

Tackling Over-smoothing and Over-squashing in Deep Learning on Graphs

“Making Decisions with AI” hosted by Yoontae Hwang (황윤태)

Jeongwhan Choi (최정환)
Postdoctoral Researcher

KAIST



BigDyL

[big di:l]

Big Data Analytics & Learning Laboratory
Korea Advanced Institute of Science & Technology

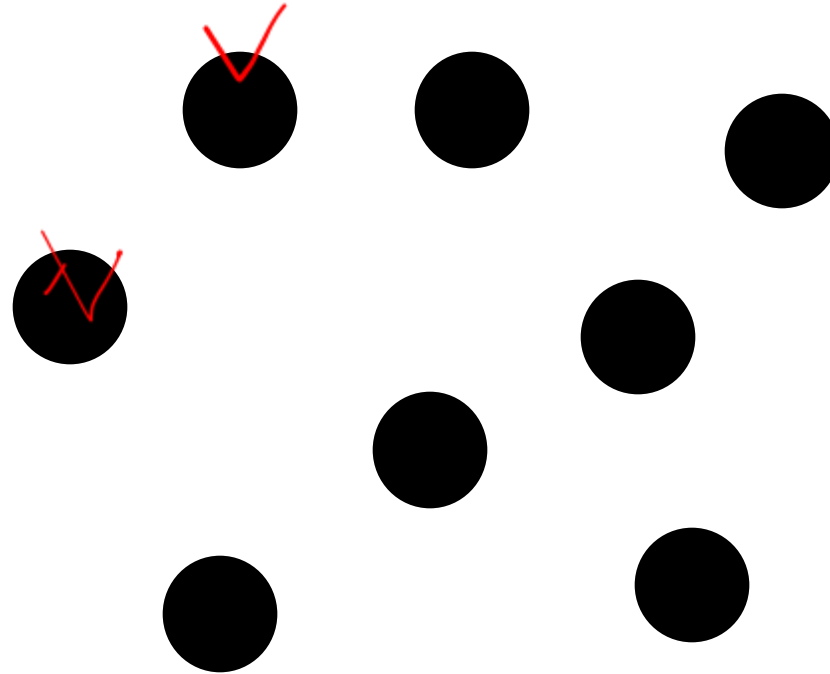
Contents

- Introduction
- Graph Neural Networks
- Overcoming the over-smoothing problem
- Overcoming the over-squashing problem

Why Graphs?

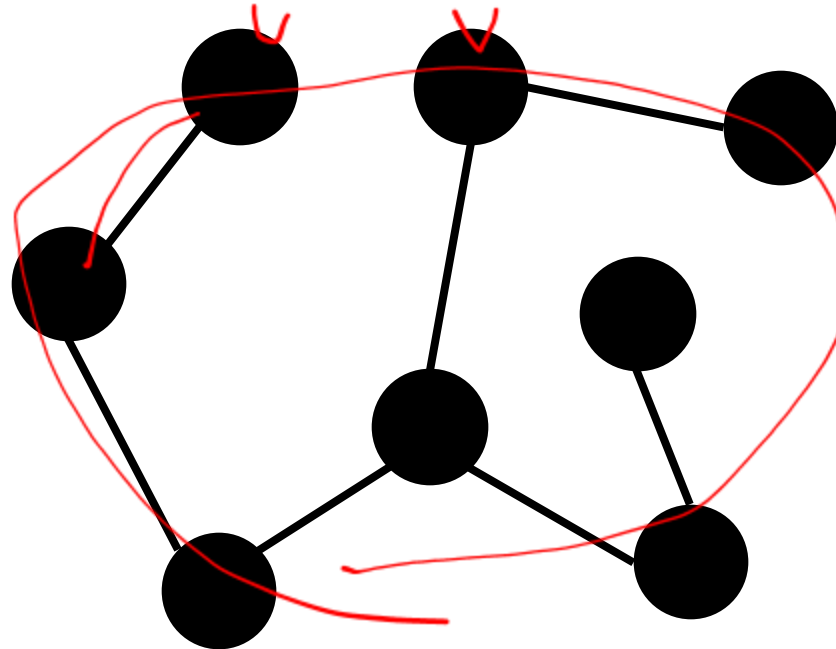
Why Graphs?

- Graphs are a general language for describing and analyzing entities with relationships/interactions



Why Graphs? – Contd.

- Graphs are a general language for describing and analyzing entities with relationships/interactions



Why Graphs? – Contd.



Image credit: [Medium](#)

Social Networks

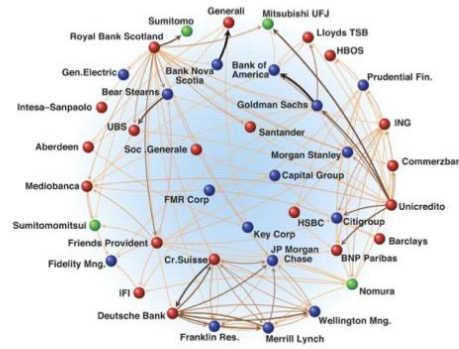


Image credit: [Science](#)

Economic Networks



Image credit: [Lumen Learning](#)

Communication Networks



Citation Networks



Image credit: [Missoula Current News](#)

Internet

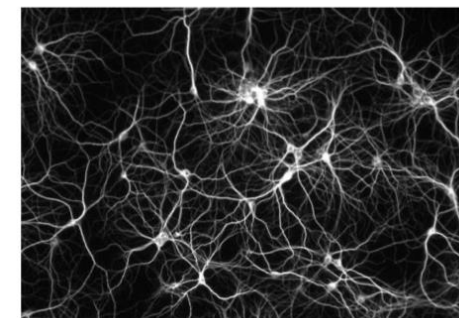
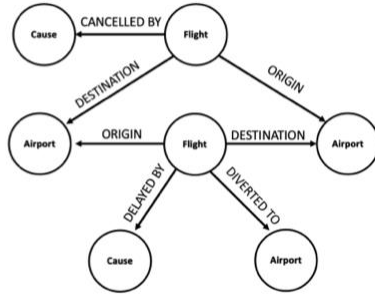


Image credit: [The Conversation](#)

Networks of Neurons

Why Graphs? – Contd.

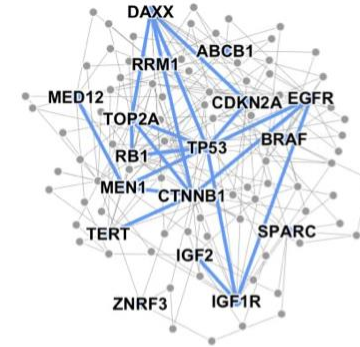


Event Graphs



Image credit: [SalientNetworks](#)

Computer Networks



Disease Pathways

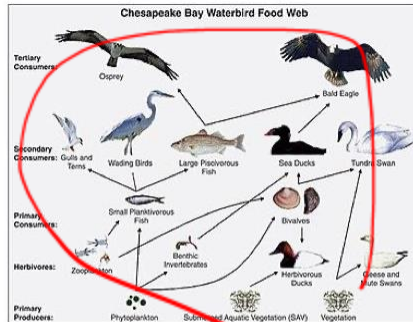


Image credit: [Wikipedia](#)

Food Webs



Image credit: [Pinterest](#)

Particle Networks



Image credit: [visitlondon.com](#)

Underground Networks

Why Graphs? – Contd.

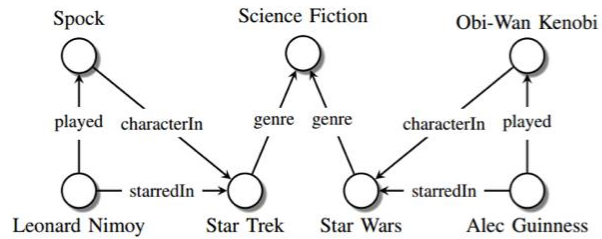


Image credit: [Maximilian Nickel et al](#)

Knowledge Graphs

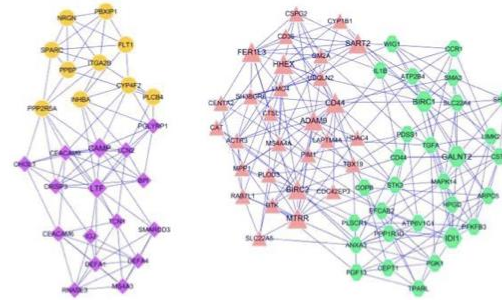


Image credit: [ese.wustl.edu](#)

Regulatory Networks

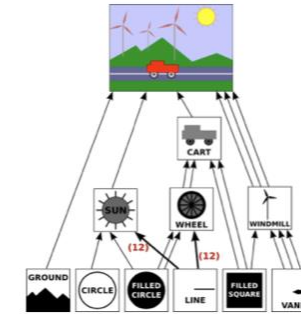


Image credit: [math.hws.edu](#)

Scene Graphs

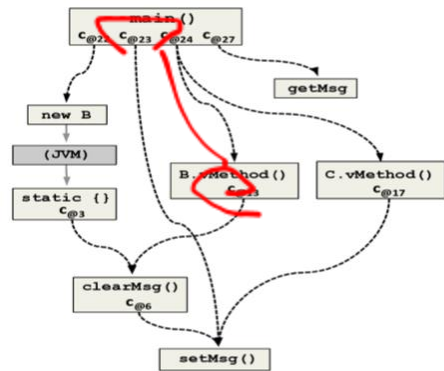


Image credit: [ResearchGate](#)

Code Graphs

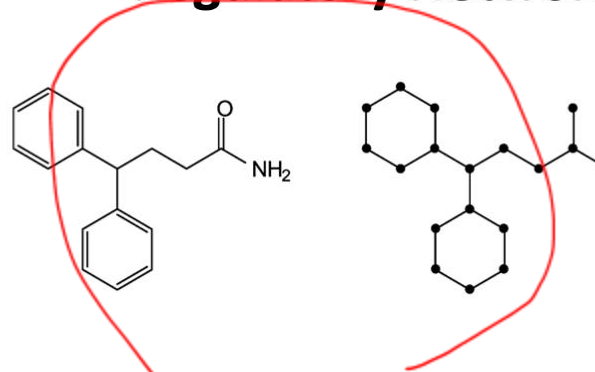


Image credit: [MDPI](#)

Molecules

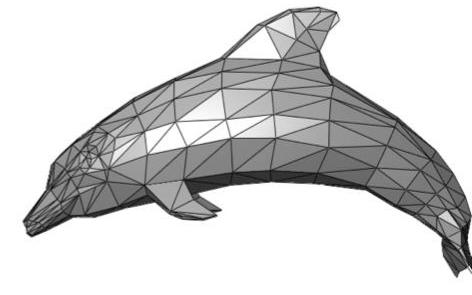
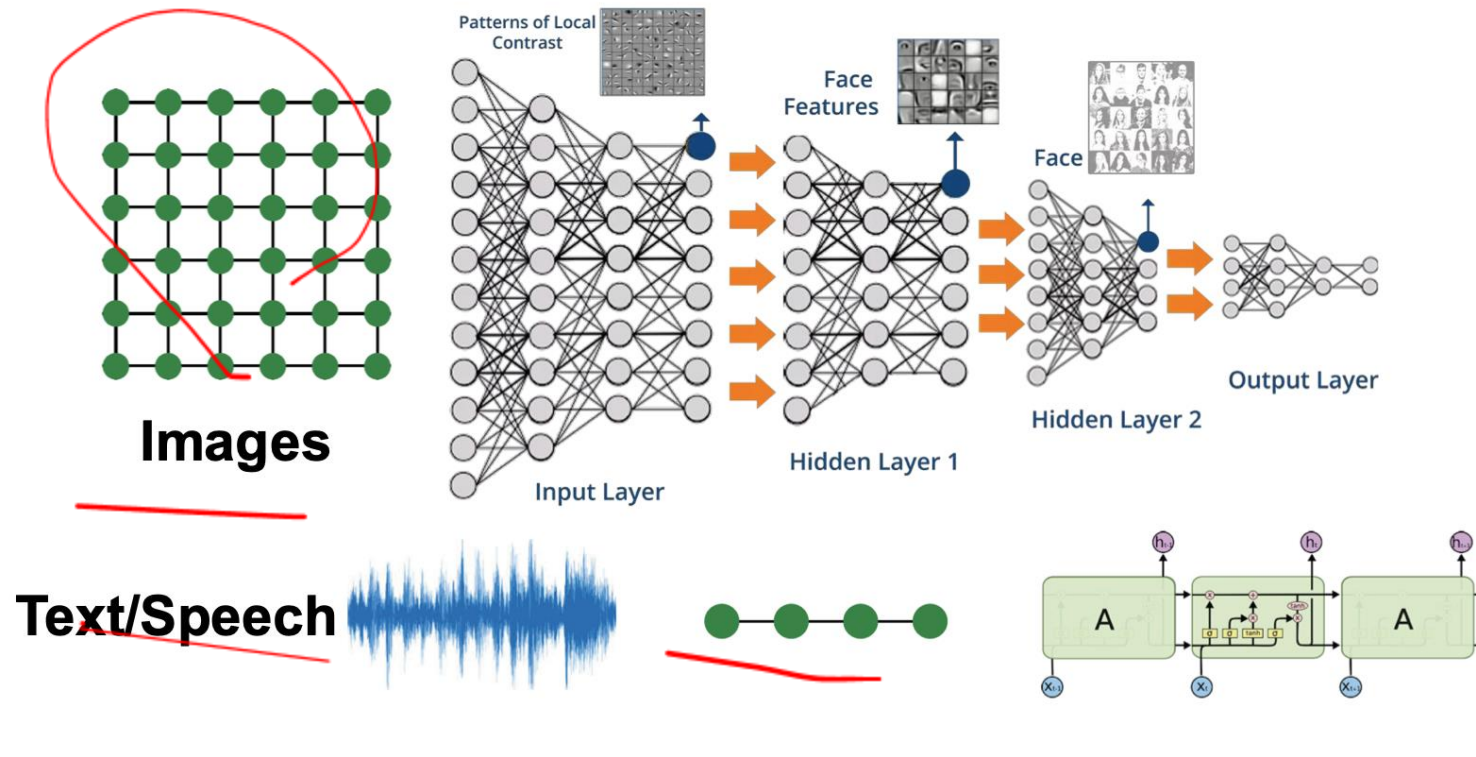


Image credit: [Wikipedia](#)

3D Shapes

Modern ML Toolbox

- Modern deep learning toolbox is designed for sequences & grids



Doubt thou the stars are fire,
Doubt that the sun doth move,
Doubt truth to be a liar,
But never doubt I love...

Text



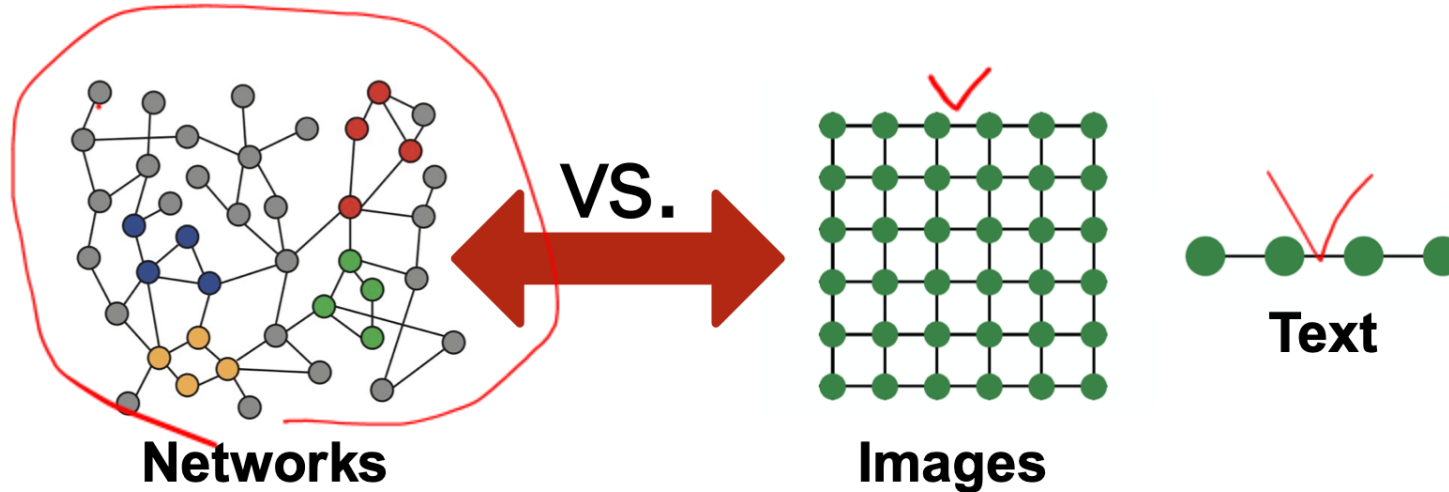
Audio signals



Images

Why is Graph Deep Learning Hard?

- Networks are complex.
 - Arbitrary size and complex topological structure (i.e., no spatial locality like grids)

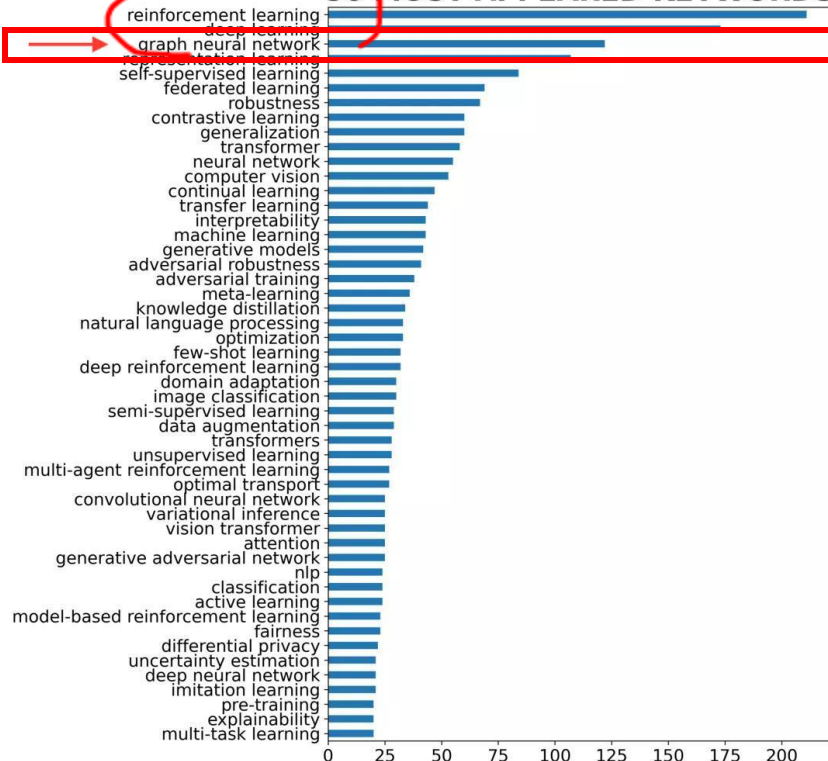


- No fixed node ordering or reference point
- Often dynamic and have multimodal features

Graphs are the New Frontier of Deep Learning

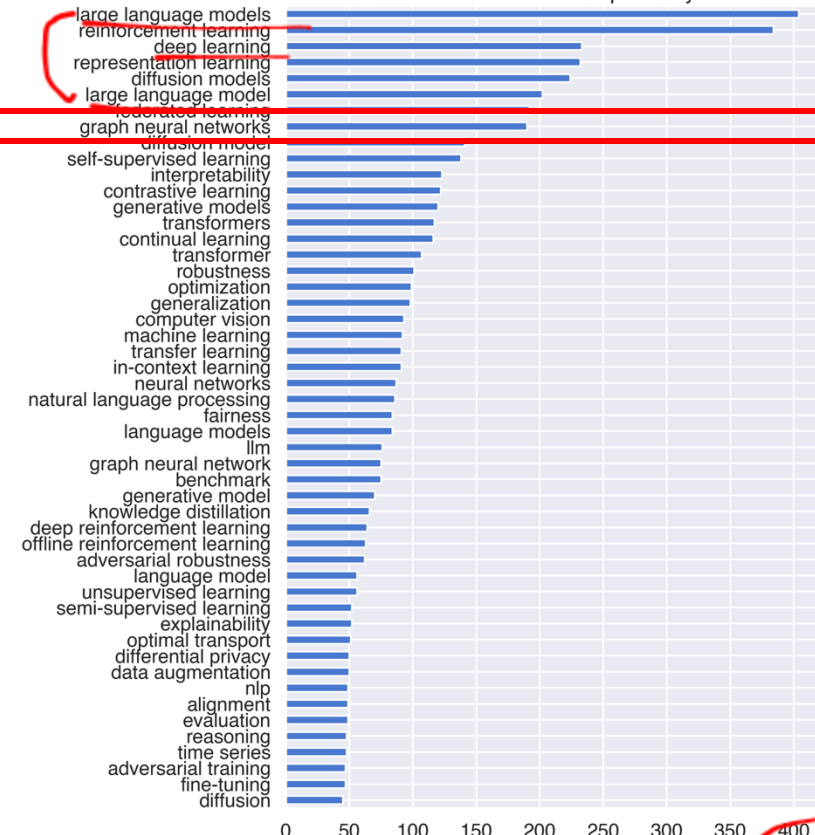
- How can we develop neural networks that are more broadly applicable?

