consider the state of j as it was when the information 'left' to flow towards i?
- Delay: $$\tau_\nu(k) = \max(0, k - \nu)$$

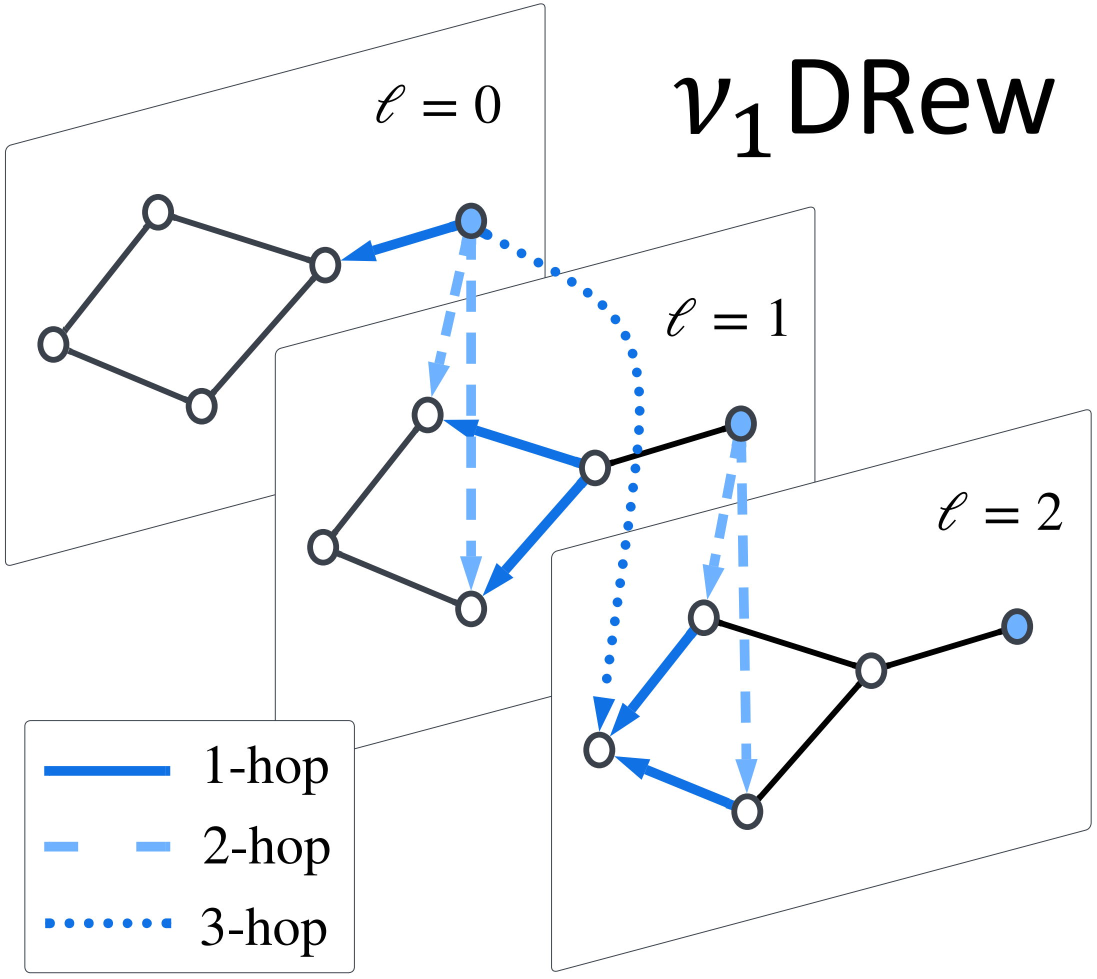
k: current k-hop

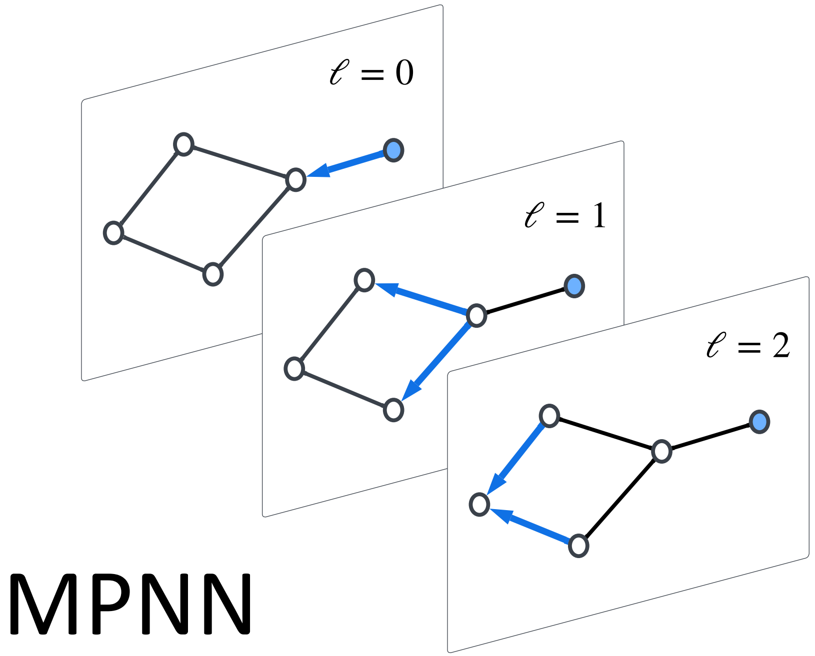
ν: 'rate' hyperparameter

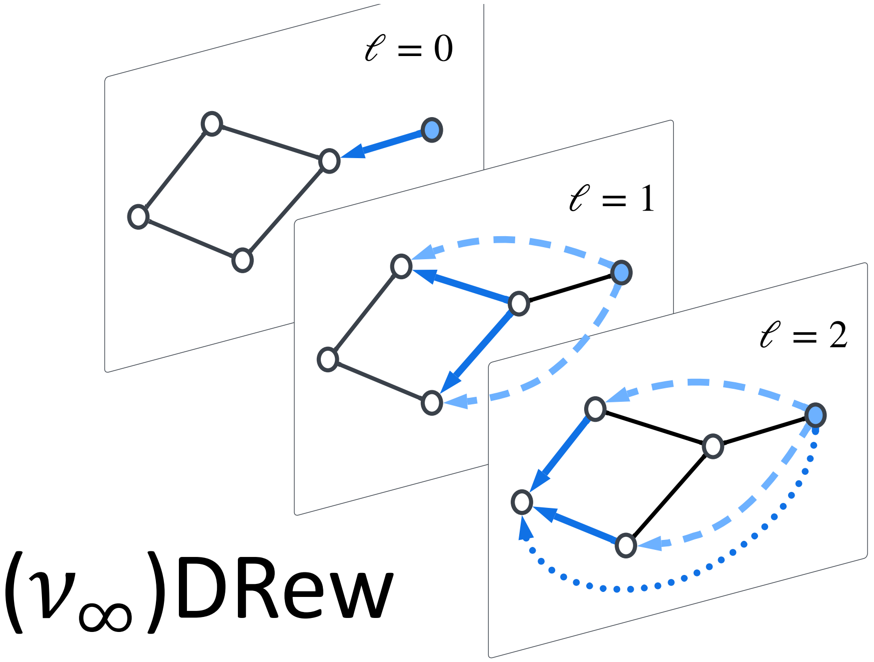
➢ (i.e. the hop radius below which node communication is instantaneous)

$$a_{i,k}^{(\ell)} = \mathrm{AGG}_k^{(\ell)}\left(\{h_j^{(\ell-\tau_\nu(k))} : j \in \mathcal{N}_k(i)\}\right), 1 \leq k \leq \ell + 1$$

$$h_i^{(\ell+1)} = \mathrm{UP}_k^{(\ell)}\left(h_i^{(\ell)}, a_{i,1}^{(\ell)}, \ldots, a_{i,\ell+1}^{(\ell)}\right). \quad (6)$$

