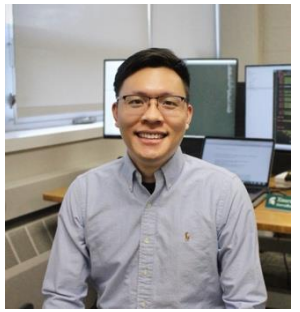
Adversarial Robustness in Graph Neural Networks: Recent Advances and New Frontier







Zhichao Hou[1], Minhua Lin[2], MohamadAli Torkamani[3], Suhang Wang[2], Xiaorui Liu[1]

[1]North Carolina State University
[2]Pennsylvania State University
[3]Amazon AWS

11:00 AM – 12:30 PM, Oct 9, DSAA
Cabrillo

Tutorial Outline



- **Introduction** of GNNs
- Adversarial **Attacks** on Graph
- Adversarial **Defenses** on Graph
- **Evaluation** of Robust GNNs
- **Summary** and Discussion

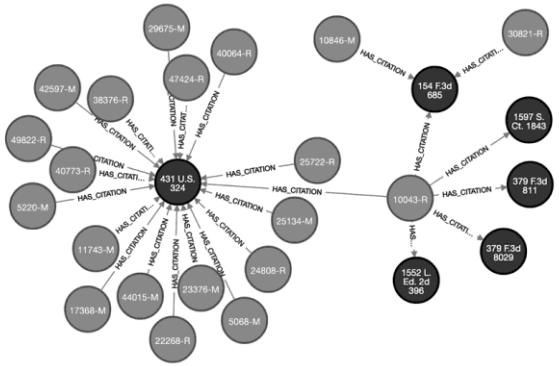
Overview

Presenter:
Xiaorui Liu

Introduction

- Brief Introduction of GNNs
- Foundation of GNNs
- Challenges in Robust GNNs

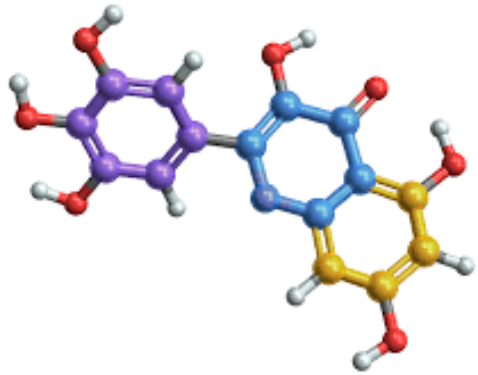