50 MOST APPEARED KEYWORDS



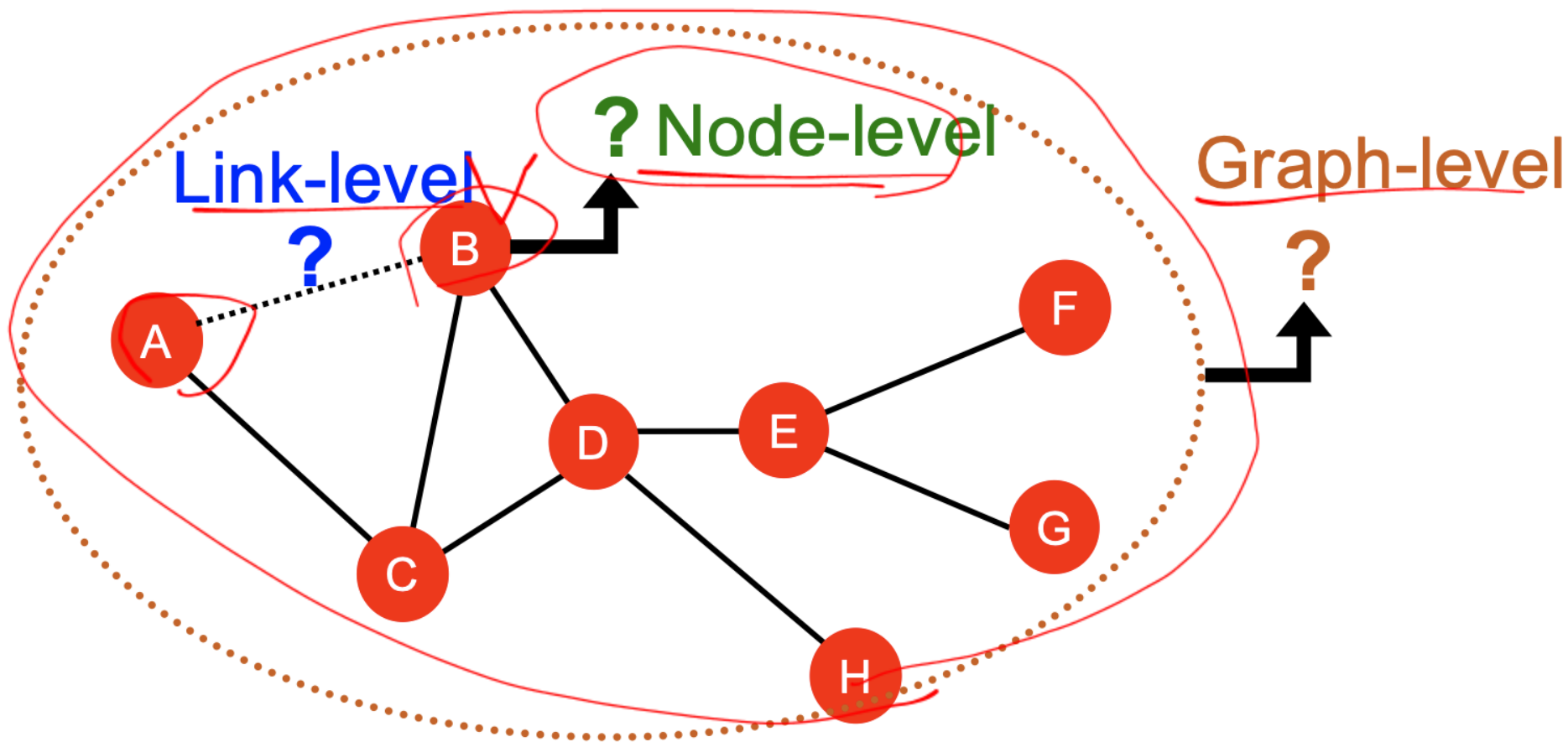
Top 50 keywords in submitted research papers at ICLR 2022 ([source](#))

ICLR 2024 Submission Top 50 Keywords

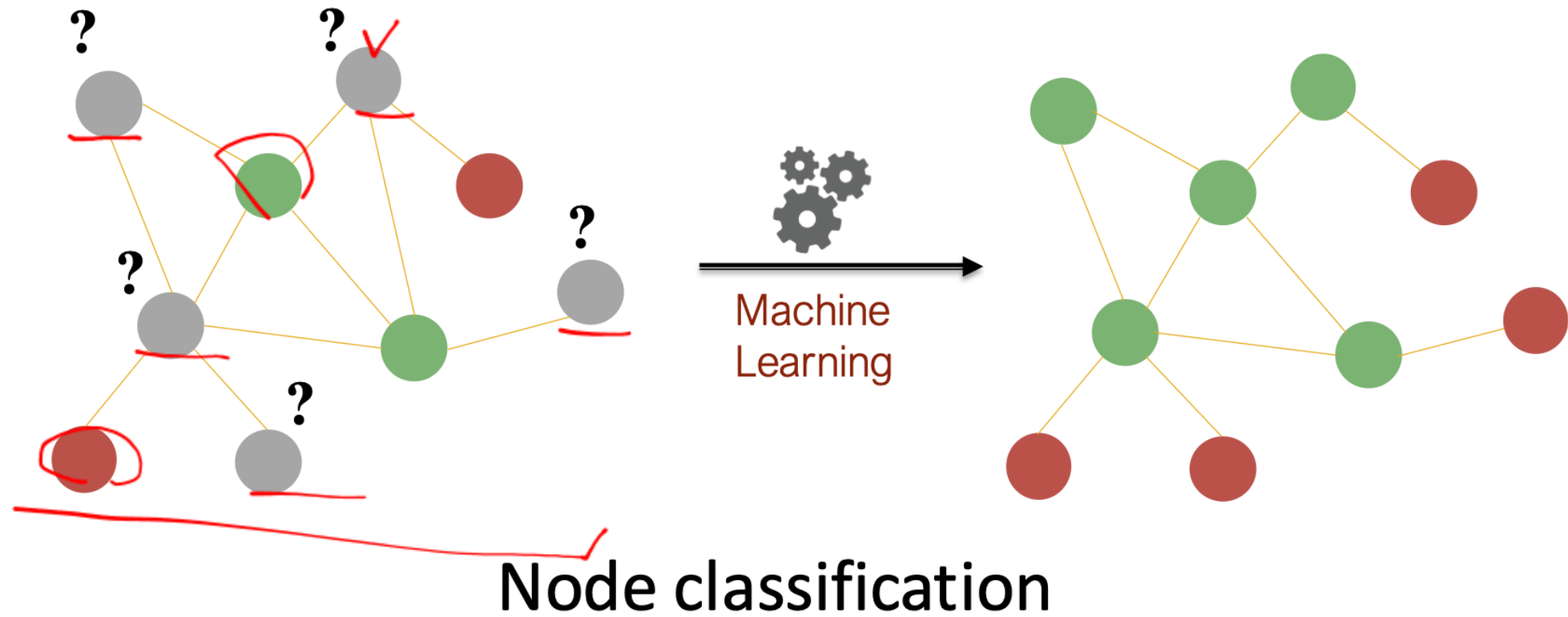


Top 50 keywords in submitted research papers at ICLR 2024 ([source](#))

Tasks on Graphs

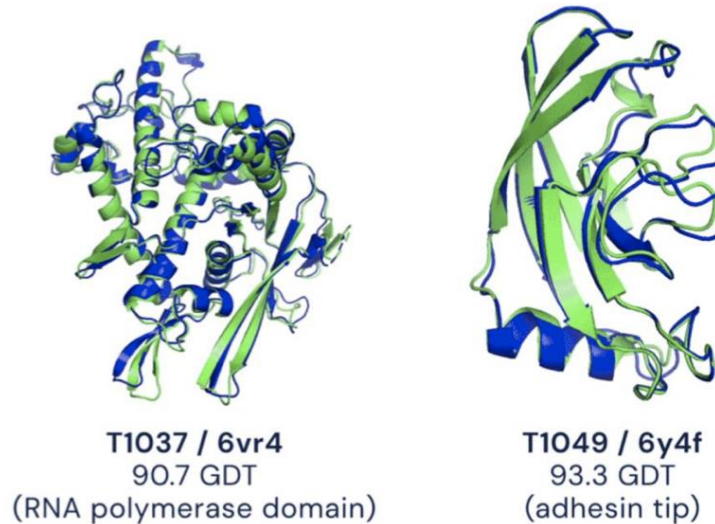


Node-Level Task



Example

- Computationally predict a protein's 3D structure based solely on its amino acid sequence:
 - For each node predict its 3D coordinates



● Experimental result
● Computational prediction

AlphaFold: Impact

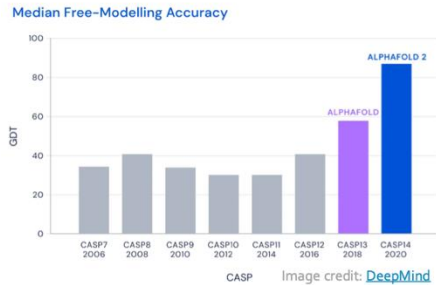


Image credit: SingularityHub

AlphaFold's AI could change the world of biological science as we know it

DeepMind's latest AI breakthrough can accurately predict the way proteins fold

Oct. 9, 2024

SCIENCE

Nobel Prize in Chemistry Goes to 3 Scientists for Predicting and Creating Proteins

The Nobel, awarded to David Baker of the University of Washington and Demis Hassabis and John M. Jumper of Google DeepMind, is the second this week to involve artificial intelligence.

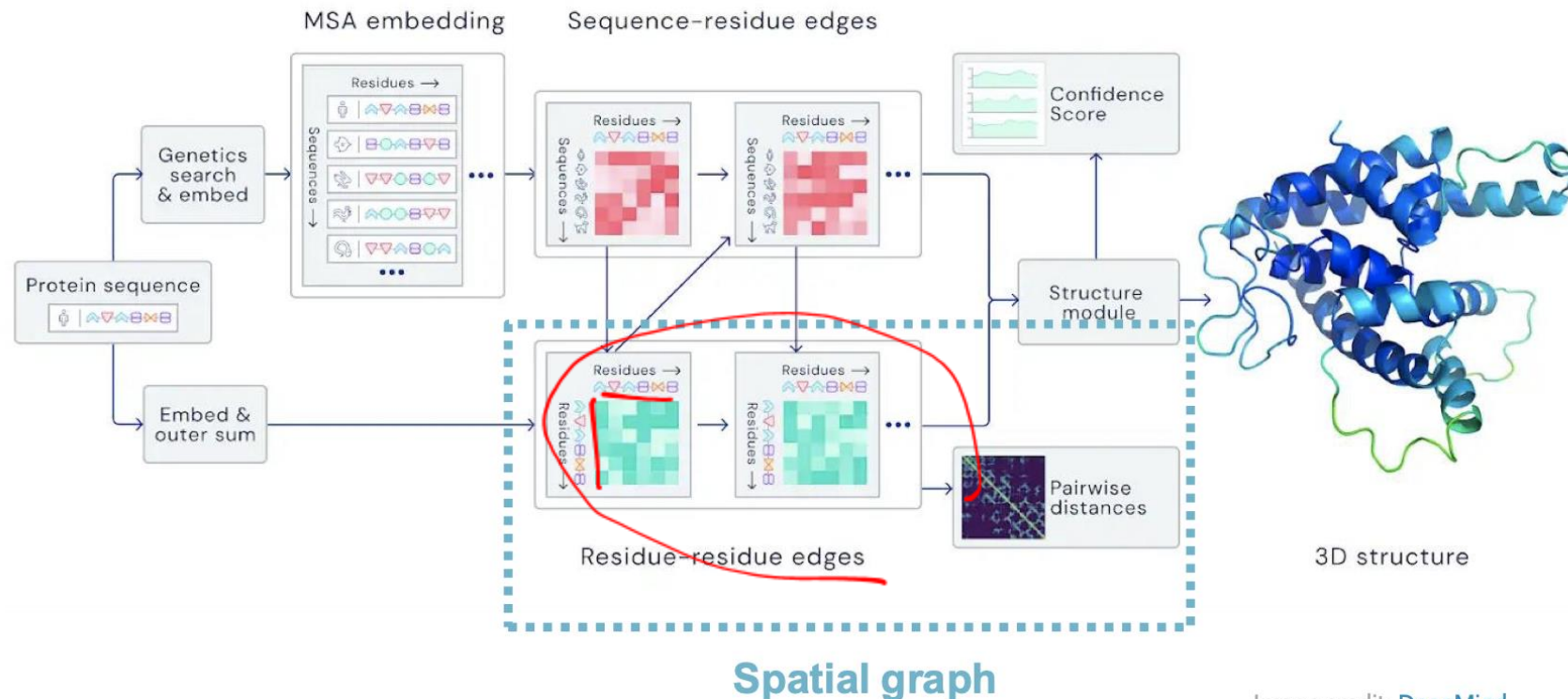
By Claire Moses, Cade Metz and Teddy Rosenbluth



PRINT EDITION 3 Scientists Used A.I. To Crack Proteins' Code | October 10, 2024, Page A11

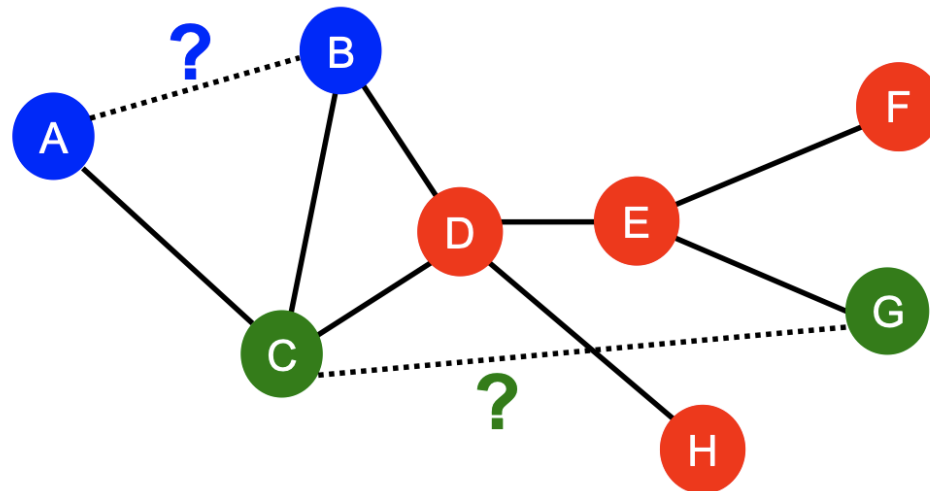
AlphaFold: Impact

- Key idea: “Spatial graph”
 - Nodes: Amino acids in a protein sequence
 - Edges: Proximity between amino acids (residues)



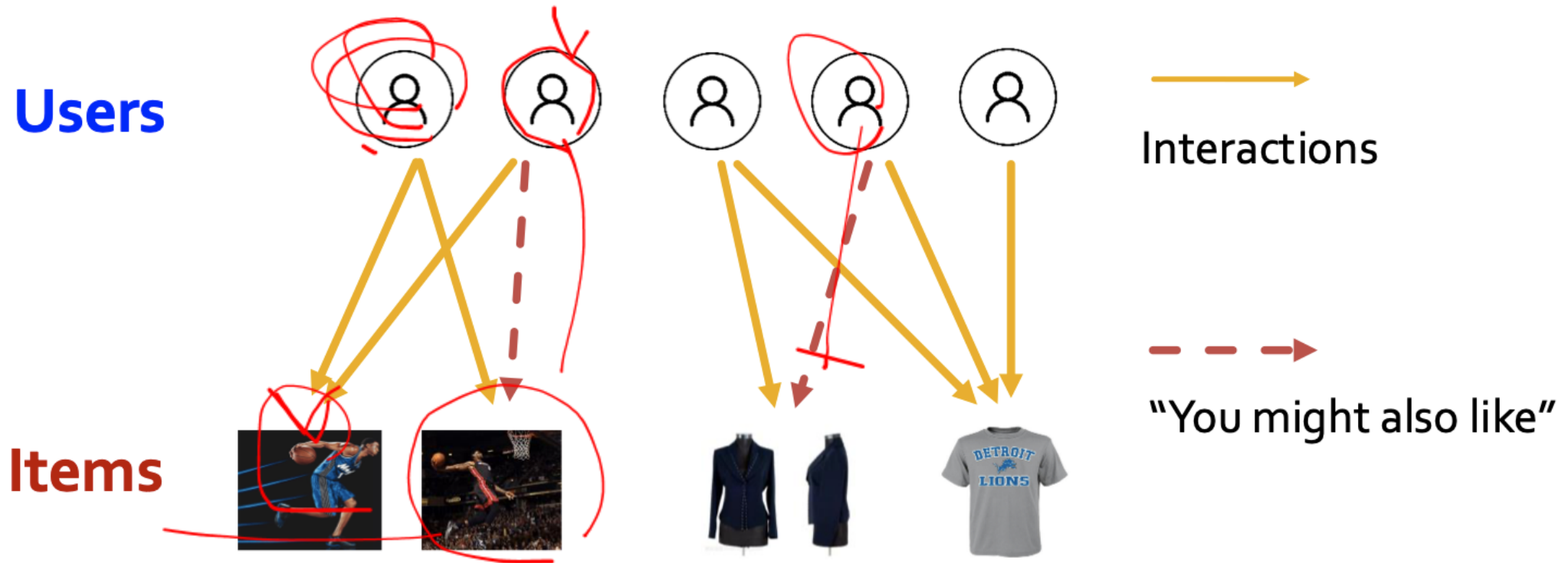
Link-Level Prediction Task

- The task is to predict new/missing/unknown links based on the existing links.
 - At test time, node pairs (with no existing links) are ranked, and top K node pairs are predicted.
- Task: Make a prediction for a pair of nodes.