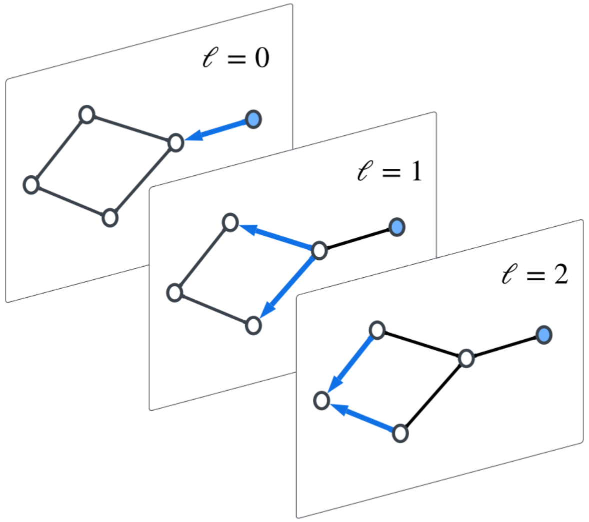






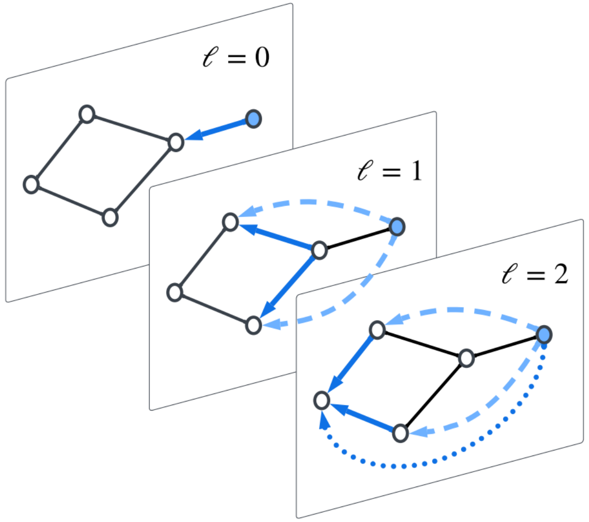
$$
\begin{aligned}
a_{i,k}^{(\ell)} &= \mathrm{AGG}_k^{(\ell)}\Big(\{h_j^{(\ell-\tau_\nu(k))} : j \in \mathcal{N}_k(i)\}\Big), 1 \leq k \leq \ell + 1 \\
h_i^{(\ell+1)} &= \mathrm{UP}_k^{(\ell)}\Big(h_i^{(\ell)}, a_{i,1}^{(\ell)}, \ldots, a_{i,\ell+1}^{(\ell)}\Big).
\end{aligned}
\tag{6}
$$

The graph-rewiring perspective: νDRew as distance-aware skip connections



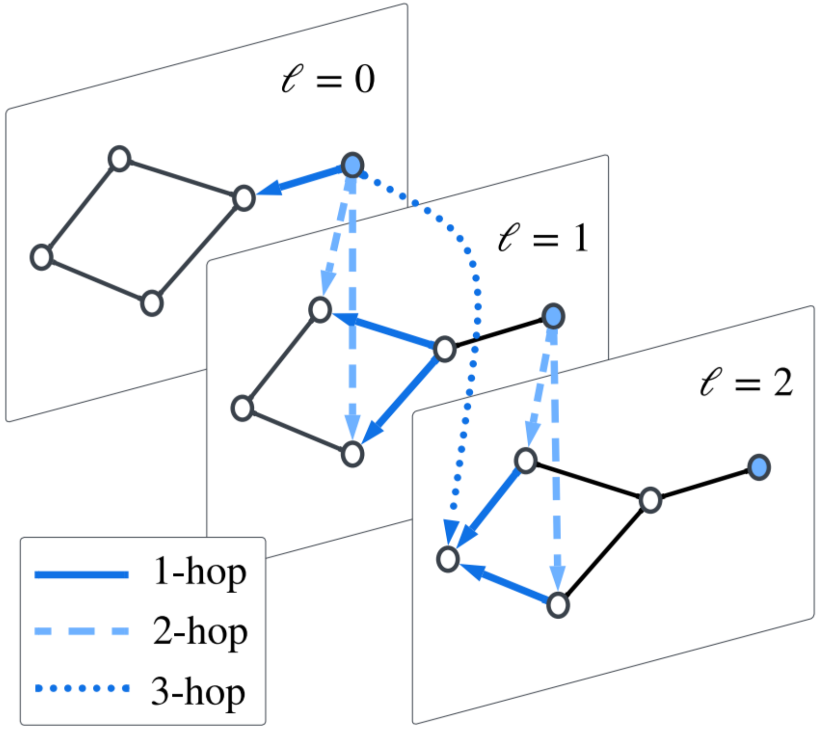
(a) Classical MPNN

- 1-hop, horizontal only



(b) DRew

- Multi-hop, horizontal only
- Computational graph gradually filled



(c) νDRew

- Multi-hop, horizontal AND vertical skip connections, through distance and time (layer)
- Skip connections between *different* nodes, dependent on geometric distance

DRew instantiations of common MPNNs

- GCN

$$h_i^{(\ell+1)} = h_i^{(\ell)} + \sigma\left(\sum_{k=1}^{\ell+1}\sum_{j\in\mathcal{N}_k(i)} \mathbf{W}_k^{(\ell)}\gamma_{ij}^k h_j^{(\ell-\tau_\nu(k))}\right)$$

$$\gamma_{ij}^k = \begin{cases}\frac{1}{\sqrt{d_i d_j}}, & \text{if } d_G(i,j)=k\\ 0, & \text{otherwise.}\end{cases}$$