Graphs in Various Domains



Citation Graph



Social Graph



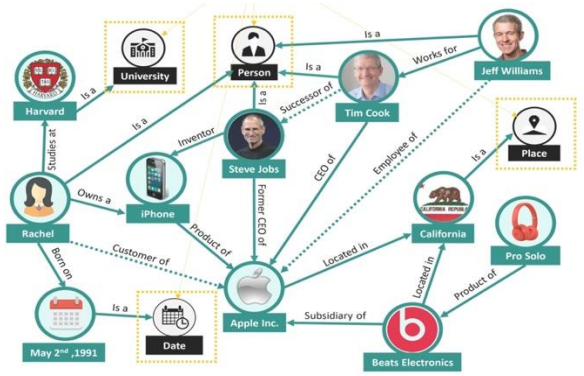
Biomedical Graph



Traffic Graph

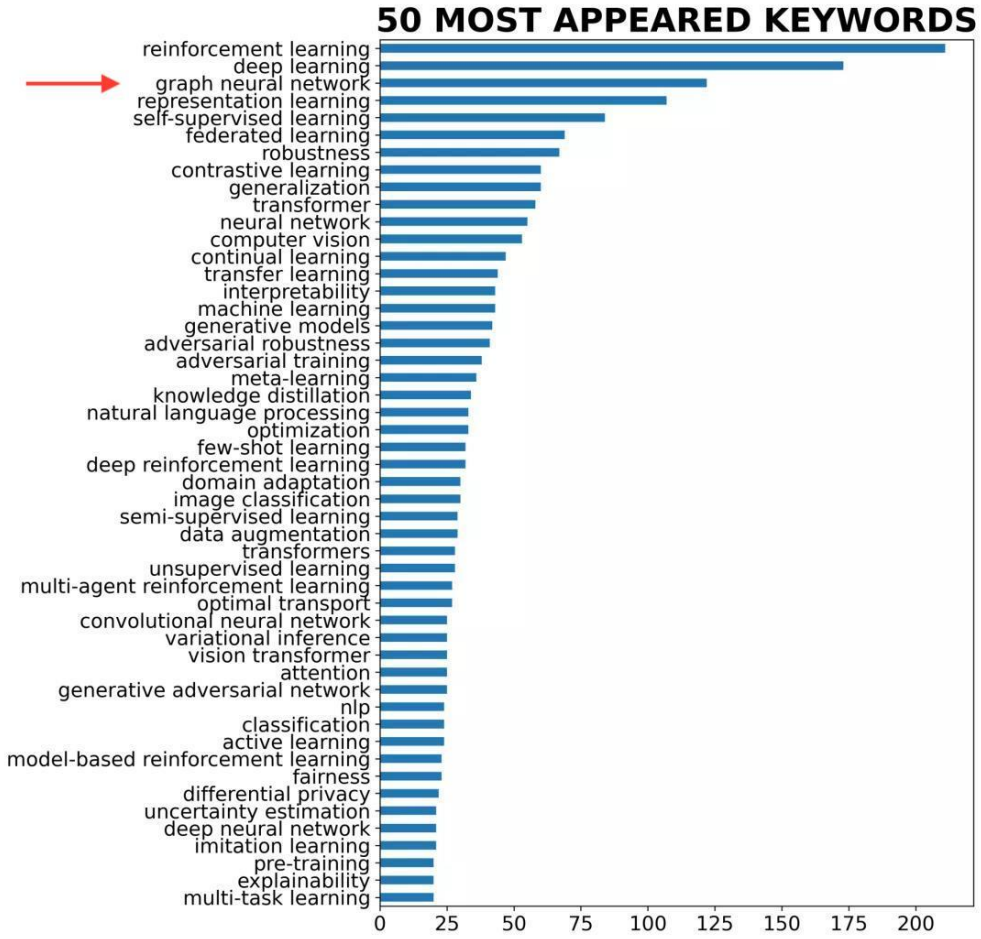


Web Graph

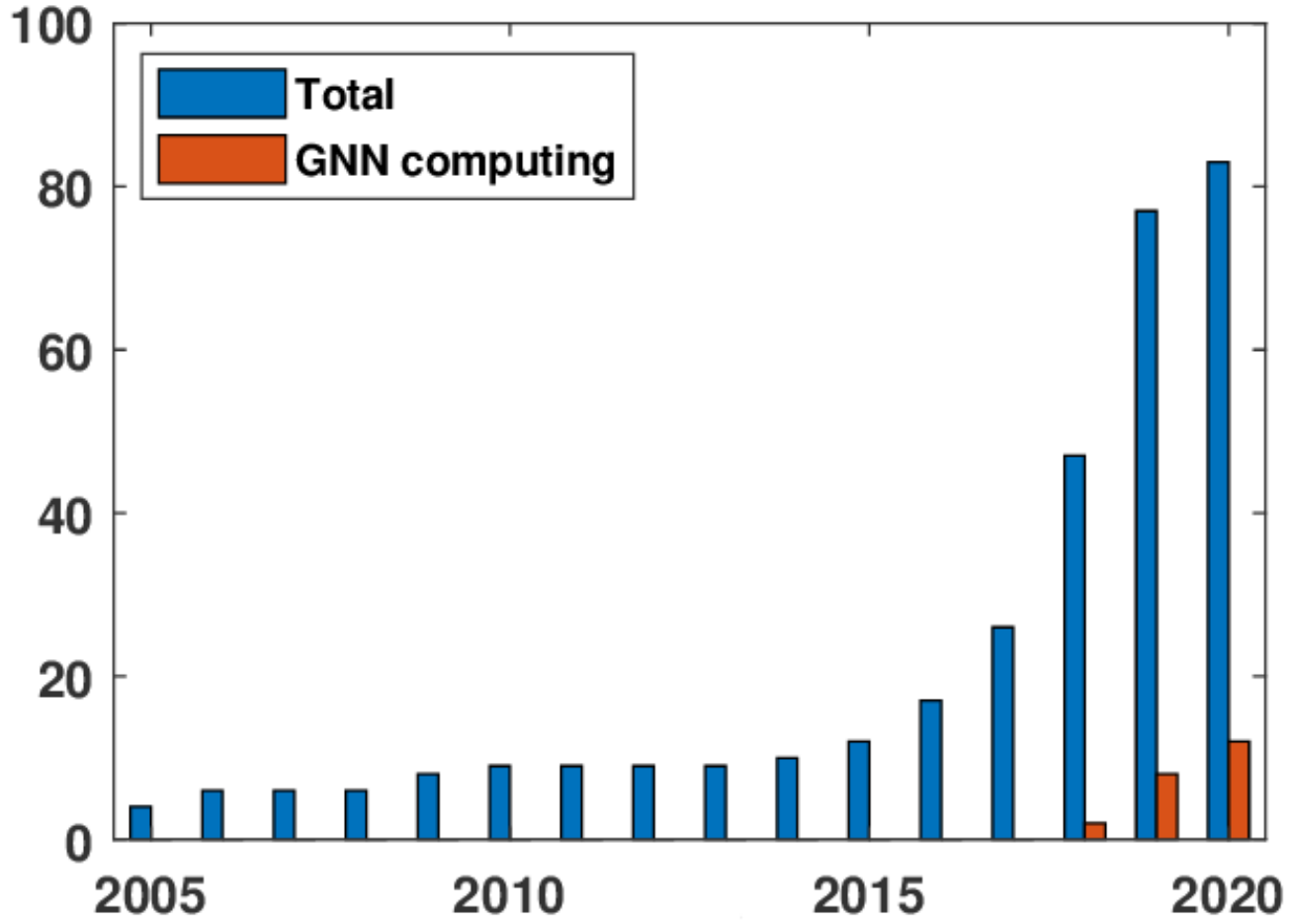


Knowledge Graph

Graph Machine Learning



Top 50 keywords in submitted research papers at ICLR 2022



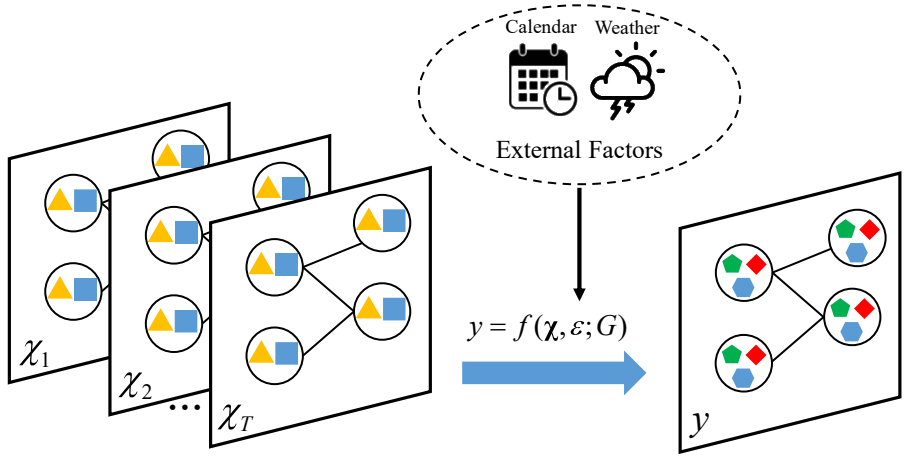
Number of publications for GNNs in general and for the subfield “GNN computing” in particular.

Applications of GNNs

- Traffic Prediction



Road Network

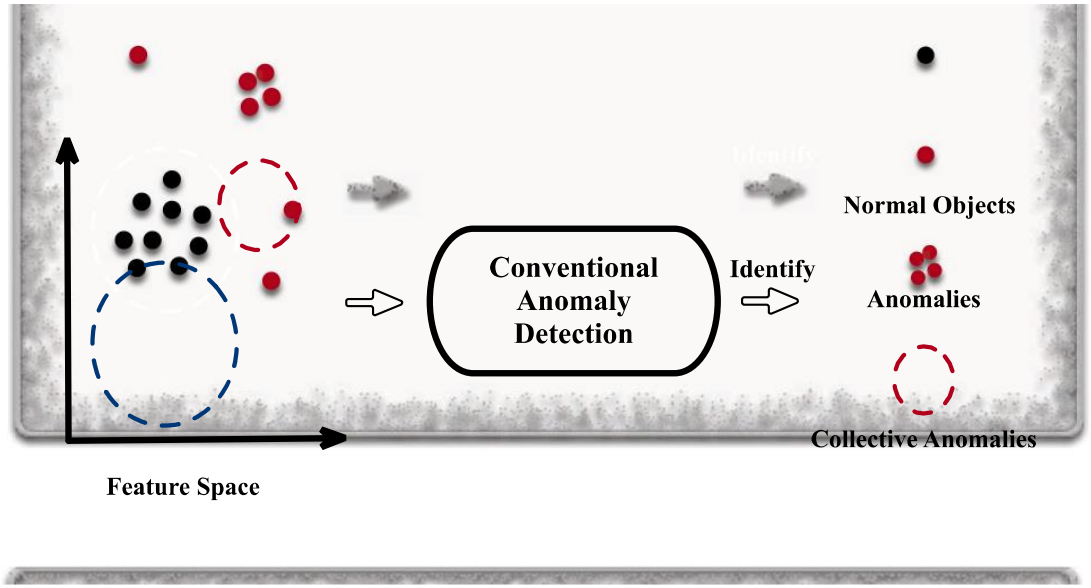


One Single-step Prediction

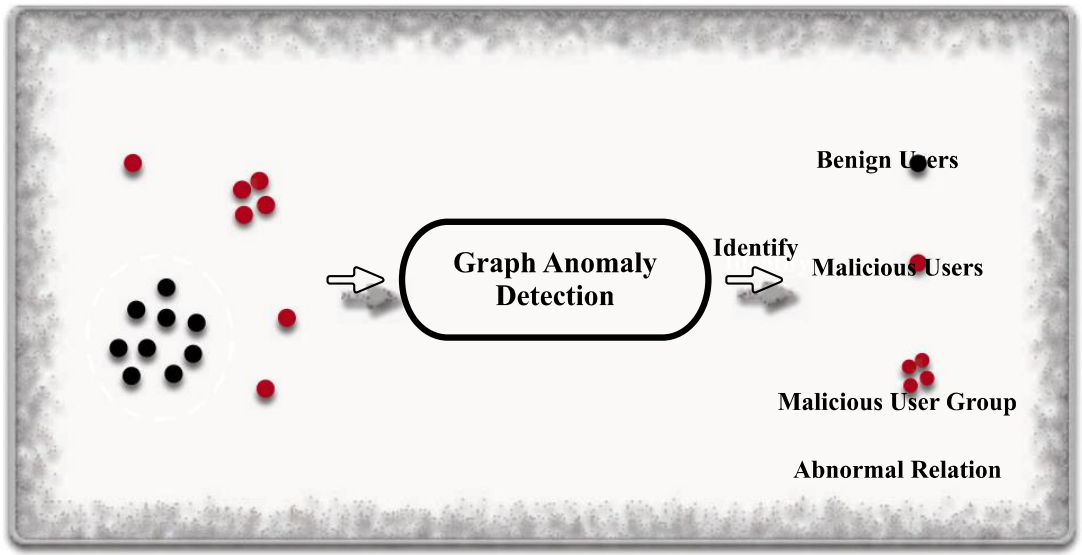
Jiang, W., & Luo, J. Graph Neural Network for Traffic Forecasting: A Survey. ArXiv Preprint. 2021