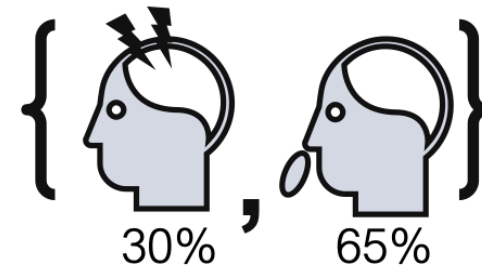
Recommender Systems

- Users interacts with items

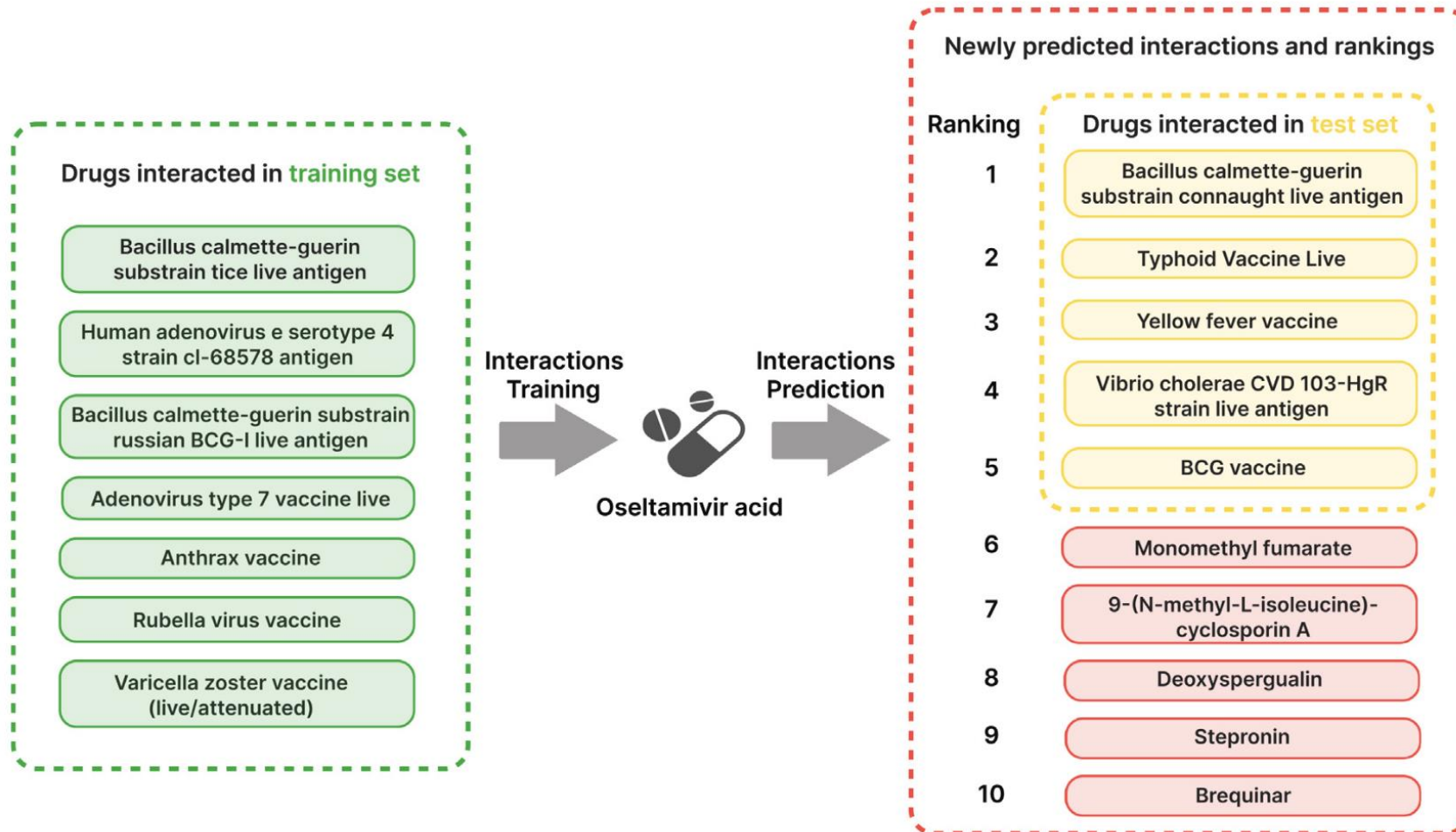


Drug-Drug Interactions

- Many patients take multiple drugs to treat complex or co-existing diseases:
 - 46% of people ages 70-79 take more than 5 drugs
 - Many patients take more than 20 drugs to treat heart disease, depression, insomnia, etc.
- Task: Given a pair of drugs predict adverse side effects



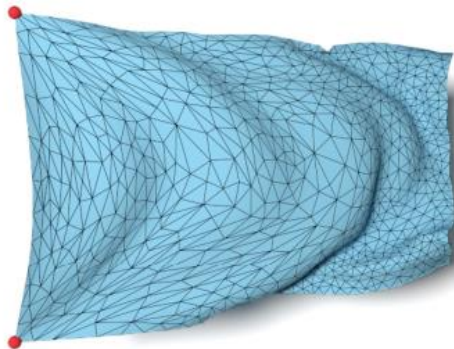
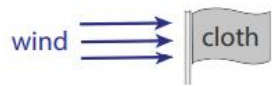
Drug-Drug Interactions – Contd.



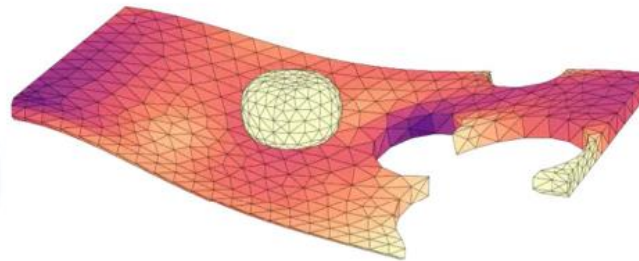
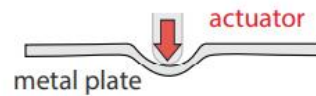
Mesh Graph-based Simulation

- Learned simulation model with very desirable properties:
 - Strong generalization
 - Graph models which scale
 - **Speed : 10-100x faster than ground truth simulator**

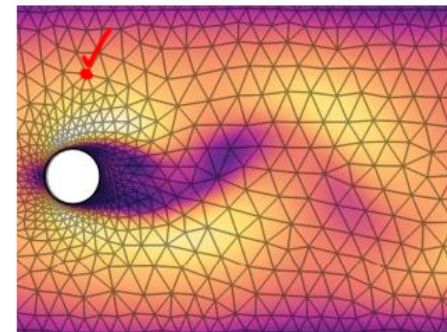
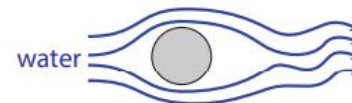
(a) FlagDynamic



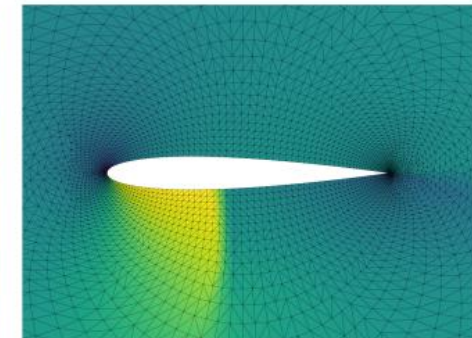
(b) DeformingPlate



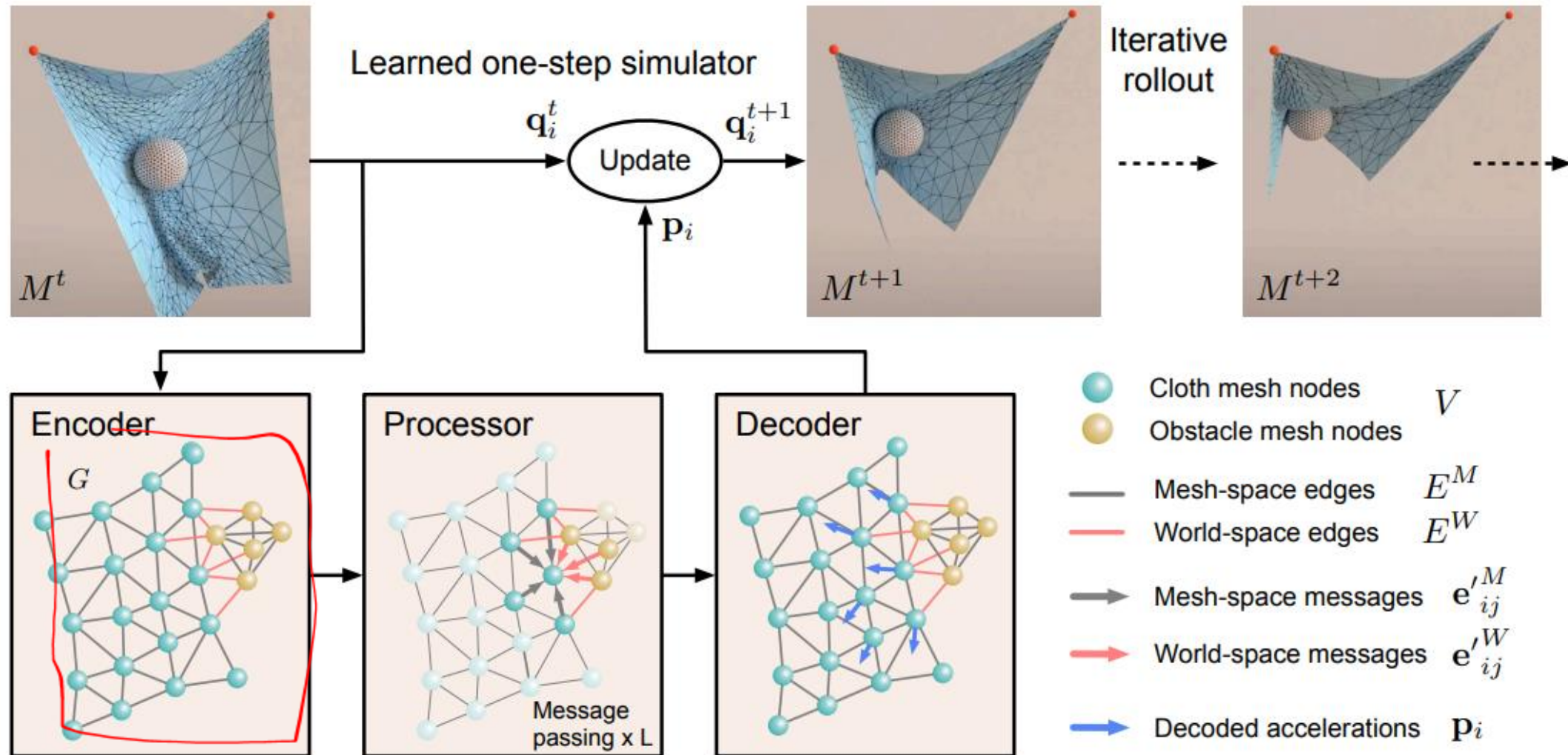
(c) CylinderFlow



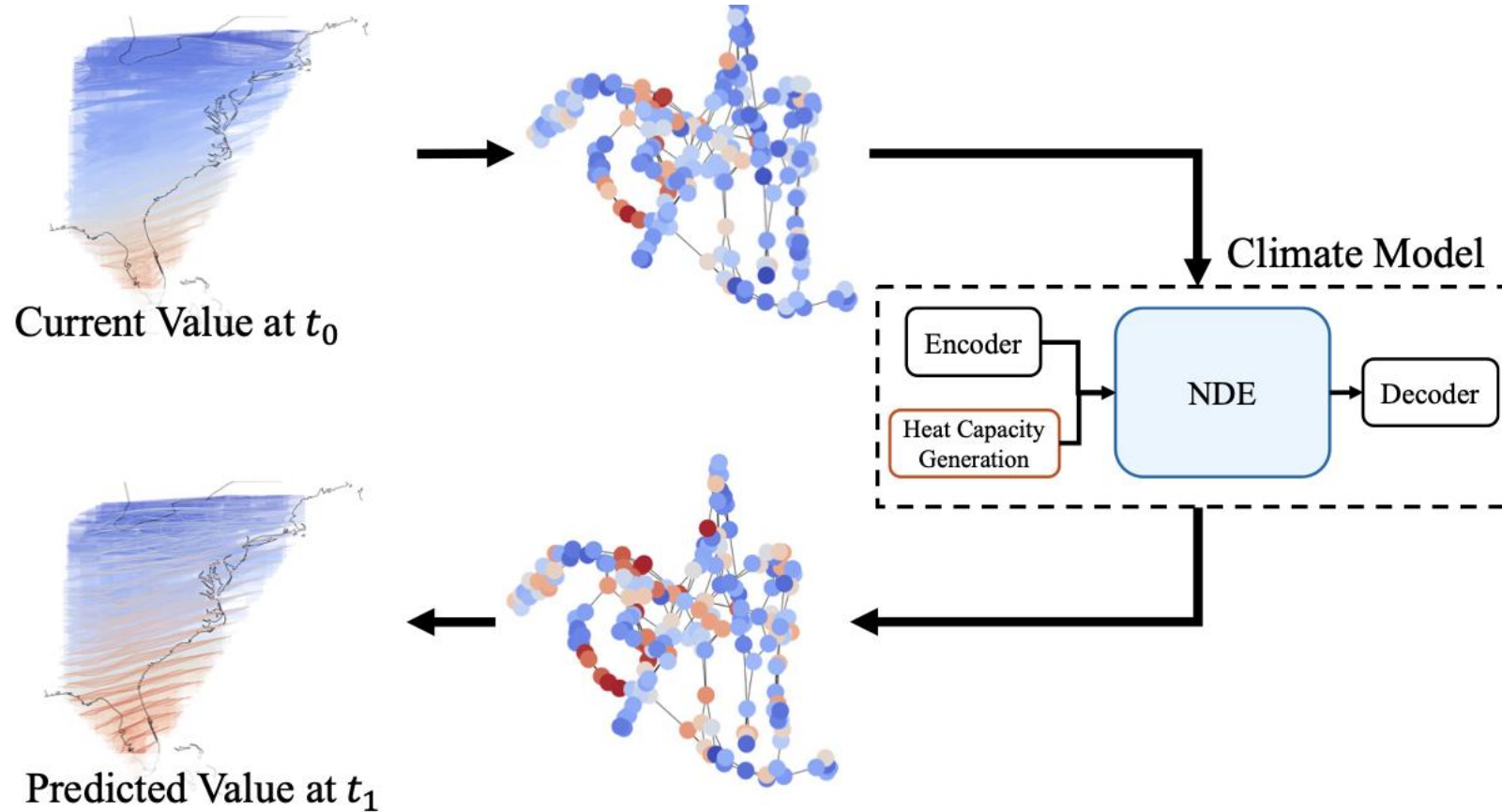
(d) Airfoil



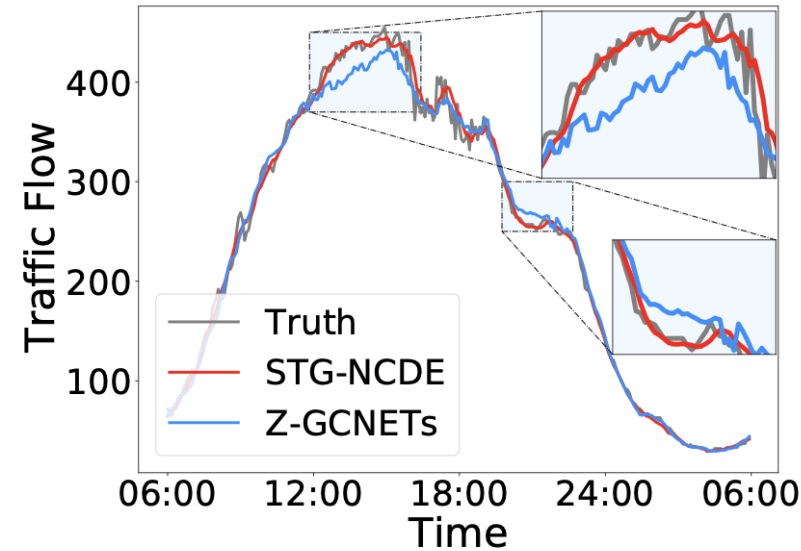
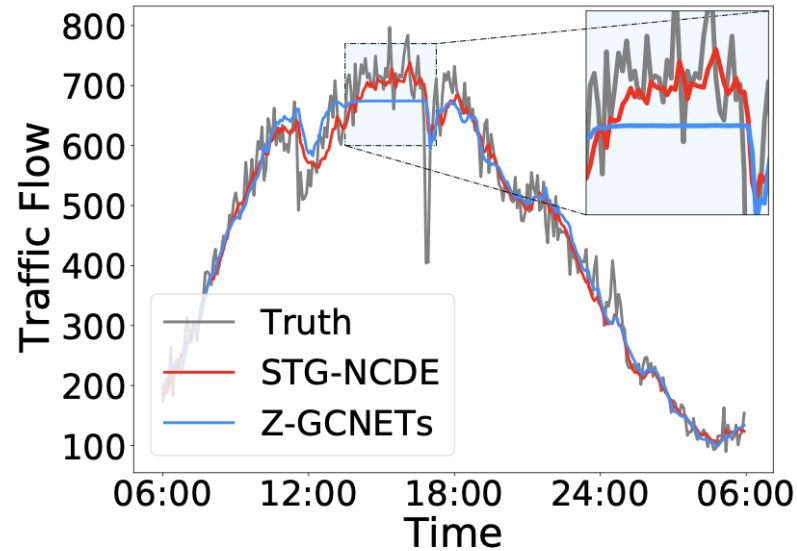
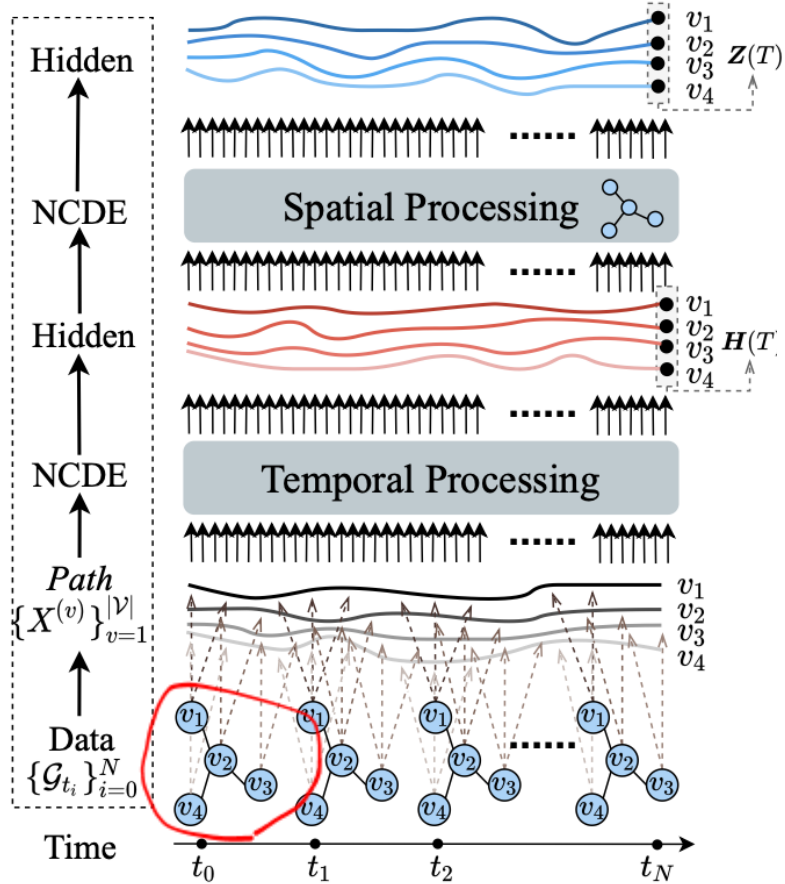
Mesh Graph-based Simulation – Contd.