- GIN

$$h_i^{(\ell+1)} = (1+\epsilon)\mathrm{MLP}_s^{(\ell)}(h_i^{(\ell)}) + \sum_{k=1}^{\ell+1}\sum_{j\in\mathcal{N}_k(i)} \mathrm{MLP}_k^{(\ell)}(h_j^{(\ell-\tau_\nu(k))}),$$

- GatedGCN

$$h_i^{(\ell+1)} = \mathbf{W}_1^{(\ell)}h_i^{(\ell)} + \sum_{k=1}^{\ell+1}\sum_{j\in\mathcal{N}_k(i)} \eta_{i,j}^k \odot \mathbf{W}_2^{(\ell)}h_j^{(\ell-\tau_\nu(k))}$$

$$\eta_{i,j}^k = \frac{\hat{\eta}_{i,j}^k}{\sum_{j\in\mathcal{N}_k(i)}(\hat{\eta}_{i,j}^k)+\epsilon},$$

$$\hat{\eta}_{i,j}^k = \sigma\left(\mathbf{W}_3^{(\ell)}h_i^{(\ell)} + \mathbf{W}_4^{(\ell)}h_j^{(\ell-\tau_\nu(k))}\right)$$

Why does νDRew help with over-squashing?

- Jacobian as a measure of sensitivity between nodes (Topping 2022)
- For vanilla MPNN, same adjacency $\boldsymbol{A}$ used in each layer (i.e. 1-hop aggregation) with which we can bound the Jacobian by power $\boldsymbol{A}^r$ for nodes i, j at hop distance r
- Due to skip connections, ν_1DRew-GCN's sensitivity bound is different – see below
- Nodes at distance r can now interact via products of message-passing matrices containing fewer than r factors
- Oversquashing arises due to the entries i, j of normalised $\boldsymbol{A}^r$ decaying to zero exponentially with r
- Powers of Γ^k ($\gamma_{i,j} \in \Gamma$) are different unlike $\boldsymbol{A}$, therefore oversquashing is mitigated

$$\left|\frac{\partial h_i^{(r)}}{\partial h_j^{(0)}}\right| \leq c\,(\mathbf{A}^r)_{ij},$$

$$\left|\frac{\partial h_i^{(r)}}{\partial h_j^{(0)}}\right| \leq C\Big(\sum_{k_1+\cdots+k_\ell=r}\Big(\prod_{k_1,\ldots,k_\ell}(\gamma^k)_{ij}\Big)\Big)$$

Why does νDRew help with over-smoothing?

- Over-smoothing occurs because by the time information from node i reaches distant node j, it has been mixed many times with neighbours
- Skip connections with delay allow i to 'see' j before too much local smoothing has occurred
- Choice of delay parameter ν can be considered amount of local smoothing
  - High ν: more local smoothing
  - Low ν: less

Experiments

- Long-range graph benchmark
  - Chemistry and computer vision
  - Graph-, node- and edge-level tasks
- QM9 (see paper)
  - Chemistry, multi-task regression
- RingTransfer
  - Synthetic ‘true’ long-range task
- Peptides-func ablation
  - Demonstrate impact of delay parameter ν for for task from LRGB

Performance on real-world datasets

Table 1. Classical MPNN benchmarks vs their DRew variants (without positional encoding) across four LRGB tasks: (from left to right) graph classification, graph regression, link prediction and node classification. All results are for the given metric on test data.

Model	Peptides-func AP ↑	Peptides-struct MAE ↓	PCQM-Contact MRR ↑	PascalVOC-SP F1 ↑
GCN	0.5930±0.0023	0.3496±0.0013	0.3234±0.0006	0.1268±0.0060
+DRew	**0.6996±0.0076**	**0.2781±0.0028**	**0.3444±0.0017**	**0.1848±0.0107**
GINE	0.5498±0.0079	0.3547±0.0045	0.3180±0.0027	0.1265±0.0076
+DRew	**0.6940±0.0074**	**0.2882±0.0025**	**0.3300±0.0007**	**0.2719±0.0043**
GatedGCN	0.5864±0.0077	0.3420±0.0013	0.3218±0.0011	0.2873±0.0219
+DRew	**0.6733±0.0094**	**0.2699±0.0018**	**0.3293±0.0005**	**0.3214±0.0021**

- Tasks from long-range graph benchmark; 4 different tasks
- DRew models **consistently beat their non-DRew counterparts**
- Fixed parameter budget of 500k
- Better performance even though *no edge features used in DRew*
  - for simplicity; we would expect use of edge features to further improve results

Table 2. Performance of various classical, multi-hop and static rewiring MPNN and graph Transformer benchmarks against DRew-MPNNs across four LRGB tasks. The **first-**, **second-** and **third-**best results for each task are colour-coded; models whose performance are within a standard deviation of one another are considered equal.