Applications of GNNs

- Anomaly Detection



Conventional Anomaly Detection

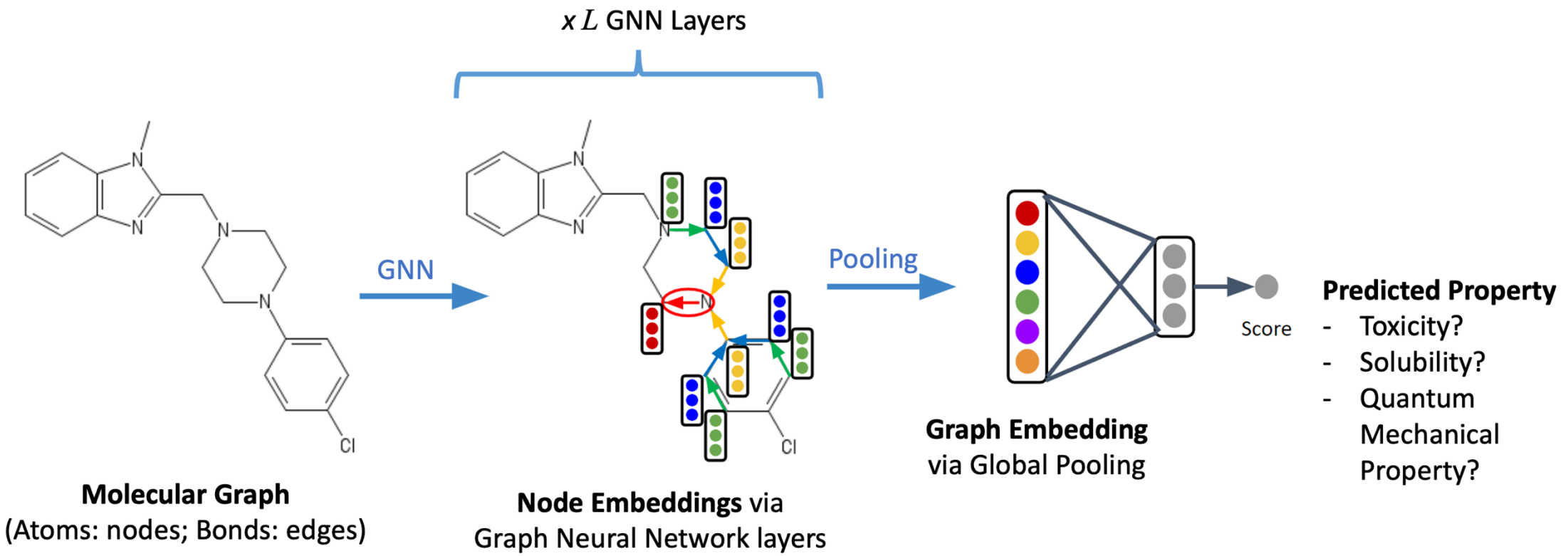


Graph-based Anomaly Detection

Ma, Xiaoxiao, et al. "A comprehensive survey on graph anomaly detection with deep learning." IEEE, 2021

Applications of GNNs

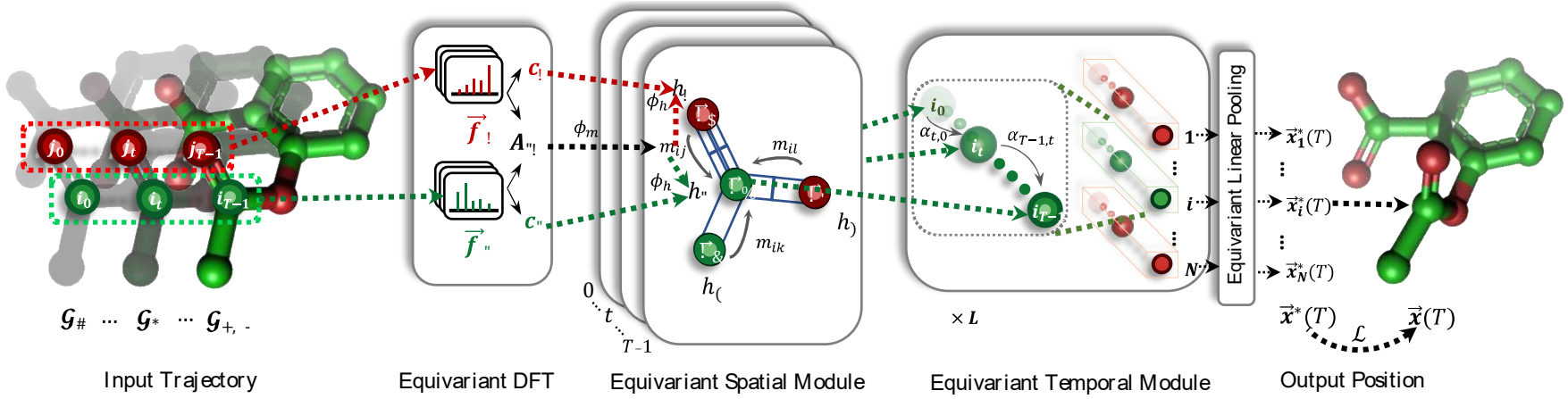
- Molecular Property Prediction



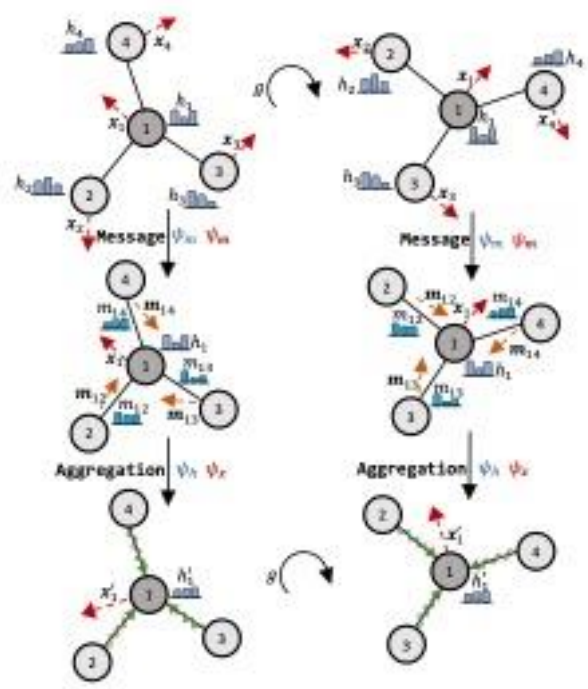
Wieder, Oliver, et al. "A compact review of molecular property prediction with graph neural networks." Drug Discovery Today: Technologies, 2020.

Applications of GNNs

- Molecular Dynamics



Equivariant Spatio-Tempral Graph Neural Networks



Equivariance

Definition 1 (Equivariance). *The function* $\phi : \mathcal{X} \mapsto \mathcal{Y}$ *is* G*-equivariant if it commutes with any transformation in* G*,*

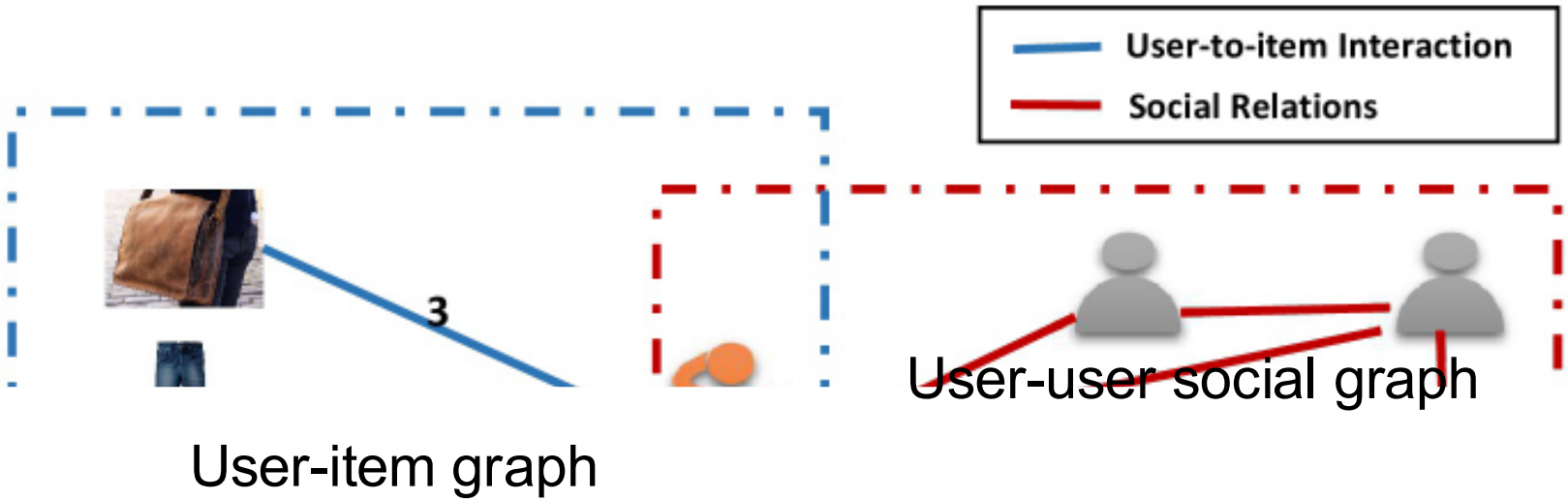
$$\phi(\rho_{\mathcal{X}}(g)x) = \rho_{\mathcal{Y}}(g)\phi(x), \forall g \in G, \tag{3}$$

where $\rho_{\mathcal{X}}$ *and* $\rho_{\mathcal{Y}}$ *are the group representations in the input and output space, respectively. Specifically,* ϕ *is called invariant if* $\rho_{\mathcal{Y}}$ *is the identity.*

Wu, L., Hou, Z., Yuan, J., Rong, Y., & Huang, W. Equivariant spatio-temporal attentive graph networks to simulate physical dynamics. NeurIPS, 2024.