Climate Modeling



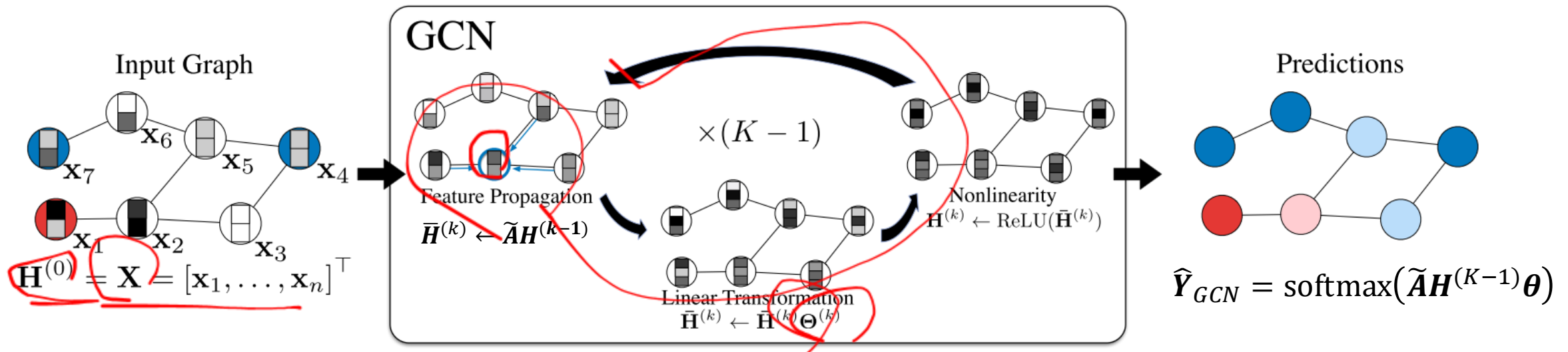
Traffic Forecasting



Graph Neural Networks

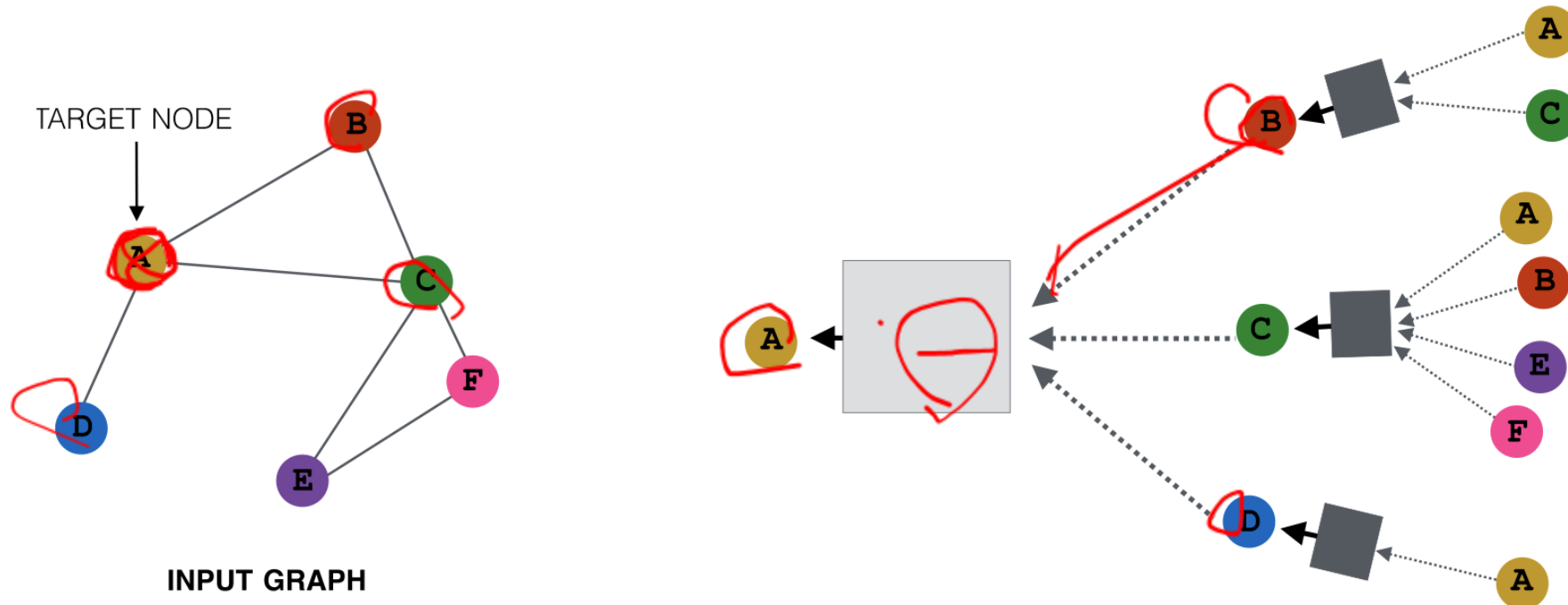
Graph Convolutional Networks (GCNs)

- Given a graph G with adjacency matrix $\mathbf{A}$ and degree matrix $\mathbf{D}$, we define the normalized adjacency matrix $\tilde{\mathbf{A}}$ as $\tilde{\mathbf{A}} = \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}$.
- Learn how to propagate information across the graph to compute node features



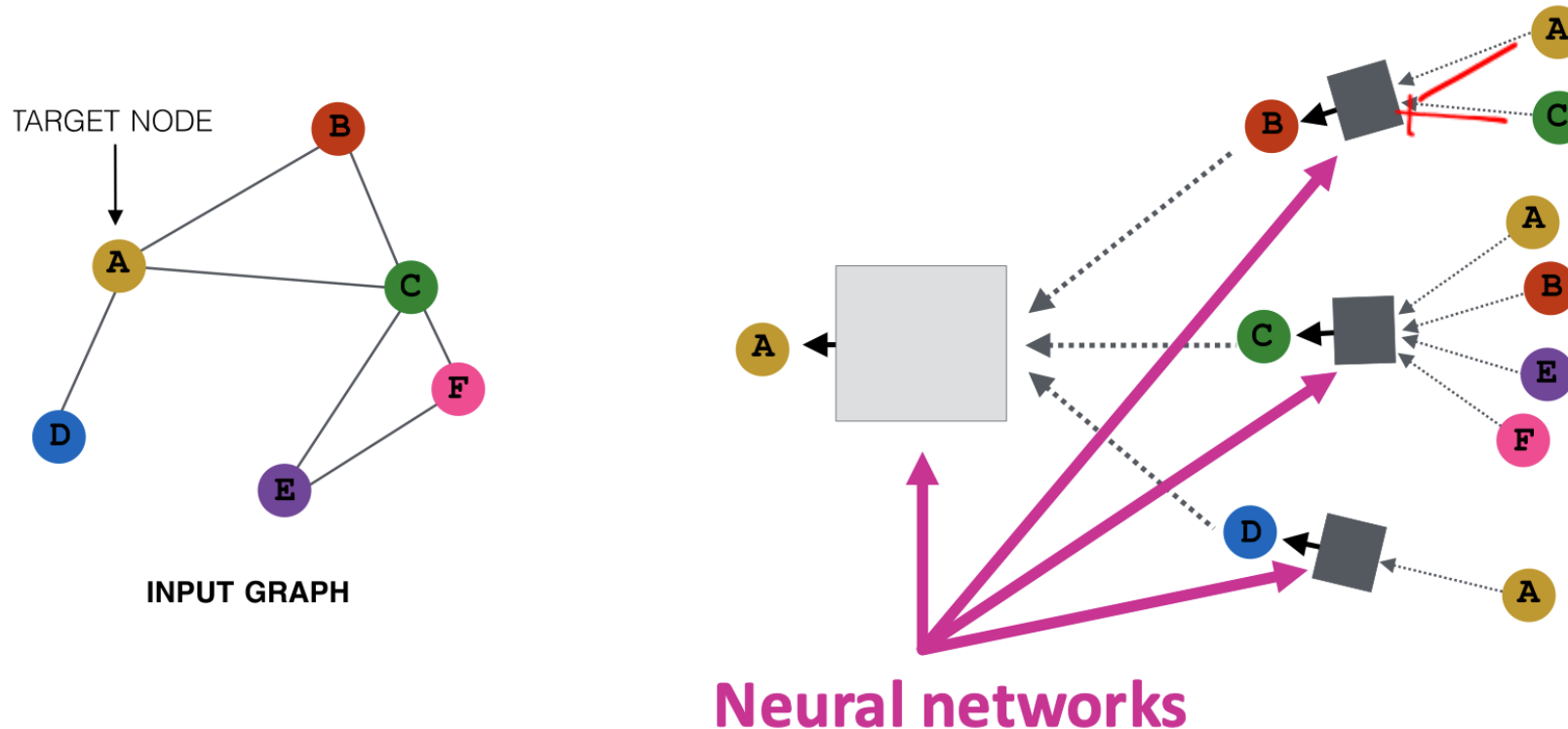
Idea: Aggregate Neighbors

- **Key Idea:** Create node embeddings based on **local network neighborhoods**



Idea: Aggregate Neighbors – cont.

- **Intuition:** Nodes aggregate information from their neighbors using neural networks



The Math: Deep Encoder of a GCN

- Basic approach: Average neighbor messages and apply a neural network

The diagram illustrates the mathematical formulation of the GCN deep encoder. It features the equation
$$\mathbf{h}_v^k = \sigma \left(\mathbf{W}_k \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|} + \mathbf{B}_k \mathbf{h}_v^{k-1} \right), \quad \forall k > 0$$
 with several annotations:

- A purple box labeled $\mathbf{h}_v^0 = \mathbf{x}_v$ is circled in red, with a blue arrow pointing to it from the text "Initial 'layer 0' embeddings are equal to node features".
- A green box labeled $\mathbf{h}_v^k$ is circled in red, with a green arrow pointing to it from the text "kth layer embedding of v ".
- A blue box labeled σ is circled in red, with a blue arrow pointing to it from the text "non-linearity (e.g., ReLU or tanh)".
- A red circle around $\mathbf{W}_k$ has a red arrow pointing to it from the $\mathbf{h}_v^0$ box.
- A red circle around the summation term $\sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|}$ has a red arrow pointing to it from the $\mathbf{h}_v^0$ box.
- A red circle around $\mathbf{h}_u^{k-1}$ has a red arrow pointing to it from the $\mathbf{h}_v^0$ box.
- An orange box labeled $\mathbf{h}_v^{k-1}$ is circled in red, with an orange arrow pointing to it from the text "previous layer embedding of v ".
- A red circle around $\mathbf{B}_k \mathbf{h}_v^{k-1}$ has a red arrow pointing to it from the text "average of neighbor's previous layer embeddings".

Matrix Formulation

Basic Neighborhood Aggregation

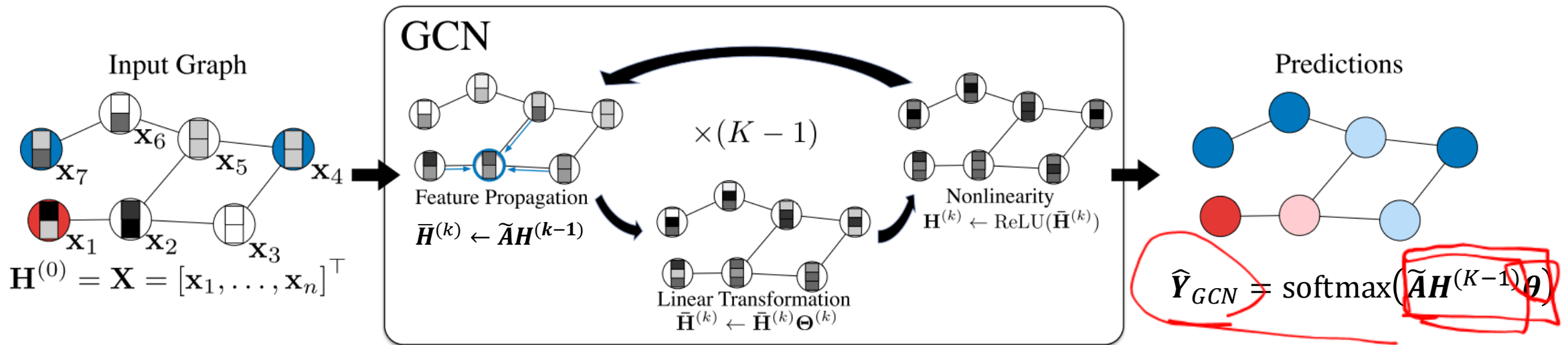
$$\mathbf{h}_v^k = \sigma \left(\mathbf{w}_k \sum_{u \in \mathcal{N}(v)} \frac{\mathbf{h}_u^{k-1}}{|\mathcal{N}(v)|} + \mathbf{B}_k \mathbf{h}_v^{k-1} \right)$$

- Many aggregations can be performed efficiently by matrix operations.
- Let $\mathbf{H}^{(k)} = [\mathbf{h}_1^{(k)} \dots \mathbf{h}_{|\mathcal{V}|}^{(k)}]^T$
- Then $\sum_{u \in \mathcal{N}(v)} \mathbf{h}_u^{(k)} = \mathbf{A}_{v,:} \mathbf{H}^{(k)}$
- Let $\mathbf{D}$ be diagonal matrix where $\mathbf{D}_{v,v} = \text{Deg}(v) = |\mathcal{N}(v)|$
- Therefore,

$$\sum_{u \in \mathcal{N}(v)} \frac{\mathbf{h}_u^{(k-1)}}{|\mathcal{N}(v)|} \rightarrow \mathbf{H}^{(k+1)} = \mathbf{D}^{-1} \mathbf{A} \mathbf{H}^{(k)}$$

Graph Convolutional Networks (GCNs)

- Given a graph G with adjacency matrix $\mathbf{A}$ and degree matrix $\mathbf{D}$, we define the normalized adjacency matrix $\tilde{\mathbf{A}}$ as $\tilde{\mathbf{A}} = \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}$.



Graph Convolutional Networks (GCNs) – cont.

Basic Neighborhood Aggregation

$$\mathbf{h}_v^k = \sigma \left(\mathbf{W}_k \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|} + \mathbf{B}_k \mathbf{h}_v^{k-1} \right)$$

VS.

GCN Neighborhood Aggregation

$$\mathbf{h}_v^k = \sigma \left(\mathbf{W}_k \sum_{u \in N(v) \cup v} \frac{\mathbf{h}_u^{k-1}}{\sqrt{|N(u)| |N(v)|}} \right)$$

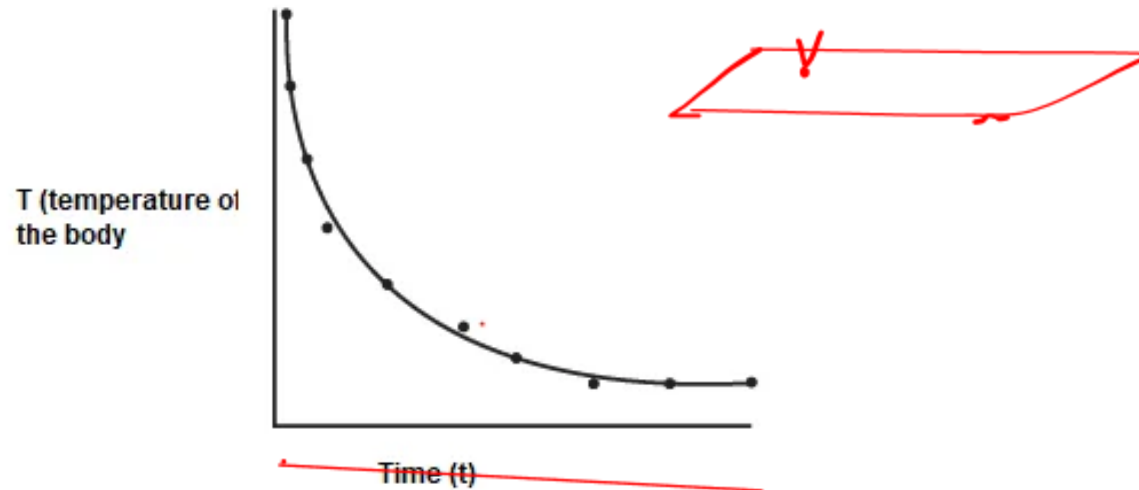
same matrix for self and
neighbor embeddings

per-neighbor normalization

Heat Diffusion Process and GNNs

Newton Law of Cooling

- “The temperature a hot body loses in a given time is proportional to the temperature difference between the object and the environment.”



(824)

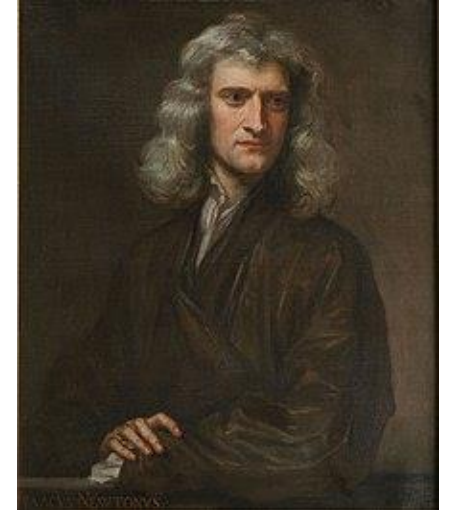
with a little pressing, I took a drop thereof, and in it discover'd a mighty number of living Creatures. I repeated my observation the same evening with the same success, but the next day I could find none of them alive; and whereas I had laid that drop upon a small Copper Plate, I fancied to my self that the exhalation of the moisture might be the cause of their death, and not the cold weather, which at that time was very moderate.

In the beginning of April I took the Male seed of a Jack or Pike, but could discover nothing more than in that of a Cod-fish, but having added about four times as much Water in quantity as the matter itself was, and then making my remarks, I could perceive that the *Animalcula* did not only wax stronger and swifter, but, to my great amazement, I saw them move with that celerity, that I could compare it to nothing more than what we have seen with our naked Eye, a River Fish chafed by its powerful Enemy, which is just ready to devour it: You must observe that this whole Course was not longer than the Diameter of a single Hair of ones Head.

VII. Scala graduum Caloris.

Calorum Descriptiones & signa.

0	Calor aeris hybernici ubi aqua incipit gelu rigescere. Innotescit hic calor accurate locando Thermometrum in nive compressa quo tempore gelu solvitur.
0,1,2	Calores aeris hybernici.
2,3,4	Calores aeris verni & autumnalis.
4,5,6	Calores aeris æstivi.
6	Calor aeris meridiani circa mensium Julium.
12	Calor maximus quem Thermometrum ad contactum

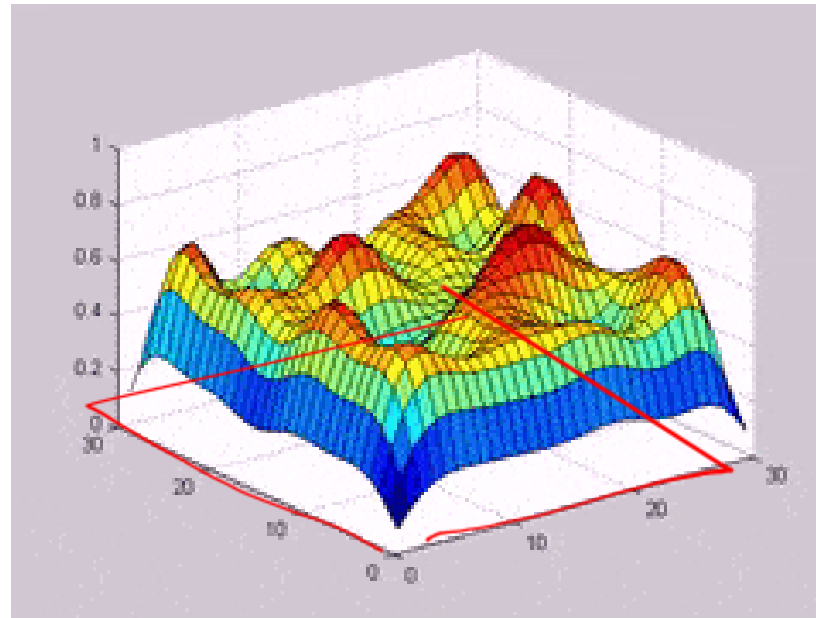


Issac Newton

Diffusion takes many forms

- Heat diffusion:

$$\frac{d\mathbf{H}(t)}{dt} = -\Delta\mathbf{H}(t)$$



Heat equation, wikipedia



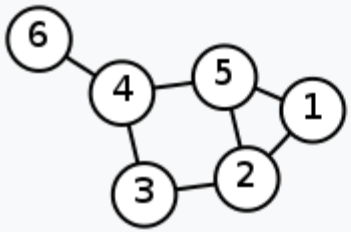
Joseph Fourier

Multiplication with the Laplacian

- Consider a graph G with Laplacian L and a graph signal $\mathbf{x}$ on G .
- Signal $\mathbf{y} = L\mathbf{x}$ results from multiplying $\mathbf{x}$ with the Laplacian.
- Component y_i of $\mathbf{y}$ is as follows:

$$y_i = \sum_{j \in \mathcal{N}_i} (\mathbf{x}_i - \mathbf{x}_j).$$

- y_i measures the mean difference between $\mathbf{x}$ at a node and its neighborhood, i.e., difference operator.