Model	Peptides-func AP ↑	Peptides-struct MAE ↓	PCQM-Contact MRR ↑	PascalVOC-SP F1 ↑
GCN	0.5930±0.0023	0.3496±0.0013	0.3234±0.0006	0.1268±0.0060
GINE	0.5498±0.0079	0.3547±0.0045	0.3180±0.0027	0.1265±0.0076
GatedGCN	0.5864±0.0077	0.3420±0.0013	0.3218±0.0011	0.2873±0.0219
GatedGCN+PE	0.6069±0.0035	0.3357±0.0006	0.3242±0.0008	0.2860±0.0085
DIGL+MPNN	0.6469±0.0019	0.3173±0.0007	0.1656±0.0029	0.2824±0.0039
DIGL+MPNN+LapPE	0.6830±0.0026	**0.2616±0.0018**	0.1707±0.0021	0.2921±0.0038
MixHop-GCN	0.6592±0.0036	0.2921±0.0023	0.3183±0.0009	0.2506±0.0133
MixHop-GCN+LapPE	0.6843±0.0049	**0.2614±0.0023**	0.3250±0.0010	0.2218±0.0174
Transformer+LapPE	0.6326±0.0126	**0.2529±0.0016**	0.3174±0.0020	0.2694±0.0098
SAN+LapPE	0.6384±0.0121	0.2683±0.0043	**0.3350±0.0003**	**0.3230±0.0039**
GraphGPS+LapPE	0.6535±0.0041	**0.2500±0.0005**	0.3337±0.0006	**0.3748±0.0109**
DRew-GCN	**0.6996±0.0076**	0.2781±0.0028	**0.3444±0.0017**	0.1848±0.0107
DRew-GCN+LapPE	**0.7150±0.0044**	**0.2536±0.0015**	**0.3442±0.0006**	0.1851±0.0092
DRew-GIN	**0.6940±0.0074**	0.2799±0.0016	0.3300±0.0007	0.2719±0.0043
DRew-GIN+LapPE	**0.7126±0.0045**	**0.2606±0.0014**	**0.3403±0.0035**	0.2692±0.0059
DRew-GatedGCN	0.6733±0.0094	0.2699±0.0018	0.3293±0.0005	**0.3214±0.0021**
DRew-GatedGCN+LapPE	**0.6977±0.0026**	**0.2539±0.0007**	0.3324±0.0014	**0.3314±0.0024**