Applications of GNNs

- Recommendation Systems



Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J., & Yin, D. Graph neural networks for social recommendation. In The world wide web conference, 2019.

Applications of GNNs

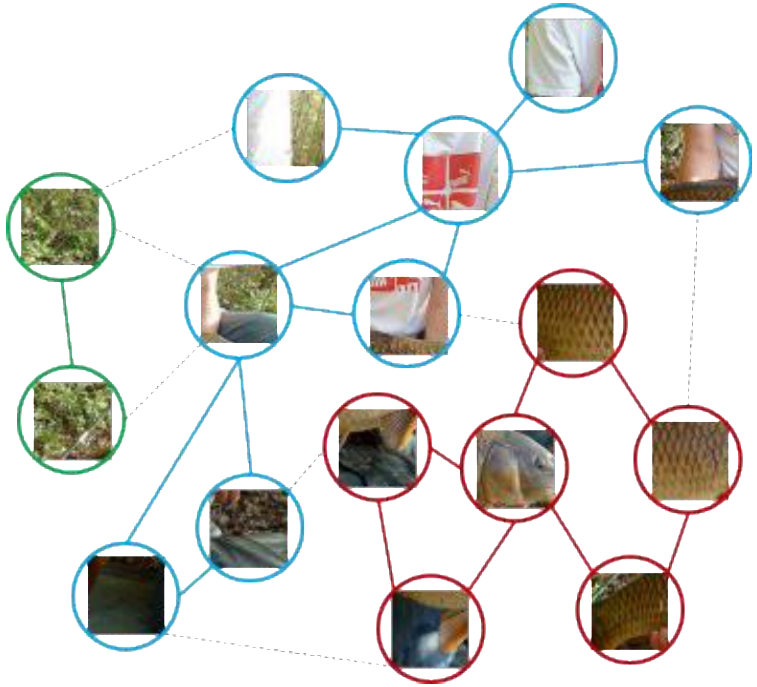
- Computer Vision



(a) Grid structure.



(b) Sequence structure.



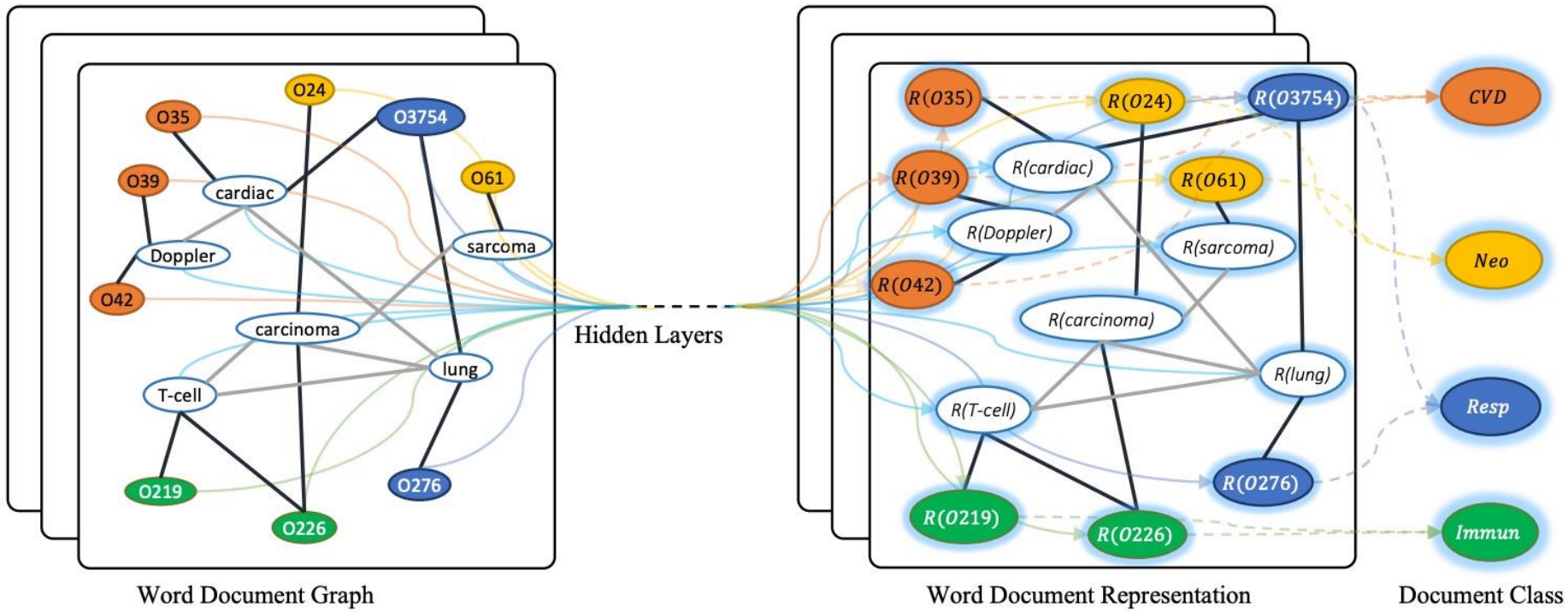
(b) Graph structure.

- Grid structure: the pixels or patches are ordered only by the spatial position.
- Sequence structure: the 2D image is transformed to a sequence of patches.
- Graph structure: the nodes are linked by its content and are not constrained by the local position.

Han, K., Wang, Y., Guo, J., Tang, Y., & Wu, E. Vision gnn: An image is worth graph of nodes. NeurIPS, 2022.