Labelled graph	Degree matrix	Adjacency matrix	Laplacian matrix
	$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$

Heat diffusion over graph

- Heat Diffusion Equation:

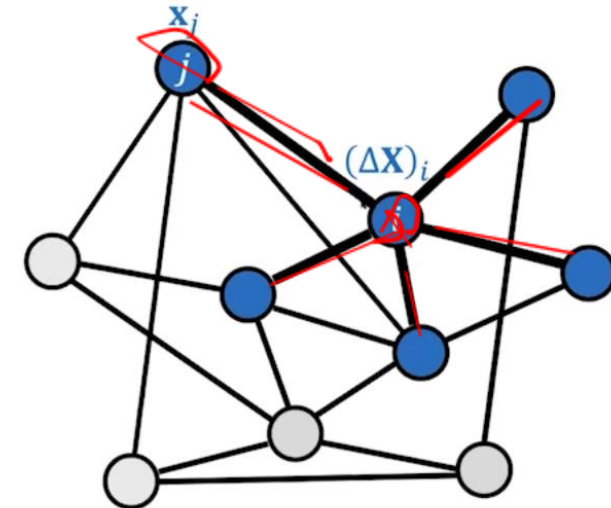
$$\frac{\partial \mathbf{x}}{\partial t} = \text{div}(\nabla \mathbf{x}).$$

- Laplacian:

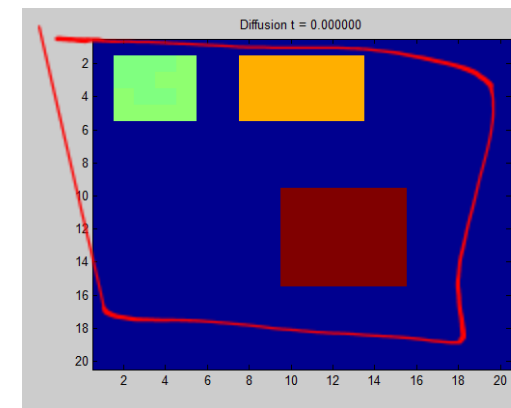
$$(\Delta \mathbf{X})_i = \text{div}(\nabla \mathbf{x})_i = \sum_j a_{ij} (\mathbf{x}_j - \mathbf{x}_i).$$

- Diffusion equations on graphs with an explicit Euler discretization:

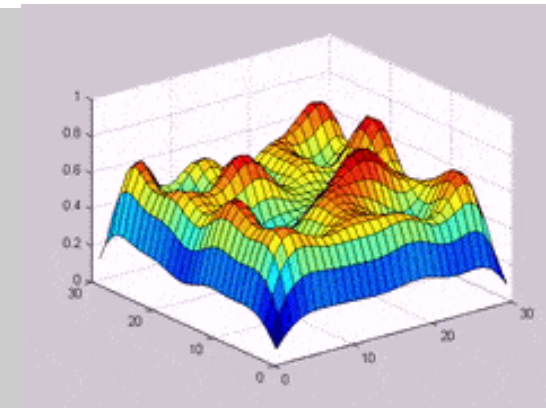
$$\mathbf{X}(t + h) = \mathbf{X}(t) + h(-\mathbf{L}\mathbf{X}(t)).$$



Diffusion on a graph



Laplacian matrix, wikipedia



Heat equation, Wikipedia

Heat diffusion over graph – contd.

- Via $\frac{d\mathbf{x}}{dt} = -\mathbf{L}\mathbf{x}$, we say that the signal diffuses through the graph, i.e., heat equation.
 - $\frac{d\mathbf{x}}{dt}$ stands for the time-derivative of $\mathbf{x}$.
 - The Euler method updates the temperature after $\mathbf{x}(t+h) = \mathbf{x}(t) - h\mathbf{L}\mathbf{x}(t)$.
- Temperature at each location is averaged with its neighbors' temperatures.
 - $-\mathbf{L}\mathbf{x}(t)$ will be negative (resp. positive) if my temperature is higher (resp. smaller) than those of neighbors.

Graph convolutional networks [Kipf & Welling, 2017]

- Let us use the normalized Laplacian $\tilde{\mathbf{L}}$ and $h = 1$:

$$\mathbf{x}(t+1) = \mathbf{x}(t) - \tilde{\mathbf{L}}\mathbf{x}(t) = (\mathbf{I} - \tilde{\mathbf{L}})\mathbf{x}(t).$$

- Therefore, GCN by Kipf and Welling uses the following **diffusion process** augmented with a trainable parameter (or **diffusivity**) $\mathbf{W}$:

$$\mathbf{x}(t+1) = \sigma((\mathbf{I} - \tilde{\mathbf{L}})\mathbf{x}(t)\mathbf{W}), \text{ where } \sigma \text{ is a non-linear activation.}$$

- At the same time, one can consider that this is a **simple graph filtering** approach.
 - Given $\mathbf{H} = \sum_{\ell=0}^L h_{\ell} \mathbf{S}^{\ell}$, $h_0 = 0$, $L = 1$ and $\mathbf{S} = \mathbf{I} - \tilde{\mathbf{L}} = \tilde{\mathbf{A}}$.
 - In fact, GCN uses the first two Chebyshev polynomial terms.

Equivalence between LinearGCN and Diffusion Process

- Key insight from a continuous perspective:
 - LinearGCN's propagation = a discretization of the diffusion equation

- Graph Diffusion Equation

$$\frac{d\mathbf{X}(t)}{dt} = -\tilde{\mathbf{L}}\mathbf{X}(t),$$

where $\tilde{\mathbf{L}} = \mathbf{I} - \tilde{\mathbf{A}}$ is the normalized Laplacian matrix, and $\mathbf{X}(0)$ is an input $\mathbf{X}$.

- Discretization

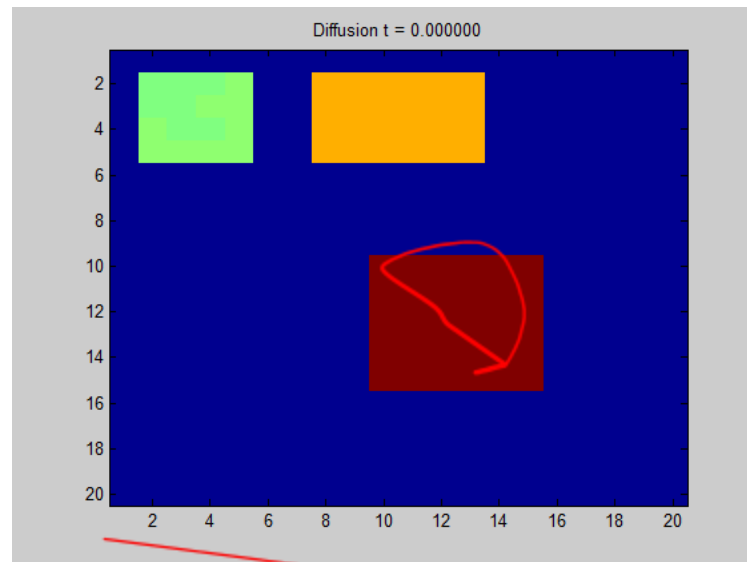
- Applying the forward Euler method with time interval Δt :

$$\begin{aligned}\mathbf{X}(t + \Delta t) &= \mathbf{X}(t) - \Delta t \tilde{\mathbf{L}} \mathbf{X}(t) \\ &= \mathbf{X}(t) - \Delta t (\mathbf{I} - \tilde{\mathbf{A}}) \mathbf{X}(t) \\ &= [(1 - \Delta t) \mathbf{I} + \Delta t \tilde{\mathbf{A}}] \mathbf{X}(t)\end{aligned}$$

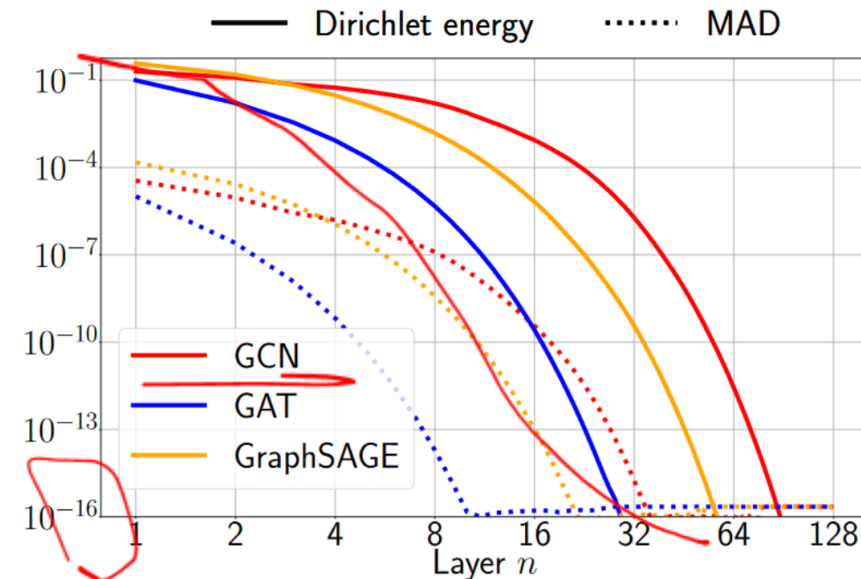
- Thus, LinearGCN is the Euler discretization of Diffusion Eq. with step size $\Delta t = 1$.

Challenges with the Diffuse Nature of GNNs

- One of major drawbacks of GCN is *oversmoothing*.



Laplacian diffusion matrix (Wikipedia)



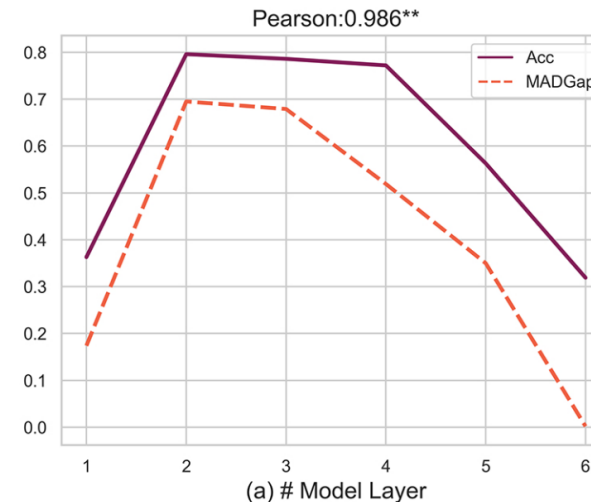
Dirichlet energy and MAD of GNNs on CORA [Rusch et al. 2023]

Oversmoothing problem

- Oversmoothing problem
 - All nodes' last hidden vectors become similar to each other when the number of GCN layers is large [Chen et al., 2020a].



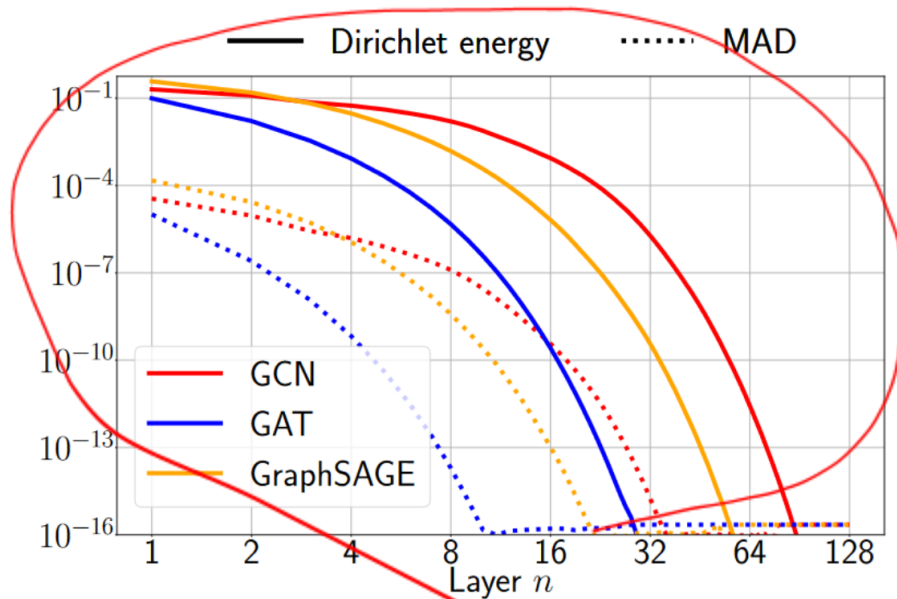
Mean Avg. Distance (MAD)



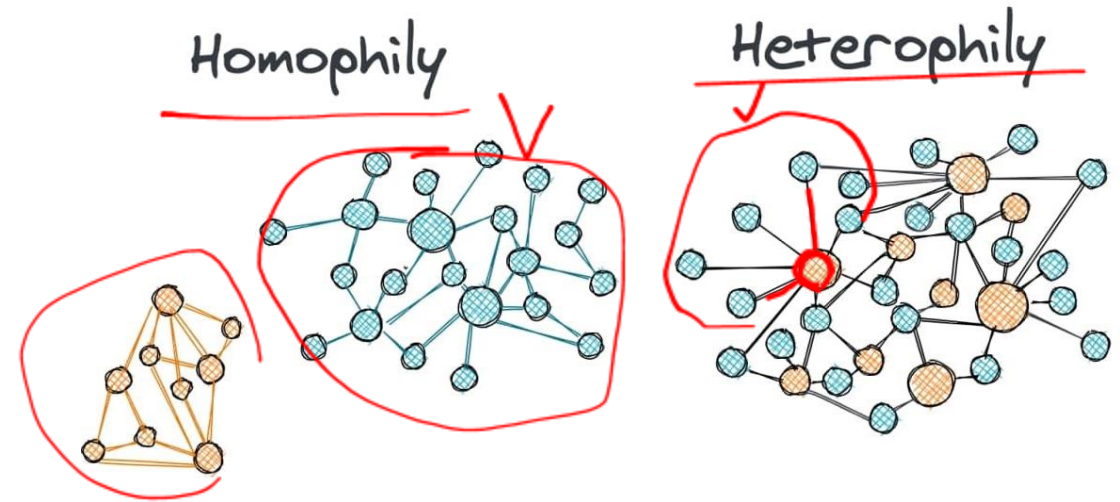
Test accuracy of GCN on CORA

Challenges with the Diffuse Nature of GNNs

- Two major drawbacks of these approaches are oversmoothing and heterophily nodes.



Dirichlet energy and MAD of GNNs on CORA [Rusch et al. 2023]



Homophily and Heterophily Graphs

(<https://graphml.substack.com/p/gml-newsletter-homophily-heterophily>)

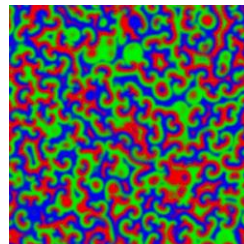
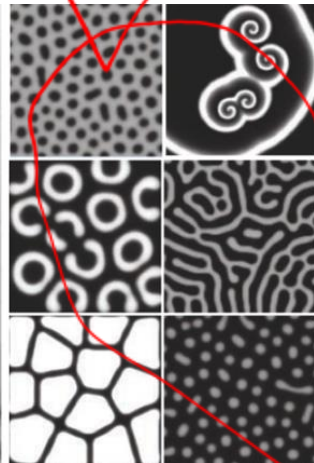
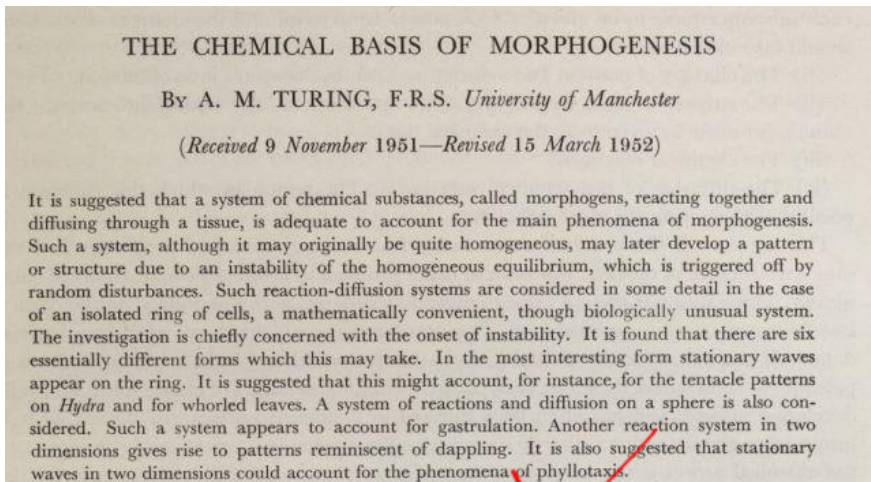
Oversmoothing problem – contd.

- The history of GCNs is basically the history of battling with the oversmoothing problem.
 - GCNII [Chen et al., 2020b] tries to overcome the problem by i) initial residual connection, and ii) identity mapping.
$$\mathbf{x}(t+1) = \sigma\left((\alpha(\mathbf{I} - \tilde{\mathbf{L}})\mathbf{x}(t) + (1 - \alpha)\mathbf{x}(0))(\beta\mathbf{I} + (1 - \beta)\mathbf{W})\right)$$
 - A series of graph sparsification methods have been proposed in [Rong et al., 2019; Hasanzadeh et al., 2020; Zheng et al., 2020].
- Our answer to this problem is to use reaction-diffusion equations.

Overcoming the over-smoothing problem

<Choi et al., GREAD: Graph Reaction-Diffusion Neural Networks, ICML 2023>

Turing Pattern and Reaction-Diffusion Equation



- The reaction-diffusion equations can be written as:

$$\frac{\partial \mathbf{x}}{\partial t} = \text{div}(\nabla \mathbf{x}) + r(\mathbf{x}).$$

- Reaction-Diffusion equations on graphs with an explicit discretization:
$$\mathbf{X}(t + s) = \mathbf{X}(t) + s \left(-\tilde{\mathbf{L}}\mathbf{X}(t) + r(\mathbf{X}(t)) \right).$$

Diffusion Equation-based GNNs

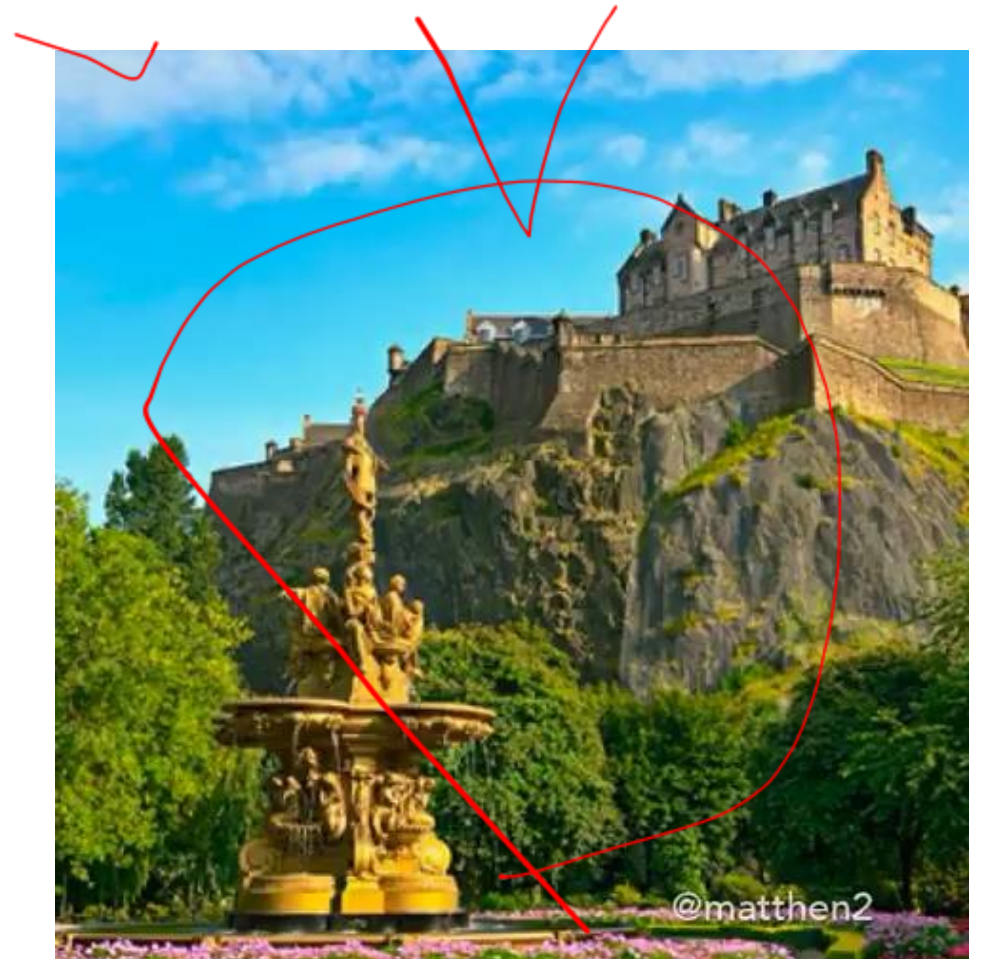
- Many papers proposed similar approaches based on the diffusion equation-based interpretation of Graph Neural Networks (GNNs), e.g., GRAND [Chamberlain et al., 2021].