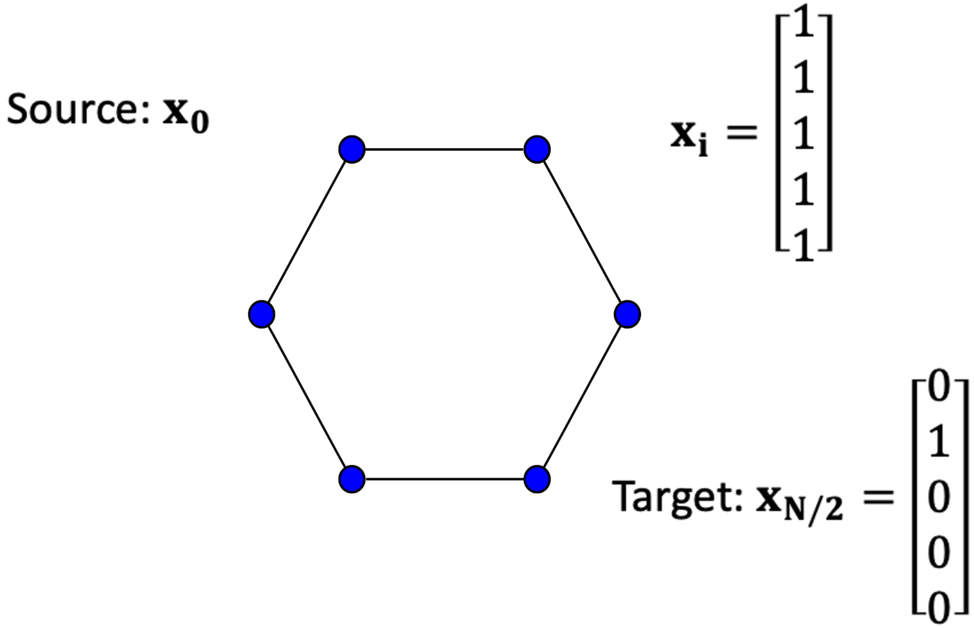
Static rewiring benchmark

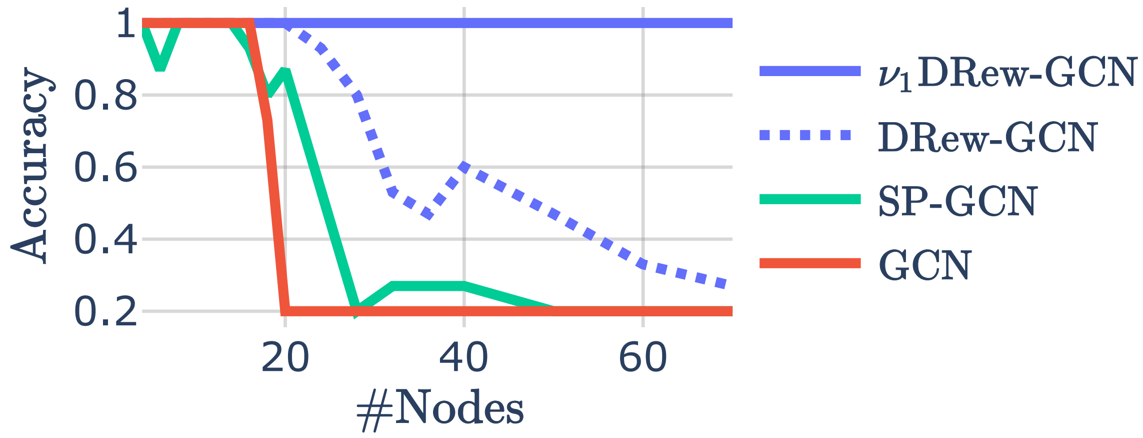
Multi-hop MPNN benchmark

DRew mostly beating or on-par with Transformers

RingTransfer

- Synthetic task for testing LR dependence
- N rings, length n
- Target node must interact with source node $n/2$ hops away
- Fixed $n/2$ layers (needed for interaction)
- $C = 5$ classes
- MPNN/multi-hop MPNN < Drew < Drew + Delay