Applications of GNNs

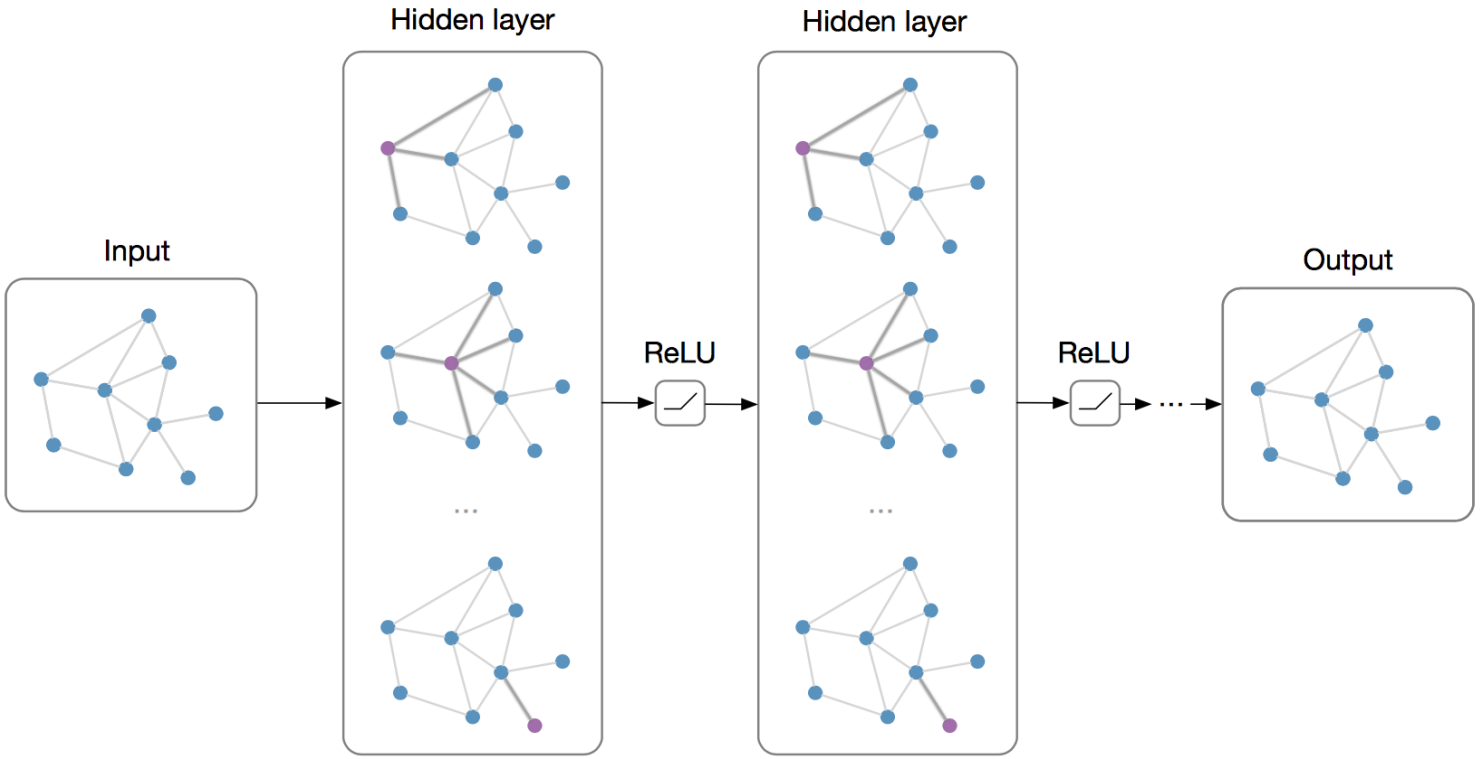
- Natural Language Processing



Yao, L., Mao, C., & Luo, Y. Graph convolutional networks for text classification. AAAI, 2019.

Foundation of GNNs

- Main idea: Pass messages between pairs of nodes and refine node representations.

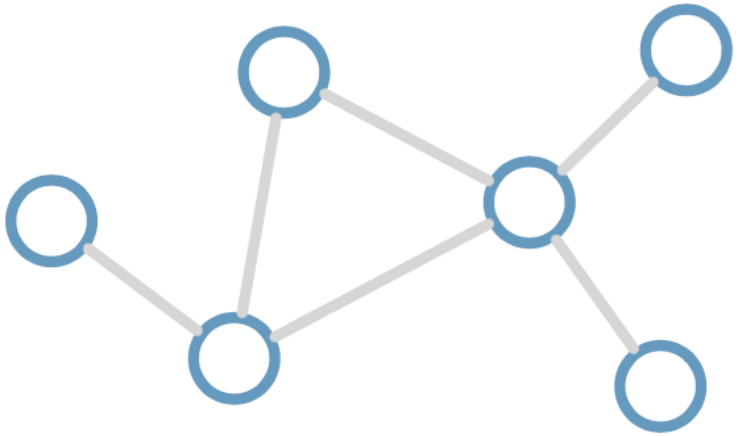


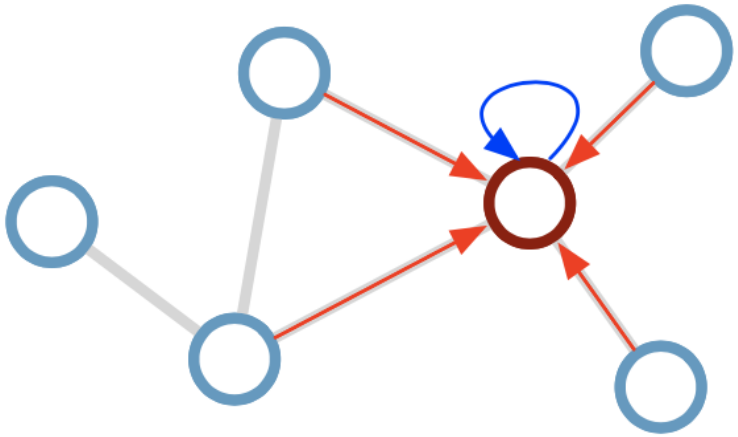
Notation: $\mathcal{G} = (\mathbf{A}, \mathbf{X})$

- Adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$
- Feature matrix $\mathbf{X} \in \mathbb{R}^{N \times F}$

Message Passing

- Main idea: Pass messages between pairs of nodes and refine node representations.





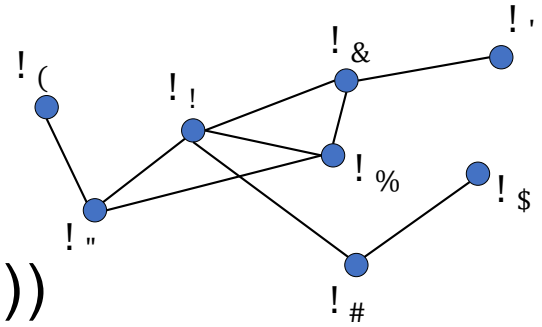
Representation Update: $$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

Matrix Representation

Adjacency Matrix: $A[i,j] = 1$ if v_i is adjacent to v_j

$A[i,j] = 0$, otherwise

Degree Matrix: $D = \mathrm{diag}(degree(v_1), \ldots, degree(v_N))$



Adjacency Matrix

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \in \mathbb{R}^{N \times N}$$

Degree Matrix

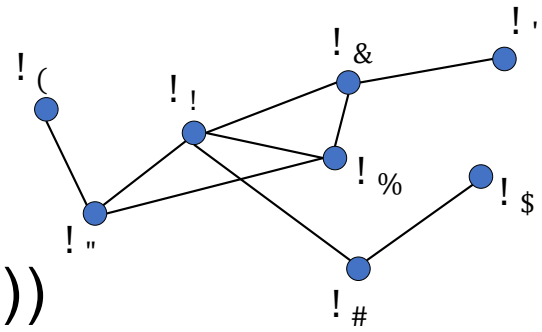
$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \in \mathbb{R}^{N \times N}$$

Substantial compuation cost and memory requirement

Matrix Representation

Adjacency Matrix: ! [", $] = 1 if ' ! is adjacent to ' "
! [", $] = 0, otherwise

Degree Matrix: D = diag($degree(v_1), \ldots, degree(v_N)$)



Sparse Adjacency Matrix

$$indices = \begin{bmatrix} 1 & 2 & 3 & 3 & 3 & 4 & 6 & 7 \\ 2 & 3 & 4 & 6 & 7 & 5 & 7 & 8 \end{bmatrix} \quad O(M)$$

$$values = [1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1]$$

"

Sparse Degree Matrix

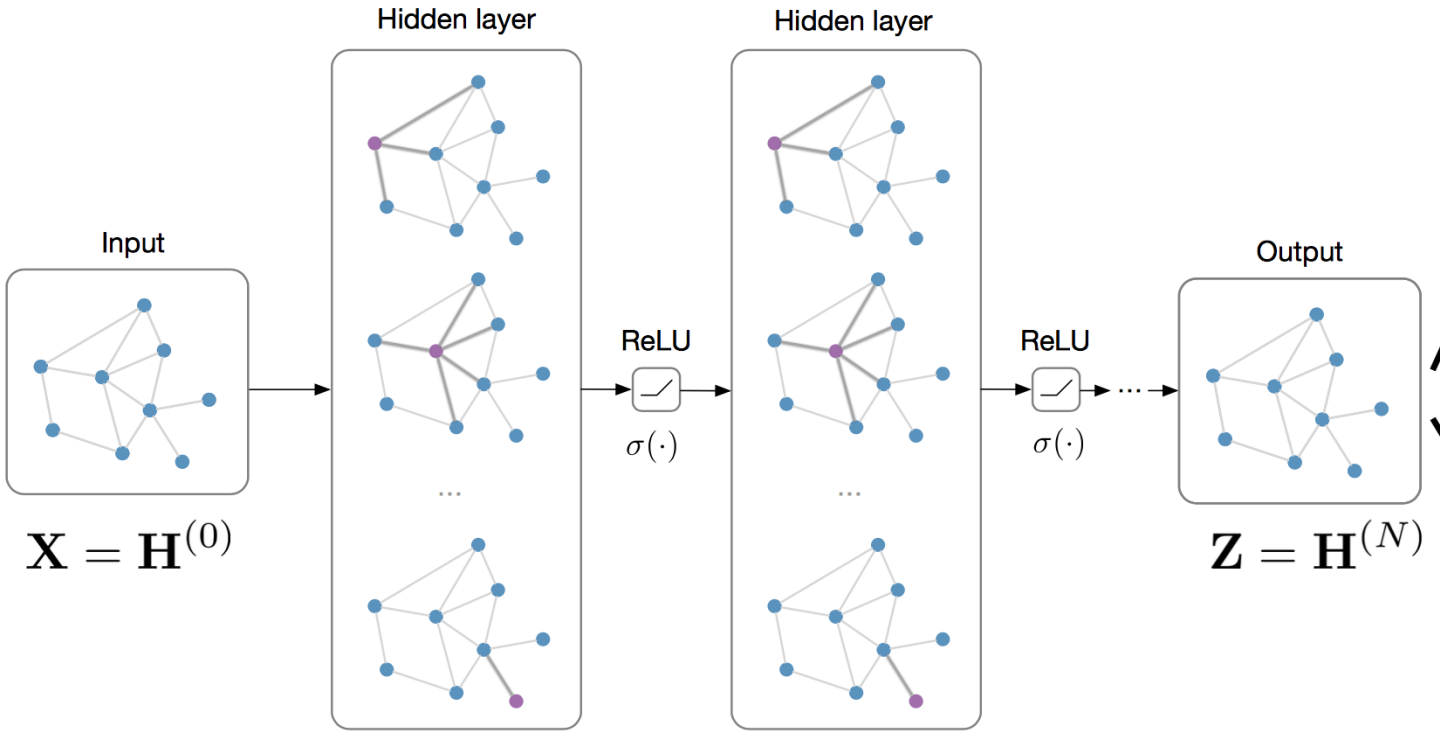
$$indices = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{bmatrix} \quad O(N)$$

$$values = [1 \quad 3 \quad 4 \quad 2 \quad 2 \quad 4 \quad 3]$$

!