Table 1. A comparison table of existing methods. ‘ $\triangle$ ’ means that it corresponds to a specific type of reaction only.

Model	Diffusion	Reaction	Continuous-time
FA-GCN	O	$\triangle$	X
GPR-GNN	O	$\triangle$	X
ACM-GCN	O	$\triangle$	X
CGNN	O	X	O
GRAND	O	X	O
BLEND	O	X	O
GREAD	O	O	O

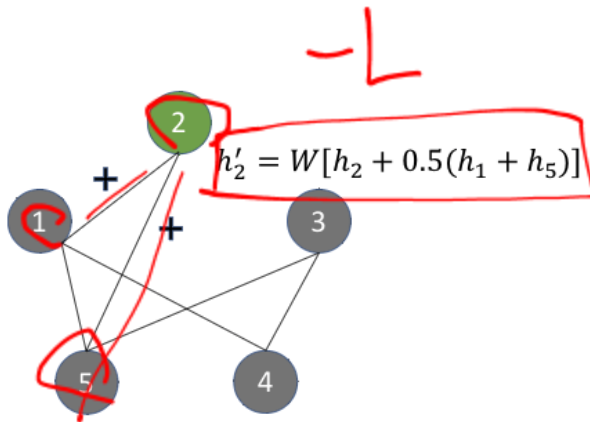
How to Design the Reaction Term?

- Iterating through the low-pass and high-pass filters is the reaction-diffusion equation.
- In computer vision, it is known that an alternating sequence of the blurring (e.g., $x_t = -\tilde{L}x$) and sharpening (e.g., $x_t = \tilde{L}x$) operations also creates Turing patterns.

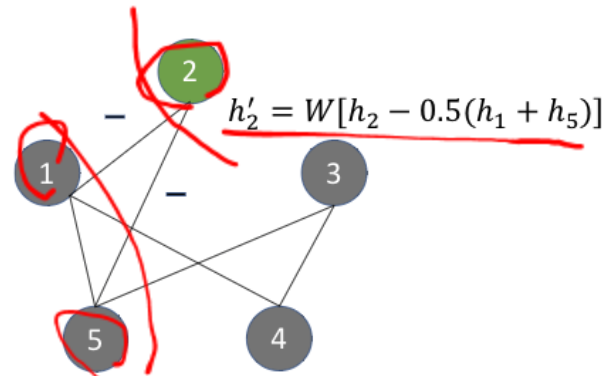


How to Design the Reaction Term?

- Iterating through the low-pass and high-pass filters is the reaction-diffusion equation.



A low-pass filter



A high-pass filter



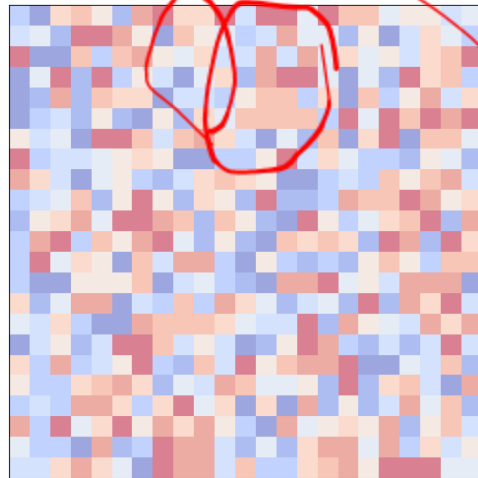
Reaction-diffusion equations – contd.

- In computer vision, it is known that an alternating sequence of the **blurring** (e.g., $x_t = -\tilde{L}x$) and **sharpening** (e.g., $x_t = \tilde{L}x$) operations also creates **Turing patterns**.
 - <https://twitter.com/i/status/1327449422532263936>

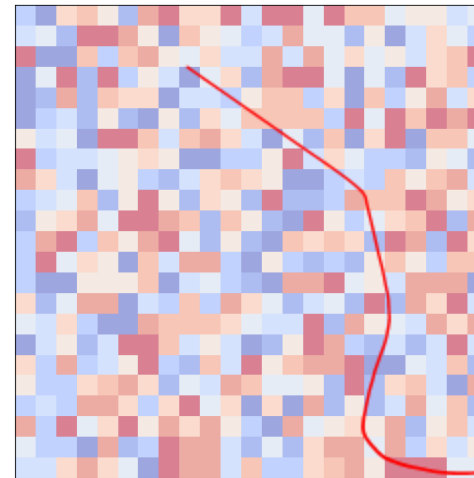


Reaction-Diffusion Equations for Graphs

- The following visualization (in our ICML paper) also delivers the intuition of GREAD's successful node classification.
 - Assume a 2D grid network with 1D node signal (red is high signal, blue is low signal).



<GREAD-BS>



<Diffusion Only>

Graph neural reaction-diffusion networks

- Given a graph signal $\mathbf{X} \in \mathbb{R}^{N \times D}$, GREAD consists of the following three parts:
 - Initial embedding layer: $\mathbf{H}(0) = \mathbf{e}(\mathbf{X})$,
 - Reaction-diffusion layer: $\mathbf{H}(T) = \mathbf{H}(0) + \int_0^T [-\alpha \tilde{\mathbf{L}} \mathbf{H}(t) + \beta \mathbf{r}(\mathbf{H}(t))] dt$,
 - Output layer: $\hat{\mathbf{y}} = \mathbf{o}(\mathbf{H}(T))$.
- We also proposed to learn a normalized adjacency matrix $\bar{\mathbf{A}}$ as in [Li et al., 2018], and its Laplacian counterpart is $\bar{\mathbf{L}} = (\mathbf{I} - \bar{\mathbf{A}})$.
 - We use the self-attention method to learn a graph from data.

Blurring-sharpening equations

- Given a hidden signal $\mathbf{H}(t) \in \mathbb{R}^{N \times D}$ at time (or layer) t , we apply the blurring operation:

$$\mathbf{B}(t) = \mathbf{H}(t) - \bar{\mathbf{L}}\mathbf{H}(t) = \mathbf{H}(t) + (\bar{\mathbf{A}} - \mathbf{I})\mathbf{H}(t) = \bar{\mathbf{A}}\mathbf{H}(t).$$

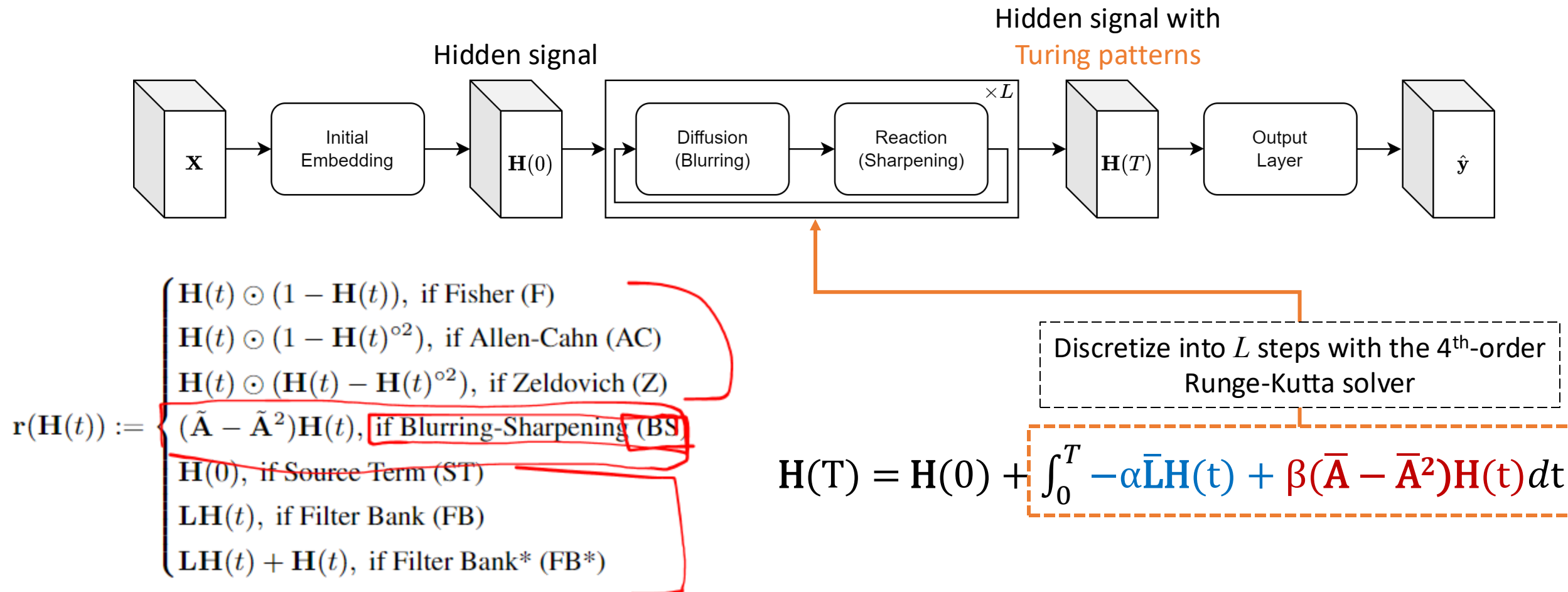
- We then apply the following sharpening operation:

$$\mathbf{H}(t+1) = \mathbf{B}(t) + \bar{\mathbf{L}}\mathbf{B}(t) = \mathbf{H}(t) - \bar{\mathbf{L}}\mathbf{H}(t) + (\bar{\mathbf{A}} - \bar{\mathbf{A}}^2)\mathbf{H}(t).$$

- Therefore, our main proposed model, GREAD-BS, is as follows:

$$\mathbf{H}(T) = \mathbf{H}(0) + \int_0^T -\alpha \bar{\mathbf{L}}\mathbf{H}(t) + \beta (\bar{\mathbf{A}} - \bar{\mathbf{A}}^2)\mathbf{H}(t) dt.$$

Architecture of GREAD-BS



Experimental Results

Table 2. The average ranking/accuracy and the Olympic ranking of some selected high-performing models on 9 real-world datasets. ‘*’ (resp. ‘†’) indicates that an improvement over GloGNN (resp. ACM-GCN) is statistically significant ($p < 0.05$) under the Wilcoxon signed-rank test.

Method	Average		Olympic Ranking		
	Ranking	Accuracy	Gold	Silver	Bronze
GREAD-BS	1.56	76.64 ^{*†}	5	4	0
GREAD-FB*	6.72	74.51 [†]	0	0	3
GREAD-F	7.50	74.13	1	1	1
GloGNN	8.17	74.99	0	0	1
GREAD-AC	8.50	73.71	1	0	0
ACM-GCN	8.67	74.92	0	1	0
GGCN	9.50	75.05	0	1	0
Sheaf	10.33	75.06	1	1	0

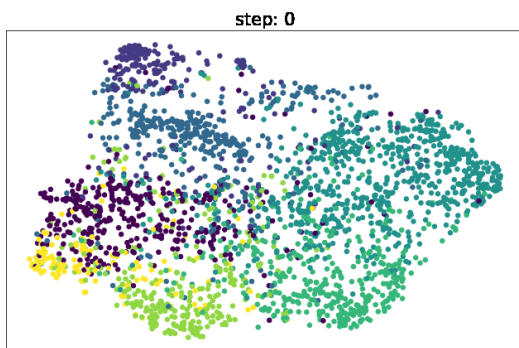
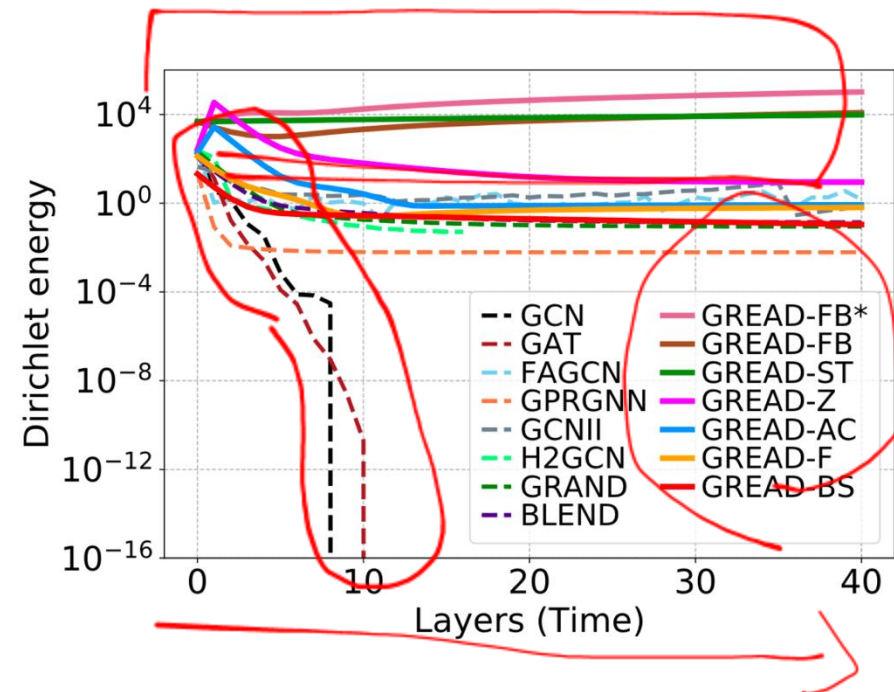
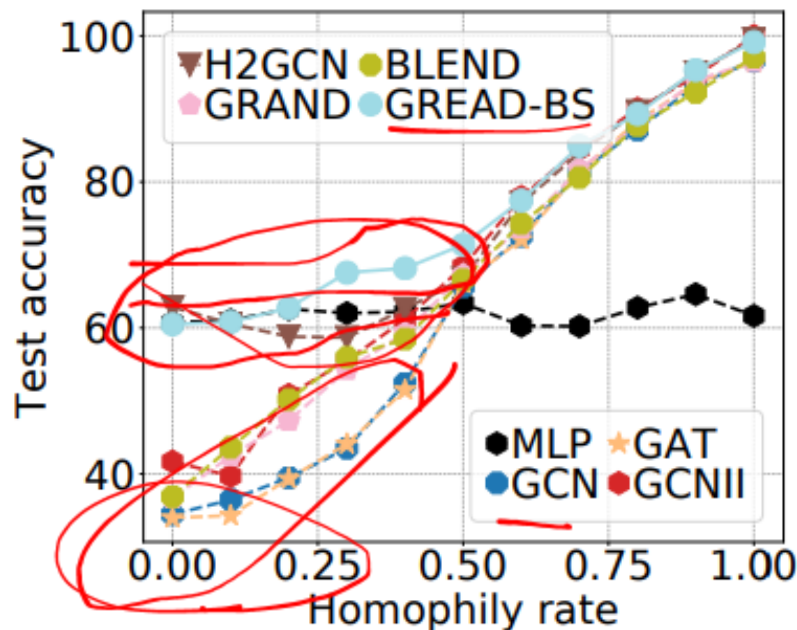


Table 4. Results on real-world datasets: mean $\pm$ std. dev. accuracy for 10 different data splits. We show the best three methods in red (first), blue (second), and purple (third). Other missing 16 baselines are in Appendix B.