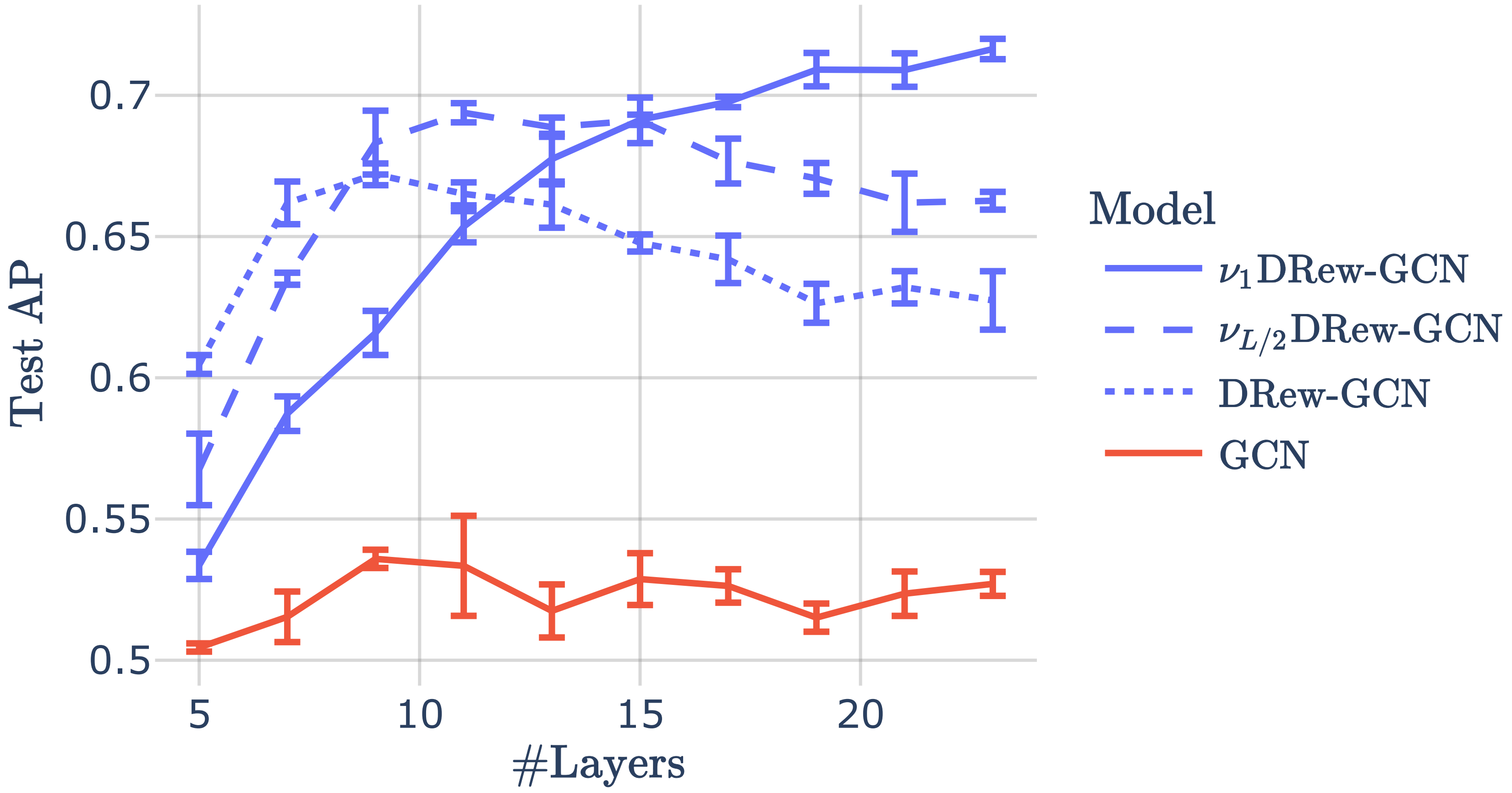




- MPNN << SP-GCN (multi-hop MPNN) << DRew << DRew + *Delay*

Fixed d ablation on peptides-func

- Looking at effect of delay hyperparam
- Param constraint lifted
- Delay reduces impact of oversmoothing
- With full delay, performance *improves* with more layers. Very unusual for MPNNs



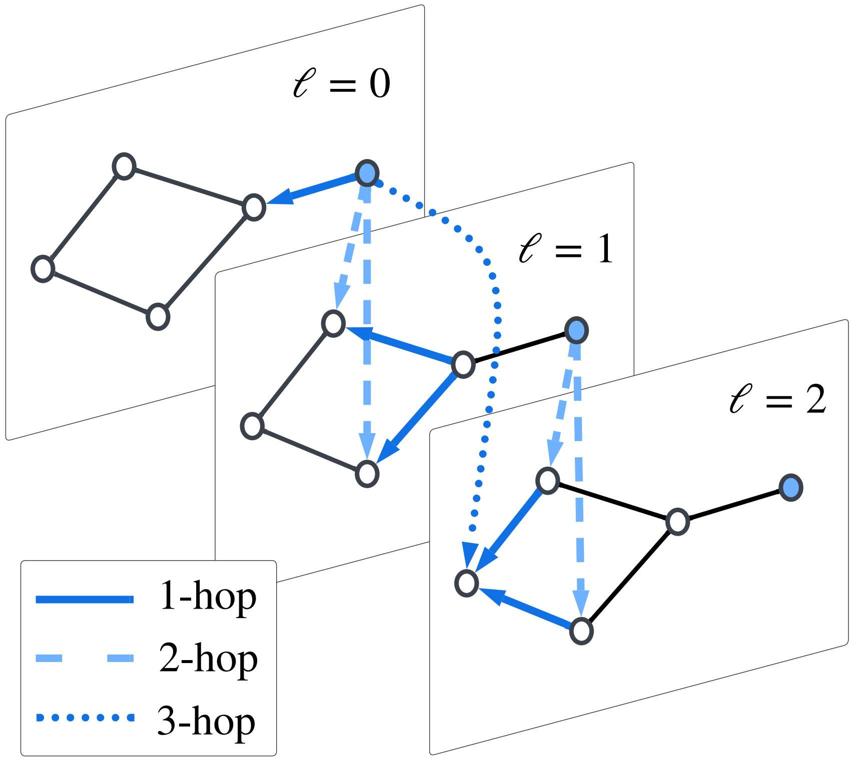
Conclusion

- Two contributions: **D**ynamically **Rew**ired message passing and **Delay**
- Framework applicable to any MPNN
- Reduces over-smoothing and over-squashing
- Improves on vanilla/multi-hop MPNNs, static rewiring approaches and Transformers for synthetic and real-world long-range tasks