Downstream Tasks

Input: Feature matrix $\mathbf{X} \in \mathbb{R}^{N \times E}$, preprocessed adjacency matrix $\hat{\mathbf{A}}$



Node classification:

$\mathrm{softmax}(\mathbf{z_n})$

e.g. Kipf & Welling (ICLR 2017)

Graph classification:

$\mathrm{softmax}(\sum_n \mathbf{z_n})$

e.g. Duvenaud et al. (NIPS 2015)

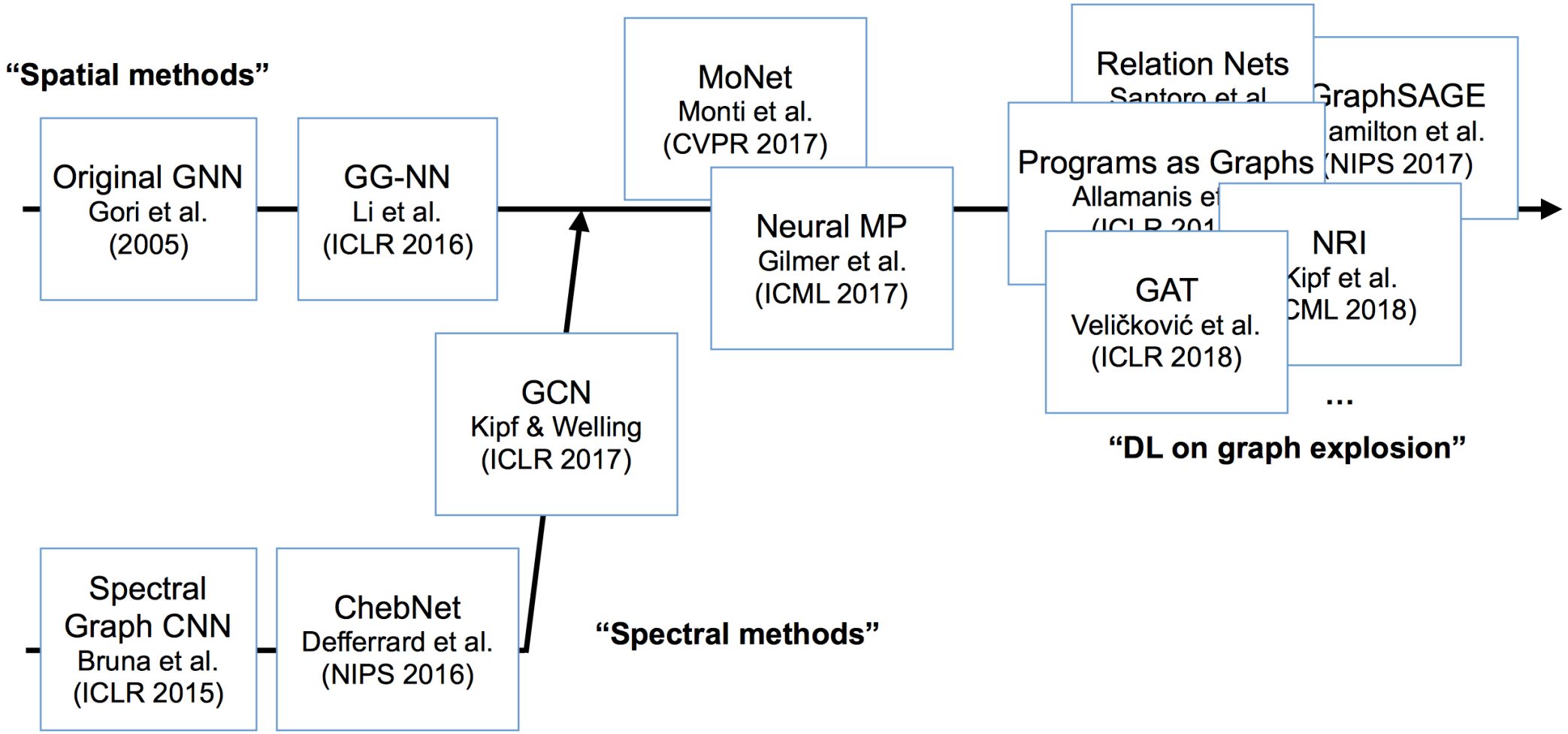
Link prediction:

$p(A_{ij}) = \sigma(\mathbf{z_i^T z_j})$

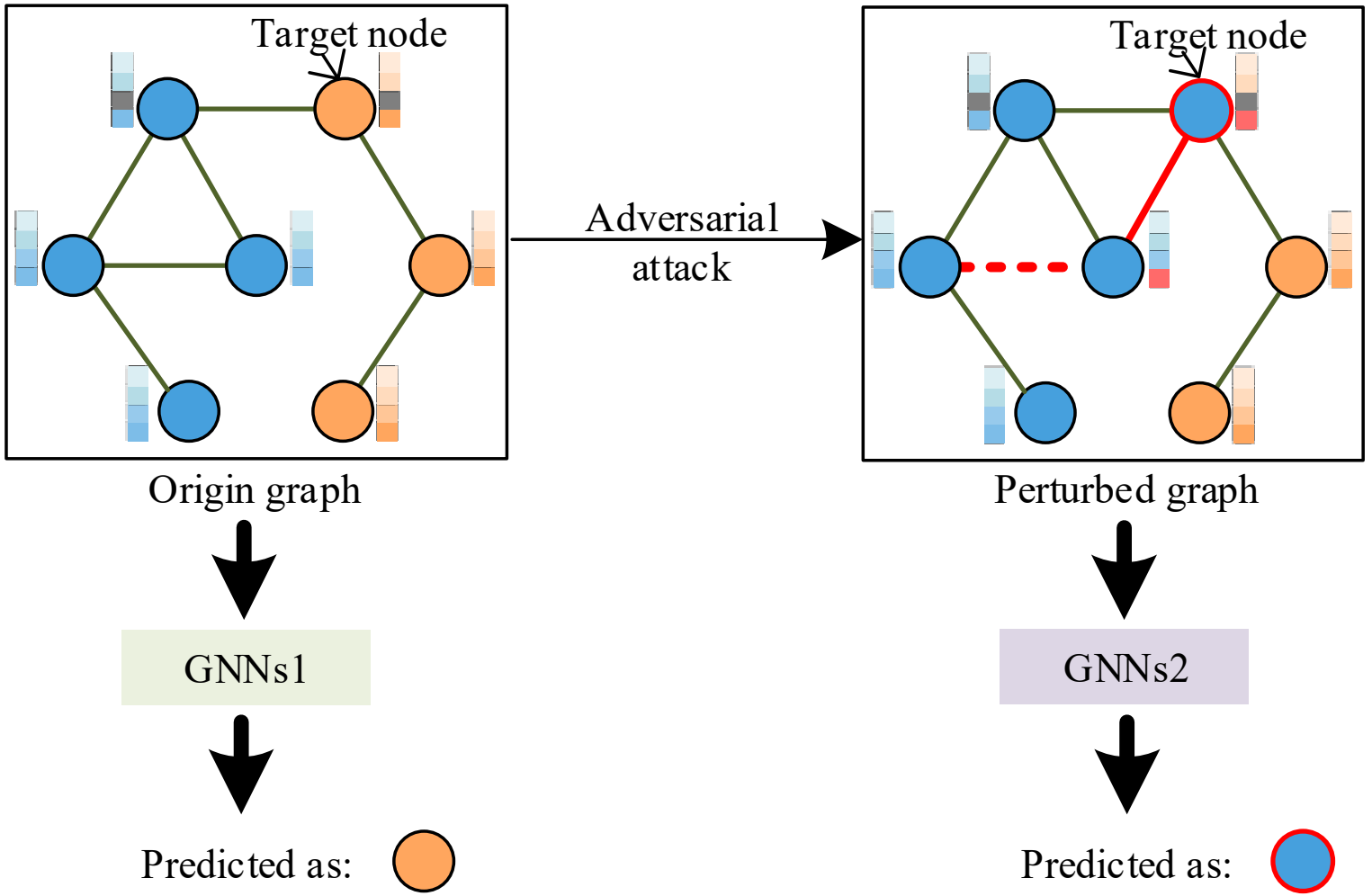
Kipf & Welling (NIPS BDL 2016)