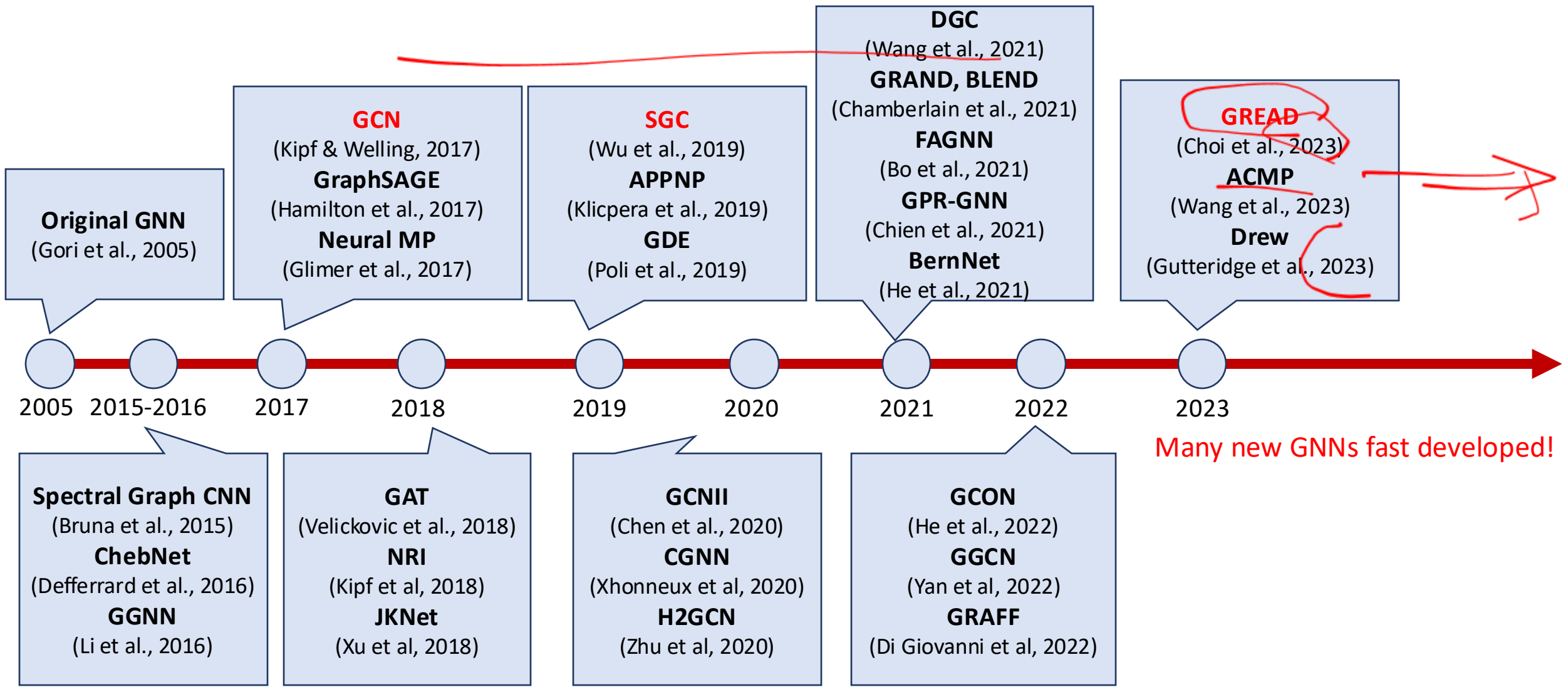
Dataset	Texas	Wisconsin	Cornell	Film	Squirrel	Chameleon	Cora	Citeseer	PubMed	Avg.
Geom-GCN	66.76 $\pm$ 2.72	64.51 $\pm$ 3.66	60.54 $\pm$ 3.67	31.59 $\pm$ 1.15	38.15 $\pm$ 0.92	60.00 $\pm$ 2.81	85.35 $\pm$ 1.57	78.02 $\pm$ 1.15	89.95 $\pm$ 0.47	63.87
H2GCN	84.86 $\pm$ 7.23	87.65 $\pm$ 4.98	82.70 $\pm$ 5.28	35.70 $\pm$ 1.00	36.48 $\pm$ 1.86	60.11 $\pm$ 2.15	87.87 $\pm$ 1.20	77.11 $\pm$ 1.57	89.49 $\pm$ 0.38	71.33
GGCN	84.86 $\pm$ 4.55	86.86 $\pm$ 3.29	85.68 $\pm$ 6.63	37.54 $\pm$ 1.56	55.17 $\pm$ 1.58	71.14 $\pm$ 1.84	87.95 $\pm$ 1.05	77.14 $\pm$ 1.45	89.15 $\pm$ 0.37	75.05
LINKX	74.60 $\pm$ 8.37	75.49 $\pm$ 5.72	77.84 $\pm$ 5.81	36.10 $\pm$ 1.55	61.81 $\pm$ 1.80	68.42 $\pm$ 1.38	84.64 $\pm$ 1.13	73.19 $\pm$ 0.99	87.86 $\pm$ 0.77	71.11
GloGNN	84.32 $\pm$ 4.15	87.06 $\pm$ 3.53	83.51 $\pm$ 4.26	37.35 $\pm$ 1.30	57.54 $\pm$ 1.39	69.78 $\pm$ 2.42	88.31 $\pm$ 1.13	77.41 $\pm$ 1.65	89.62 $\pm$ 0.35	74.99
ACM-GCN	87.84 $\pm$ 4.40	88.43 $\pm$ 3.22	85.14 $\pm$ 6.07	36.28 $\pm$ 1.09	54.40 $\pm$ 1.88	66.93 $\pm$ 1.85	87.91 $\pm$ 0.95	77.32 $\pm$ 1.70	90.00 $\pm$ 0.52	74.92
GCNII	77.57 $\pm$ 3.83	80.39 $\pm$ 3.40	77.86 $\pm$ 3.79	37.44 $\pm$ 1.30	38.47 $\pm$ 1.58	63.86 $\pm$ 3.04	88.37 $\pm$ 1.25	77.33 $\pm$ 1.48	90.15 $\pm$ 0.43	70.16
CGNN	71.35 $\pm$ 4.05	74.31 $\pm$ 7.26	66.22 $\pm$ 7.69	35.95 $\pm$ 0.86	29.24 $\pm$ 1.09	46.89 $\pm$ 1.66	87.10 $\pm$ 1.35	76.91 $\pm$ 1.81	87.70 $\pm$ 0.49	63.96
GRAND	75.68 $\pm$ 7.25	79.41 $\pm$ 3.64	82.16 $\pm$ 7.09	35.62 $\pm$ 1.01	40.05 $\pm$ 1.50	54.67 $\pm$ 2.54	87.36 $\pm$ 0.96	76.46 $\pm$ 1.77	89.02 $\pm$ 0.51	68.94
BLEND	83.24 $\pm$ 4.65	84.12 $\pm$ 3.56	85.95 $\pm$ 6.82	35.63 $\pm$ 1.01	43.06 $\pm$ 1.39	60.11 $\pm$ 2.09	88.09 $\pm$ 1.22	76.63 $\pm$ 1.60	89.24 $\pm$ 0.42	71.79
Sheaf	85.05 $\pm$ 5.51	89.41 $\pm$ 4.74	84.86 $\pm$ 4.71	37.81 $\pm$ 1.15	56.34 $\pm$ 1.32	68.04 $\pm$ 1.58	86.90 $\pm$ 1.13	76.70 $\pm$ 1.57	89.49 $\pm$ 0.40	75.06
GRAFF	88.38 $\pm$ 4.53	87.45 $\pm$ 2.94	83.24 $\pm$ 6.49	36.09 $\pm$ 0.81	54.52 $\pm$ 1.37	71.08 $\pm$ 1.75	87.61 $\pm$ 0.97	76.92 $\pm$ 1.70	88.95 $\pm$ 0.52	74.92
GREAD-BS	88.92 $\pm$ 3.72	89.41 $\pm$ 3.30	86.49 $\pm$ 7.15	37.90 $\pm$ 1.17	59.22 $\pm$ 1.44	71.38 $\pm$ 1.31	88.57 $\pm$ 0.66	77.60 $\pm$ 1.81	90.23 $\pm$ 0.55	76.64
GREAD-F	89.73 $\pm$ 4.49	86.47 $\pm$ 4.84	86.49 $\pm$ 5.13	36.72 $\pm$ 0.66	46.16 $\pm$ 1.44	65.20 $\pm$ 1.40	88.39 $\pm$ 0.91	77.40 $\pm$ 1.54	90.09 $\pm$ 0.31	74.13
GREAD-AC	85.95 $\pm$ 2.65	86.08 $\pm$ 3.56	87.03 $\pm$ 4.95	37.21 $\pm$ 1.10	45.10 $\pm$ 2.11	65.09 $\pm$ 1.08	88.29 $\pm$ 0.67	77.38 $\pm$ 1.53	90.10 $\pm$ 0.36	73.71
GREAD-Z	87.30 $\pm$ 5.68	86.29 $\pm$ 4.32	85.68 $\pm$ 5.41	37.01 $\pm$ 1.11	46.25 $\pm$ 1.72	62.70 $\pm$ 2.30	88.31 $\pm$ 1.10	77.39 $\pm$ 1.90	90.11 $\pm$ 0.27	73.45
GREAD-ST	81.08 $\pm$ 5.67	86.67 $\pm$ 3.01	86.22 $\pm$ 5.98	37.66 $\pm$ 0.90	45.83 $\pm$ 1.40	63.03 $\pm$ 1.32	88.47 $\pm$ 1.19	77.25 $\pm$ 1.47	90.13 $\pm$ 0.36	72.93
GREAD-FB	86.76 $\pm$ 5.05	87.65 $\pm$ 3.17	86.22 $\pm$ 5.85	37.40 $\pm$ 0.55	50.83 $\pm$ 2.27	66.05 $\pm$ 1.21	88.03 $\pm$ 0.78	77.28 $\pm$ 1.73	90.07 $\pm$ 0.45	74.48
GREAD-FB*	87.03 $\pm$ 3.97	88.04 $\pm$ 1.63	85.95 $\pm$ 5.64	37.70 $\pm$ 0.51	50.57 $\pm$ 1.52	65.83 $\pm$ 1.10	88.01 $\pm$ 0.80	77.42 $\pm$ 1.93	90.08 $\pm$ 0.46	74.51

Experimental Results

- We also have other experimental results saying that
 - GREAD-BS works well irrespective of the homophily rate.
 - GREAD-BS alleviates the oversmoothing problem.



GNNs: A Brief History

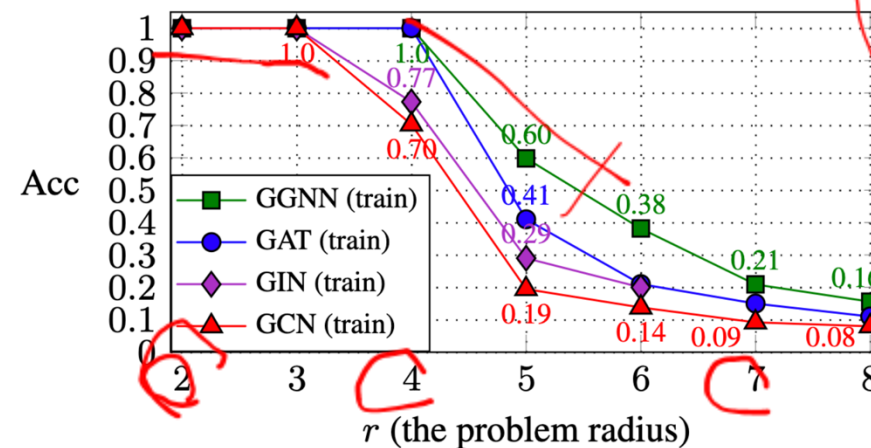
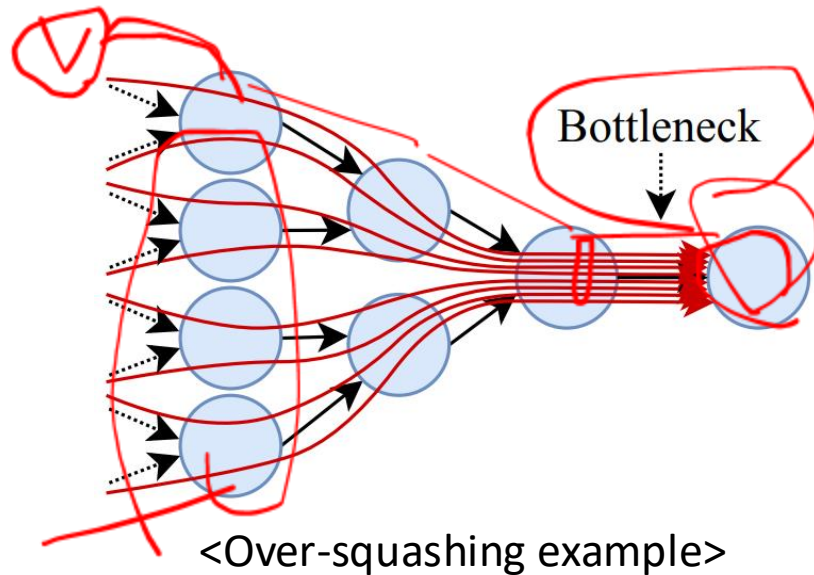


Overcoming the over-squashing problem

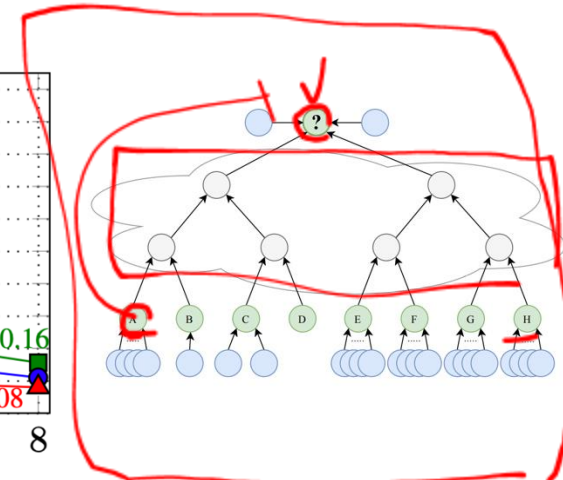
<Choi et al., PANDA: Expanded Width-Aware Message Passing Beyond Rewiring,
ICML 2024>

Over-squashing problem

- The over-squashing problem was introduced in [Alon & Yahav, 2021].
 - information from a node's exponentially-growing receptive field is compressed into a fixed-size vector.



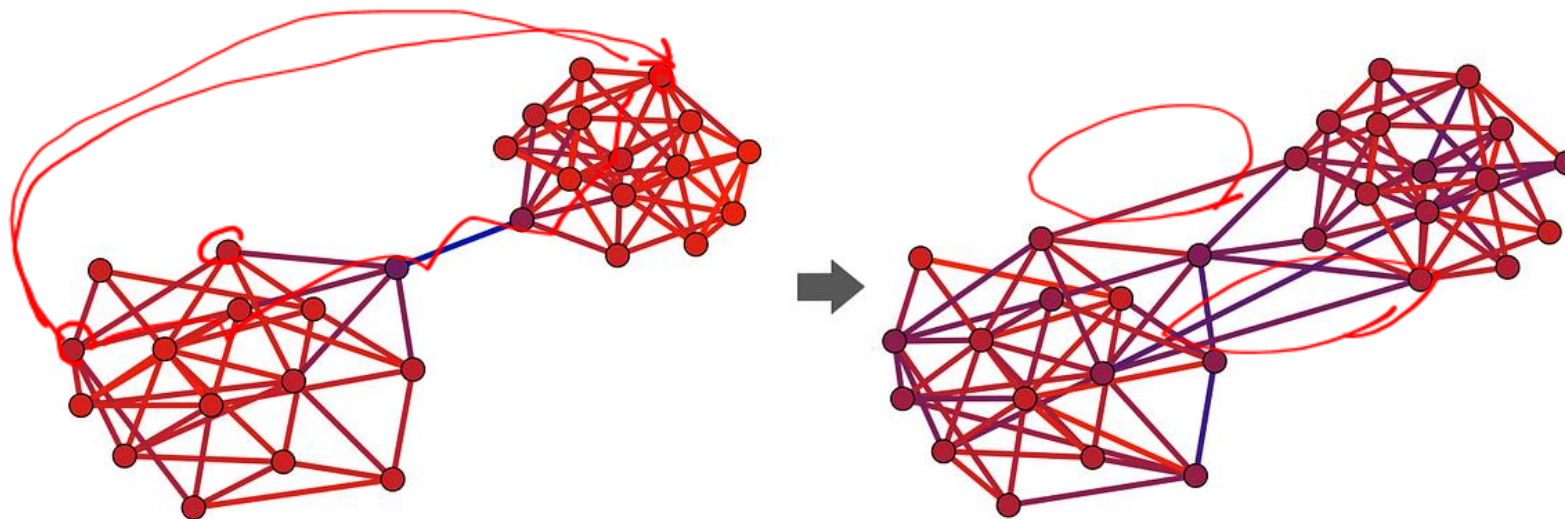
<Accuracy across problem radius (tree depth)>



Long-distance dependency + Fast volume growth = Over-squashing

Over-squashing Problem – contd.

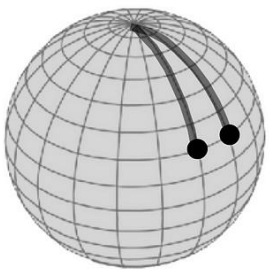
- The rewiring method is popular to solve the over-squashing problem [Topping et al., 2021, Karhadkar et al., 2022, Arnaiz-Rodríguez et al., 2022, Deac et al., 2022].



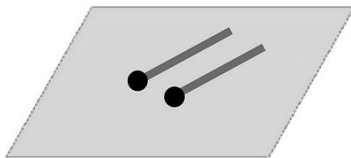
<An example of rewiring method in [Topping et al., 2021]>

Over-squashing problem – contd.

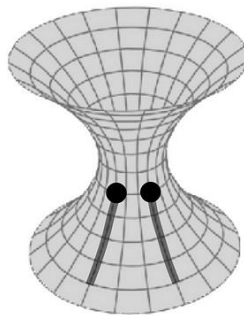
- Following the study by Alon & Yahav, it became popular to find indicators for over-squashing and propose rewiring methods.
- The Ricci curvature was used in [Topping et al., 2021].
 - In differential geometry, a natural object that allows us to distinguish different geometries is the Ricci curvature.
 - The over-squashing problem is caused by strongly negatively-curved edges.



Spherical (>0)

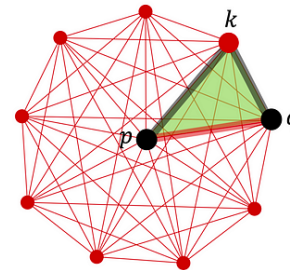


Euclidean ($=0$)

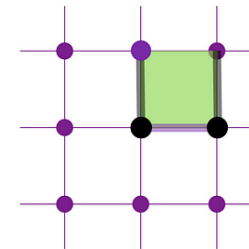


Hyperbolic (<0)

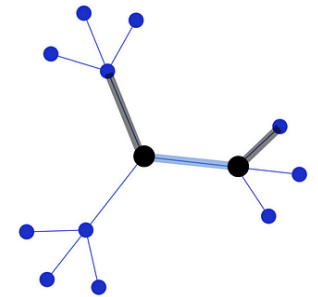
<Ricci curvature on manifolds>



Clique (>0)



Grid ($=0$)

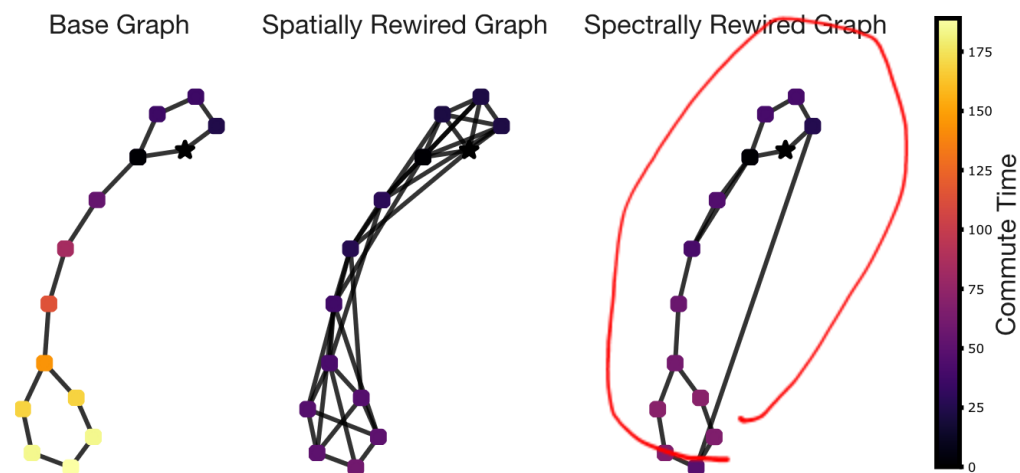


Tree (<0)

<Ricci curvature on graphs>

Over-squashing problem – contd.

- Large commute-time distances contribute to oversquashing.
 - Spectral rewiring: increase the Cheeger constant of the graph (“clusterdness”), which leads to lower commute time [Karhadkar et al., 2022, Arnaiz-Rodríguez et al., 2022].
 - Spatial rewiring: inserting edges reduces the total effective resistance of the graph (=commute-time distance up to scale) [Topping et al., 2021, Deac et al., 2022].



<Effect of different rewiring methods on the graph connectivity in [Di Giovanni et al., 2023]>

Over-squashing problem – contd.

- Let us consider an MPNN (GNN) of the following form.

$$\mathbf{h}_v^{(\ell+1)} = \phi^{(\ell)} \left(\mathbf{h}_v^{(\ell)}, \psi^{(\ell)}(\{\mathbf{h}_u^{(\ell)} : u \in \mathcal{N}(v)\}) \right)$$

- Small Jacobian norms indicate poor information propagations [Di Giovanni et al., 2023].
 - L = depth (number of layers)
 - p = width (hidden dimension)
 - z = Lipschitz constant
 - w = maximum element of weight matrices

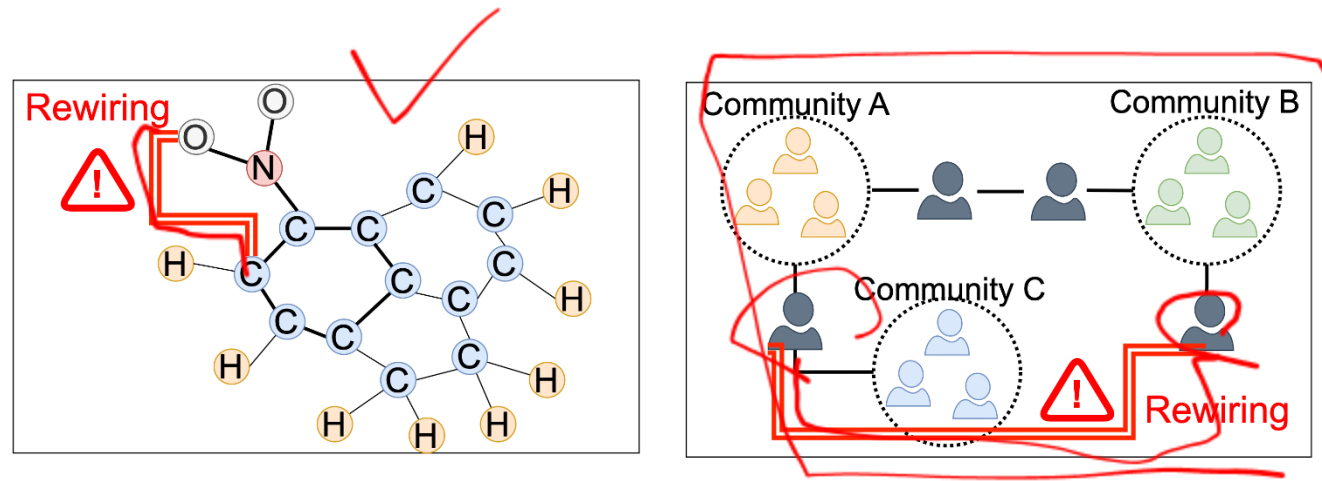
Theorem (Sensitivity bound): For any $u, v \in V$

$$\left\| \frac{\partial \mathbf{h}_v^{(\ell)}}{\partial \mathbf{h}_u^{(0)}} \right\|_1 \leq (zwp)^L (I + A)_{ij}^L$$

model topology

Limitations of rewiring methods

- Existing rewiring methods only focus on rewiring that changes the graph topology to address over-squashing.
- The rewiring methods can inadvertently introduce inaccuracies within domain-specific contexts.