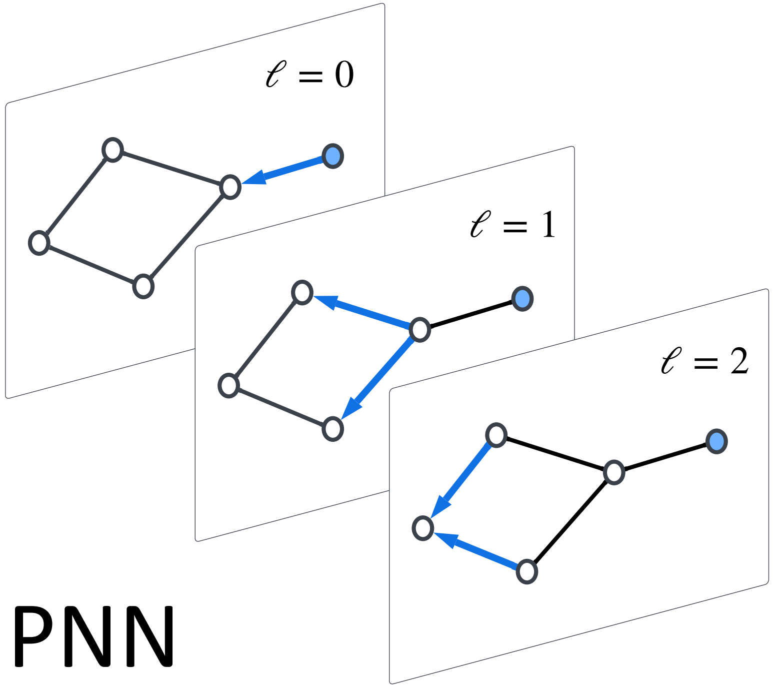
Future Work

- Investigating expressive power
- Reduce parameter scaling (good progress already on this front!)
- Alternate distance measures

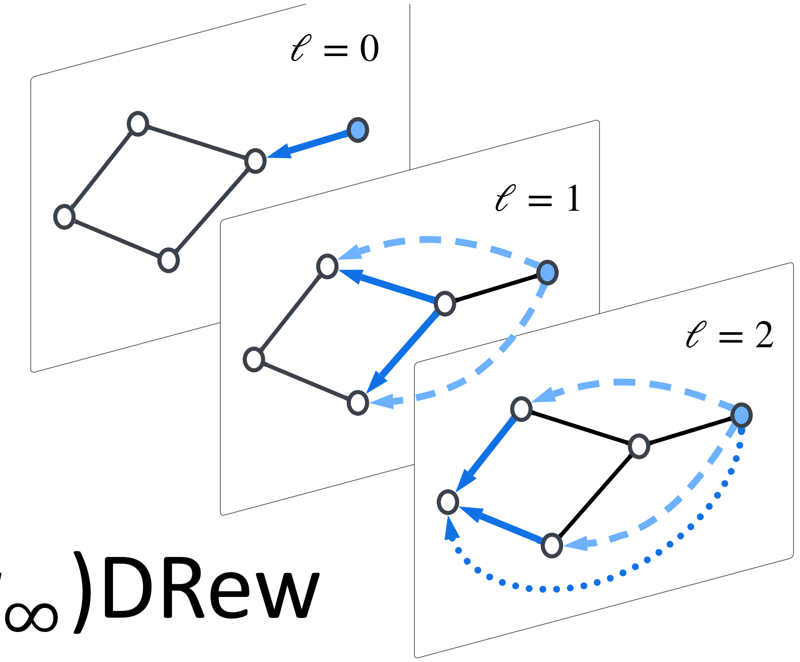
Thanks for watching!



ν_1DRew



MPNN



(ν_∞)DRew

$$a_{i,k}^{(\ell)} = \mathrm{AGG}_k^{(\ell)}\left(\{h_j^{(\ell-\tau_\nu(k))} : j \in \mathcal{N}_k(i)\}\right), 1 \leq k \leq \ell + 1$$
$$h_i^{(\ell+1)} = \mathrm{UP}_k^{(\ell)}\left(h_i^{(\ell)}, a_{i,1}^{(\ell)}, \ldots, a_{i,\ell+1}^{(\ell)}\right). \qquad (6)$$