"Graph Auto-Encoders"

Brief History of GNNs

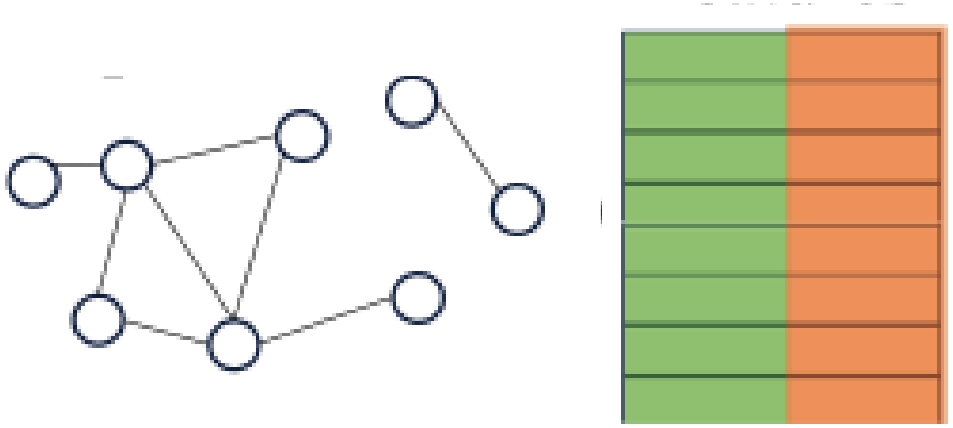


GNNs are Vulnerable

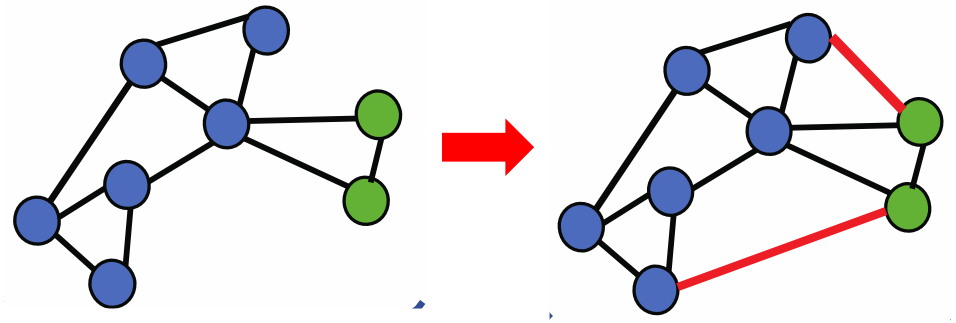


Challenges in Robust GNNs

- Complex Perturbation Space
  - Vast Searching Space



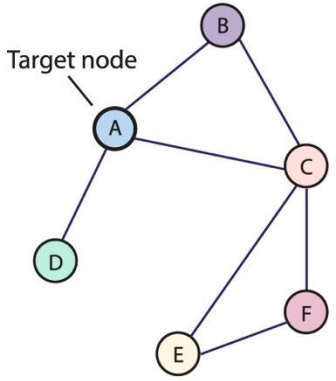
    - Graph structure
    - Node attributes
  - Discrete data domain



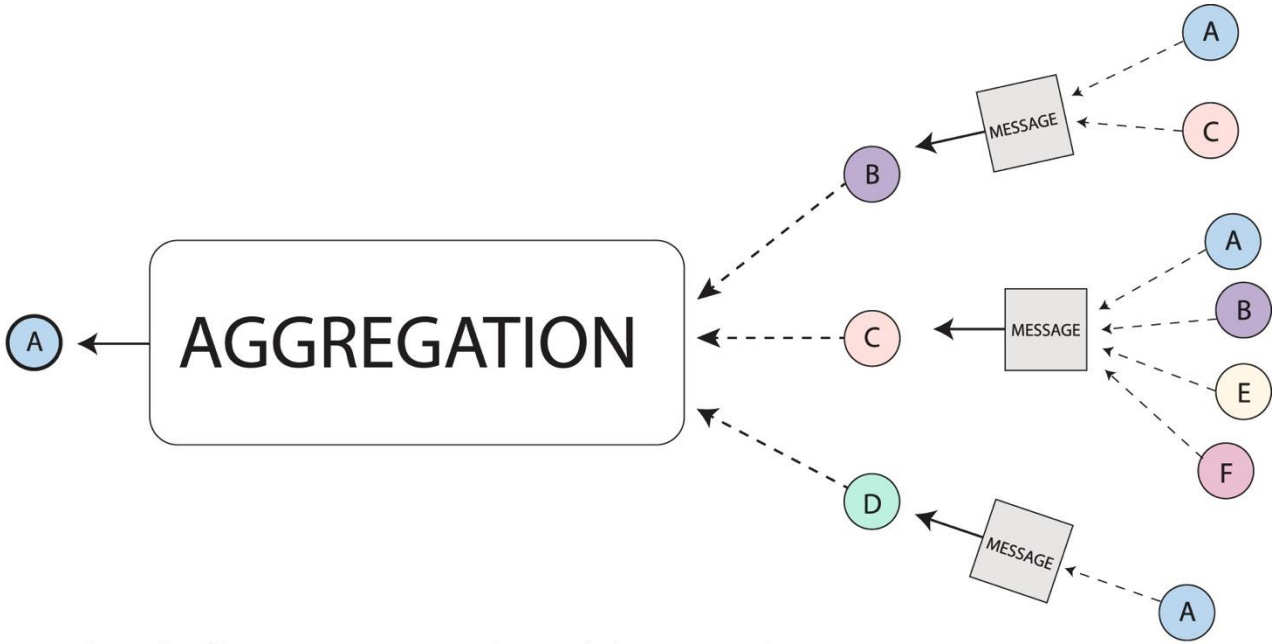
    - Adding or removing edges instead of mortifying pixels
    - Intractable discrete optimization problem

Challenges in Robust GNNs

- Non-IID Data
  - Interdependence between instances in graphs
  - Structural perturbations alter the message-passing mechanism, impacting representation propagation.
  - A change in one node can influence multiple other instances.