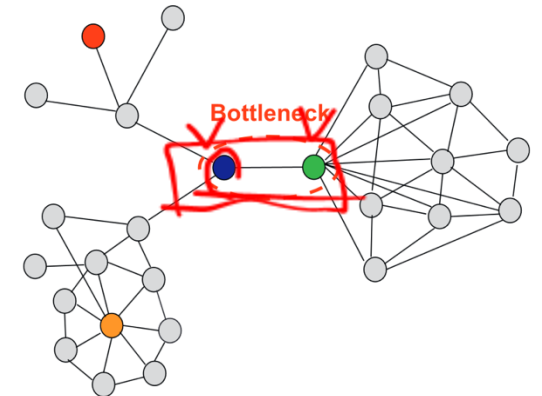
<Potential pitfalls of rewiring in domain-specific graphs>

Motivations

- Di Giovanni et al. (2023) provide a theoretical justification that increasing the width of the model (i.e., the hidden dimension) can also improve its sensitivity.

Theorem (Sensitivity bound): For any $u, v \in V$

$$\left\| \frac{\partial \mathbf{h}_v^{(\ell)}}{\partial \mathbf{h}_u^{(0)}} \right\|_1 \leq \underbrace{(zw_p)^L}_{\text{model}} \underbrace{(I + A)^L}_{\text{topology}}_{ij}$$



<Bottleneck nodes [Yu et al, 2007]>

- We aim to design a new message passing paradigm that mitigates over-squashing by selectively expanding the widths of bottleneck nodes.

Proposed method

- We can define bottleneck nodes as high centrality nodes in terms of betweenness centrality and so on [Yu et al., 2007, Topping et al., 2022].
- Increasing the hidden widths of the bottleneck nodes enables capturing more information.

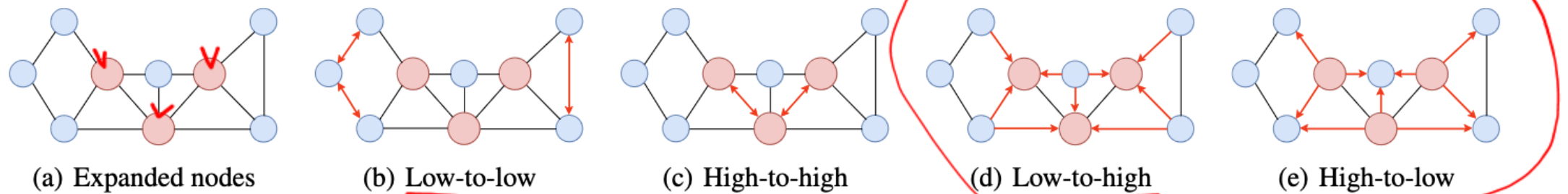
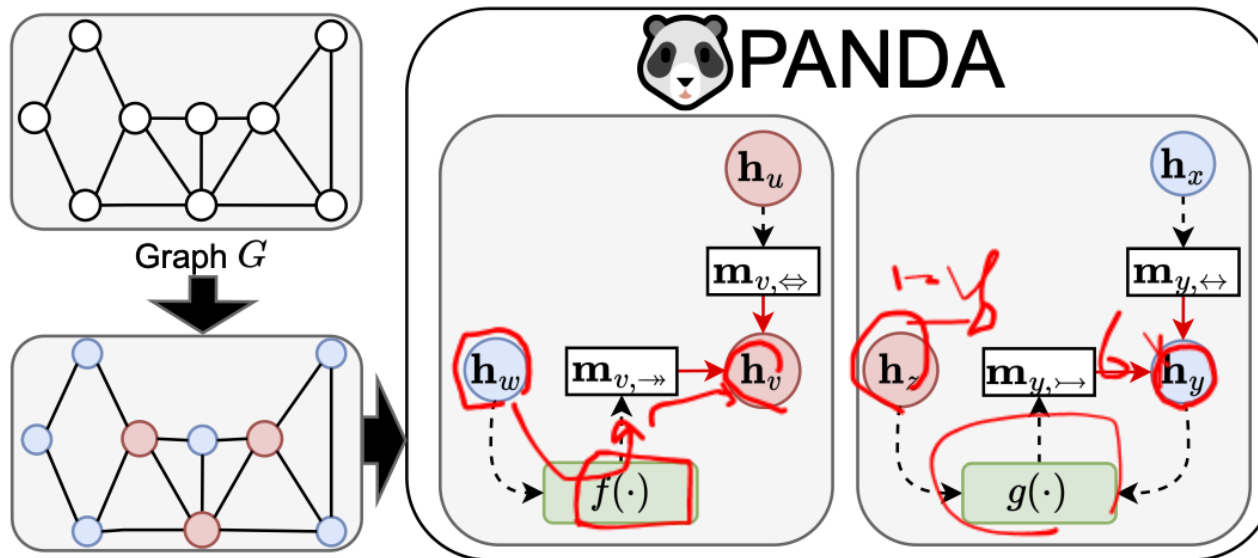


Figure 3. Examples of PANDA's message passing mechanism. The size of the node indicates the size of hidden dimension.

Proposed method – contd.

- Our PANDA message passing works in the following way.
 - First, we selectively expand widths according to centrality measures.
 - Then, our PANDA message passing enables signal propagation among nodes with different widths (low and high-width nodes).



$f(\cdot)$: A linear transformation that expands the width of low-dim nodes.

$g(\cdot)$: A dimension selector that selects dimensions to be propagated from high-dim nodes to low-dim nodes.

Experimental Results

Table 1. Results of PANDA and baselines for GCN and GIN. We show the best three in red (first), blue (second), and purple (third).

Method	REDDIT-BINARY	IMDB-BINARY	MUTAG	ENZYMES	PROTEINS	COLLAB
GCN (None)	68.255 $\pm$ 1.098	49.770 $\pm$ 0.817	72.150 $\pm$ 2.442	27.667 $\pm$ 1.164	70.982 $\pm$ 0.737	33.784 $\pm$ 0.488
+ Last Layer FA	68.485 $\pm$ 0.945	48.980 $\pm$ 0.945	70.050 $\pm$ 2.027	26.467 $\pm$ 1.204	71.018 $\pm$ 0.963	33.320 $\pm$ 0.435
+ Every Layer FA	48.490 $\pm$ 1.044	48.170 $\pm$ 0.801	70.450 $\pm$ 1.960	18.333 $\pm$ 1.038	60.036 $\pm$ 0.925	51.798 $\pm$ 0.419
+ DIGL	49.980 $\pm$ 0.680	49.910 $\pm$ 0.841	71.350 $\pm$ 2.391	27.517 $\pm$ 1.053	70.607 $\pm$ 0.731	15.530 $\pm$ 0.294
+ SDRF	68.620 $\pm$ 0.851	49.400 $\pm$ 0.904	71.050 $\pm$ 1.872	28.367 $\pm$ 1.174	70.920 $\pm$ 0.792	33.448 $\pm$ 0.472
+ FoSR	70.330 $\pm$ 0.727	49.660 $\pm$ 0.864	80.000 $\pm$ 1.574	25.067 $\pm$ 0.994	73.420 $\pm$ 0.811	33.836 $\pm$ 0.584
+ BORF	Time-out	50.100 $\pm$ 0.900	75.800 $\pm$ 1.900	24.700 $\pm$ 1.000	71.000 $\pm$ 0.800	Time-out
+ GTR	68.990 $\pm$ 0.610	49.920 $\pm$ 0.990	79.100 $\pm$ 1.860	27.520 $\pm$ 0.990	72.590 $\pm$ 2.480	33.050 $\pm$ 0.400
+ CT-Layer	51.580 $\pm$ 1.019	50.320 $\pm$ 0.944	75.899 $\pm$ 3.024	17.383 $\pm$ 1.030	60.357 $\pm$ 1.060	52.146 $\pm$ 0.415
+ PANDA	80.690 $\pm$ 0.721	63.760 $\pm$ 1.012	85.750 $\pm$ 1.396	31.550 $\pm$ 1.230	76.000 $\pm$ 0.774	68.400 $\pm$ 0.452
GIN (None)	86.785 $\pm$ 1.056	70.180 $\pm$ 0.992	77.700 $\pm$ 0.360	33.800 $\pm$ 0.115	70.804 $\pm$ 0.827	72.992 $\pm$ 0.384
+ Last Layer FA	90.220 $\pm$ 0.475	70.910 $\pm$ 0.788	83.450 $\pm$ 1.742	47.400 $\pm$ 1.387	72.304 $\pm$ 0.666	75.056 $\pm$ 0.406
+ Every Layer FA	50.360 $\pm$ 0.684	49.160 $\pm$ 0.870	72.550 $\pm$ 3.016	28.383 $\pm$ 1.052	70.375 $\pm$ 0.910	32.984 $\pm$ 0.390
+ DIGL	76.035 $\pm$ 0.774	64.390 $\pm$ 0.907	79.700 $\pm$ 2.150	35.717 $\pm$ 1.198	70.759 $\pm$ 0.774	54.504 $\pm$ 0.410
+ SDRF	86.440 $\pm$ 0.590	69.720 $\pm$ 1.152	78.400 $\pm$ 2.803	35.817 $\pm$ 1.094	69.813 $\pm$ 0.792	72.958 $\pm$ 0.419
+ FoSR	87.350 $\pm$ 0.598	71.210 $\pm$ 0.919	78.400 $\pm$ 2.803	29.200 $\pm$ 1.367	75.107 $\pm$ 0.817	73.278 $\pm$ 0.416
+ BORF	Time-out	71.300 $\pm$ 1.500	80.800 $\pm$ 2.500	35.500 $\pm$ 1.200	74.200 $\pm$ 0.800	Time-out
+ GTR	86.980 $\pm$ 0.660	71.280 $\pm$ 0.860	77.600 $\pm$ 2.840	30.570 $\pm$ 1.420	73.130 $\pm$ 0.690	72.930 $\pm$ 0.420
+ CT-Layer	54.589 $\pm$ 1.757	50.000 $\pm$ 0.974	56.850 $\pm$ 4.253	16.583 $\pm$ 0.907	61.107 $\pm$ 1.184	52.304 $\pm$ 0.605
+ PANDA	91.055 $\pm$ 0.402	72.560 $\pm$ 0.917	88.750 $\pm$ 1.570	46.200 $\pm$ 1.410	75.759 $\pm$ 0.856	75.200 $\pm$ 0.481

Experimental Results – contd.

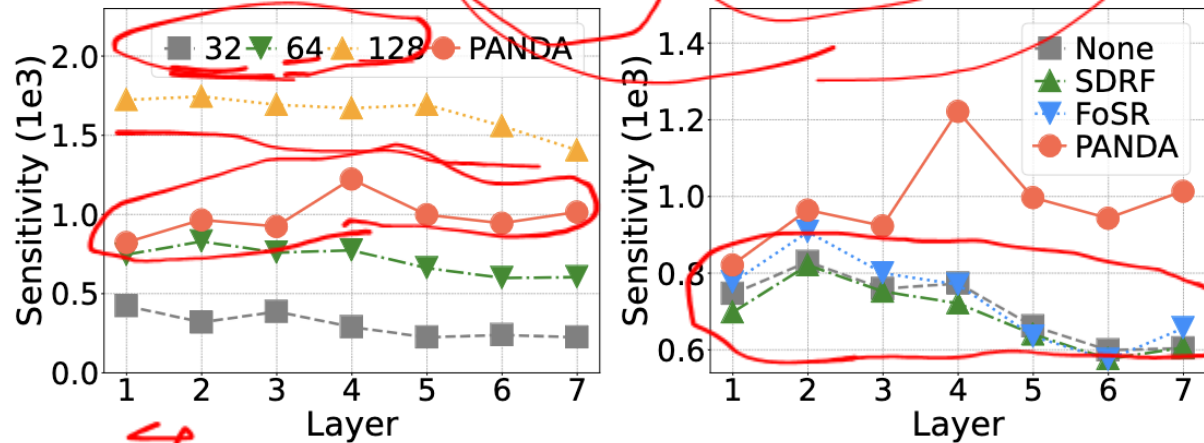
Method	REDDIT-BINARY	IMDB-BINARY	MUTAG	ENZYMES	PROTEINS	COLLAB
R-GCN	49.850 $\pm$ 0.653	50.012 $\pm$ 0.917	69.250 $\pm$ 2.085	28.600 $\pm$ 1.186	69.518 $\pm$ 0.725	33.602 $\pm$ 1.047
PANDA-GCN	80.690 $\pm$ 0.721	63.760 $\pm$ 1.012	85.750 $\pm$ 1.396	31.550 $\pm$ 1.230	76.000 $\pm$ 0.774	68.400 $\pm$ 0.452

<PANDA-GCN vs. R-GCN>

$C(\mathcal{G})$	REDDIT-BINARY	IMDB-BINARY	MUTAG	ENZYMES	PROTEINS	COLLAB
Degree	80.690 $\pm$ 0.721	62.100 $\pm$ 1.043	85.200 $\pm$ 1.568	31.117 $\pm$ 1.258	75.375 $\pm$ 0.800	68.162 $\pm$ 0.471
Betweenness	80.000 $\pm$ 0.659	59.630 $\pm$ 1.152	85.750 $\pm$ 1.396	29.600 $\pm$ 1.208	74.589 $\pm$ 0.791	67.844 $\pm$ 0.547
Closeness	79.700 $\pm$ 0.664	61.160 $\pm$ 0.992	84.700 $\pm$ 1.554	29.967 $\pm$ 1.231	76.000 $\pm$ 0.774	68.400 $\pm$ 0.452
PageRank	80.340 $\pm$ 0.826	63.760 $\pm$ 1.012	85.450 $\pm$ 1.569	31.550 $\pm$ 1.230	74.098 $\pm$ 0.851	67.540 $\pm$ 0.500
Load	79.500 $\pm$ 0.732	59.840 $\pm$ 1.153	85.700 $\pm$ 1.549	28.167 $\pm$ 1.090	74.188 $\pm$ 0.814	67.802 $\pm$ 0.506

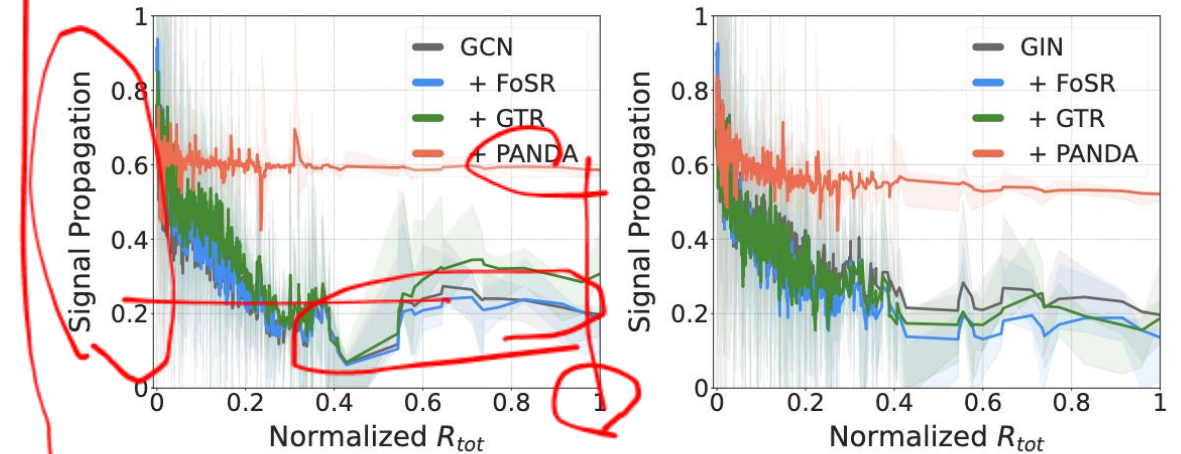
<Performance comparison by various centrality measures for PANDA-GCN>

Experimental Results – contd.



(a) Sensitivity w.r.t. p (b) Sensitivity w.r.t. methods

Empirical sensitivity across layers for GCN on MUTAG. Compared to other methods, PANDA shows higher sensitivity that is maintained even in deeper layer.



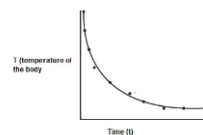
(a) GCN on REDDIT-BINARY (b) GIN on REDDIT-BINARY

The amount of signal propagated across the graph w.r.t. the normalized total effective resistance. PANDA maintains continuous information flows even under high bottleneck conditions.

Summary

GCN and Physics Inductive Bias

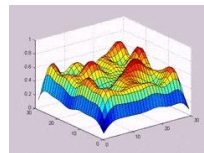
- Newton Law of Cooling
- Heat Diffusion Equation



VII. *Scala graduum Celsii.*
Celsius Descriptiones & signa.
Calor aeris hybernus ubi aqua incipit gelu rigescere. Innotescit hac calor accurate locando Thermometrum in nive compressa quo tempore gelu solvitur.
Calores aeris hybernici.
Calores aeris vernalis & autumnalis.
Calores aeris a fluvio.
Calor aeris meridiani circa medium Julii.
Calor maximus quem Thermometer ad con-



Issac Newton



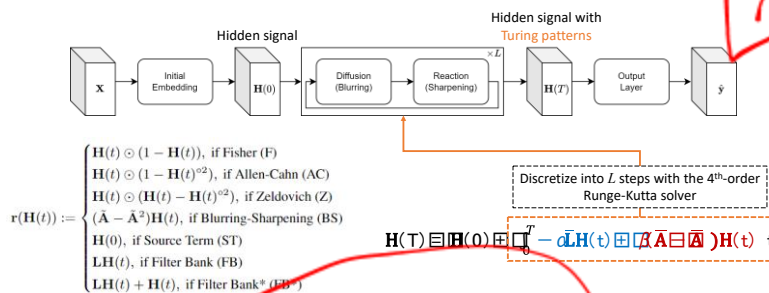
Heat equation (Wikipedia)



Joseph Fourier

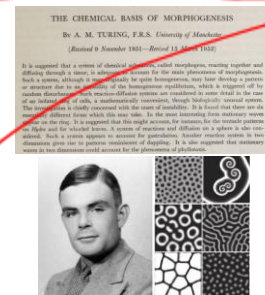
Anonymous (March–April 1701). "Scala graduum Caloris. Calorum Descriptiones & signa." *Philosophical Transactions*. 22 (270): 824–829

Architecture of GREAD-BS



Over-smoothing

Turing Pattern and Reaction-Diffusion Equation



Alan Turing. "The chemical basis of morphogenesis". *Phil. Trans. R. Soc. Lond. B*. 1952. <http://doi.org/10.1098/rstb.1952.0012>

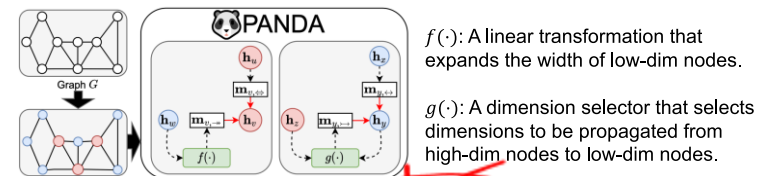
- The reaction-diffusion equations can be written as:

$$\frac{\partial \mathbf{x}}{\partial t} = \text{div}(\nabla \mathbf{x}) + r(\mathbf{x}).$$
- Reaction-Diffusion equations on graphs with an explicit discretization:

$$\mathbf{X}(t + s) = \mathbf{X}(t) + s \left(-\tilde{L}\mathbf{X}(t) + r(\mathbf{X}(t)) \right).$$

Proposed method

- Our PANDA message passing works in the following way.
 - First, we selectively expand widths according to centrality measures.
 - Then, our PANDA message passing enables signal propagation among nodes with different widths (low and high-width nodes).



Over-squashing

Tackling Over-smoothing and Over-squashing in Deep Learning on Graphs

“Making Decisions with AI” hosted by Yoontae Hwang (황윤태)

Jeongwhan Choi (최정환)
Postdoctoral Researcher

KAIST



BigDyL

[big di:l]

Big Data Analytics & Learning Laboratory
Korea Advanced Institute of Science & Technology

Thank You

JEONGWHAN.CHOI@KAIST.AC.KR

<https://jeongwhanchoi.com>