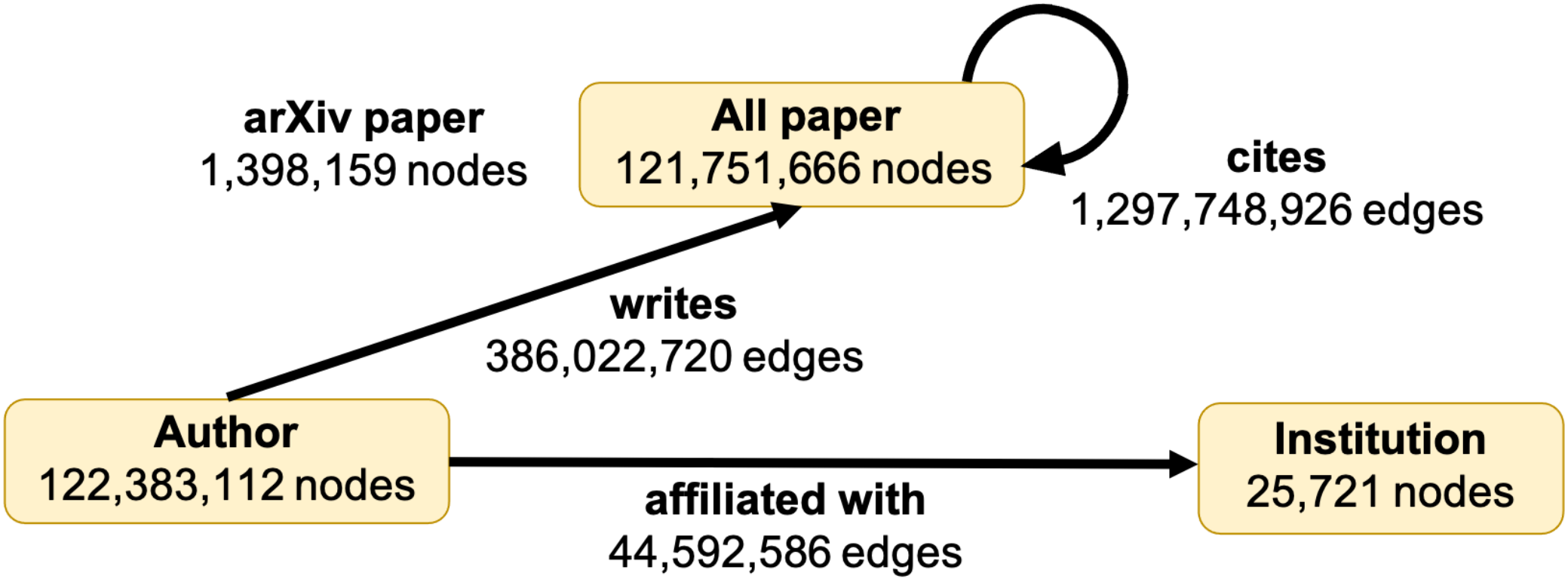
Input Graph



Example of how message aggregation works between nodes

Challenges in Robust GNNs

- Scalibility
  - Large-scale graph



https://ogb.stanford.edu/docs/lsc/mag240m/

Challenges in Robust GNNs







- Complex Perturbation Space
  - Vast searching space
    - Graph structure
    - Node attributes
  - Discrete data domain
    - Adding or removing edges instead of mortifying pixels
    - Intractable discrete optimization problem
- Interdependent Data
  - Interdependence between instances in graphs
  - Structural perturbations alter the message-passing mechanism, impacting representation propagation.
  - A change in one node can influence multiple other instances.
- Scalability
  - Large-scale graphs
- Unnoticeability
  - GNN models should remain robust to minor graph changes, while defining graph similarity is challenging

Tutorial Outline



- **Introduction** of in GNNs Robustness
- Adversarial **Attacks** on Graph
- Adversarial **Defenses** on Graph
- **Evaluation** of Robust GNNs
- **Summary** and Discussion

Overview

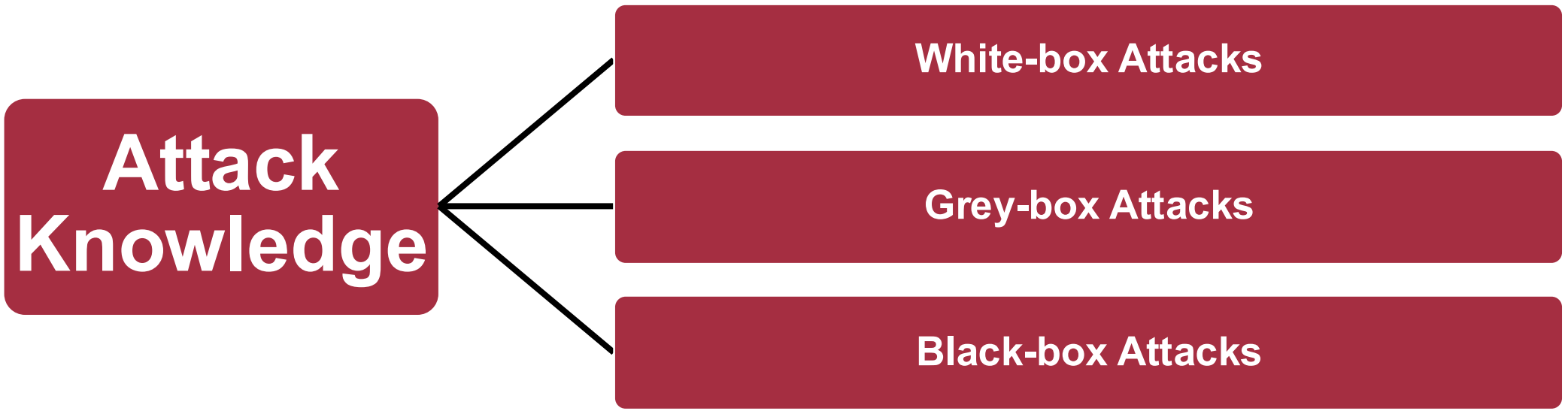
Presenter:
Minhua Lin

Threat Models

- Attack Knowledge
- Attack Phase
- Attack Mechanism
- Attack Target

Attack Knowledge

Attack knowledge is the knowledge the attackers need to possess about the dataset and the target model.



Sun, Lichao, et al. “Adversarial attack and defense on graph data: A survey.” *IEEE Transactions on Knowledge and Data Engineering,* 2022.

White-box Attacks: PGD

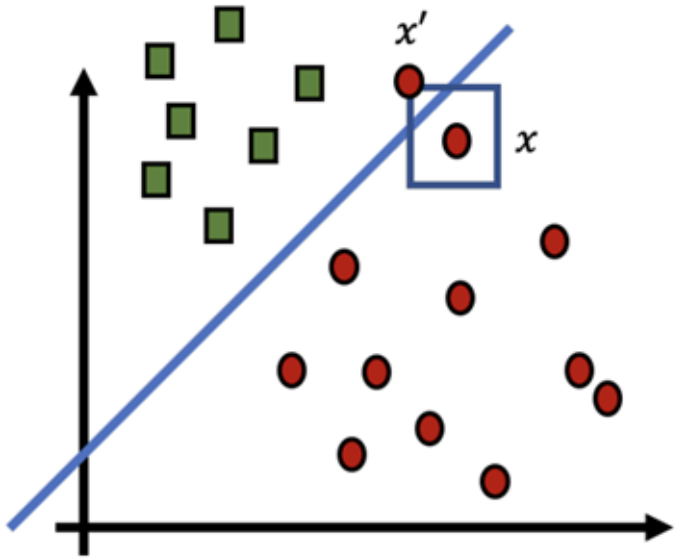
- **Attacker's knowledge:**
  - Suppose model F with parameter θ is given to attacker;

- **PGD Topology Attack:**

$$\underset{\mathbf{s}}{\text{minimize}} \quad \sum_{i \in \mathcal{V}} f_i(\mathbf{s}; \mathbf{W}, \mathbf{A}, \{\mathbf{x}_i\}, y_i) \qquad \text{Gradient-based}$$

$$\text{subject to} \quad \mathbf{1}^\top \mathbf{s} \leq \epsilon, \ \mathbf{s} \in \{0, 1\}^n, \qquad \text{Unnoticeable}$$

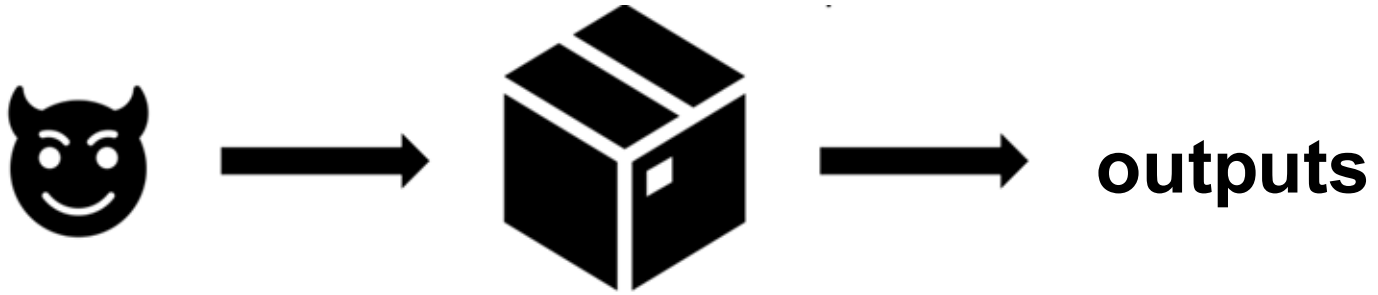
  - $\mathbf{s}$ is the vector form of perturbation



Xu, Kaidi, et al. "Adversarial attack and defense on graph data: A survey." IJCAI, 2019.

Black-box Attacks

- **Attacker's knowledge:**
  - Only input data and output response are available to the attacker;

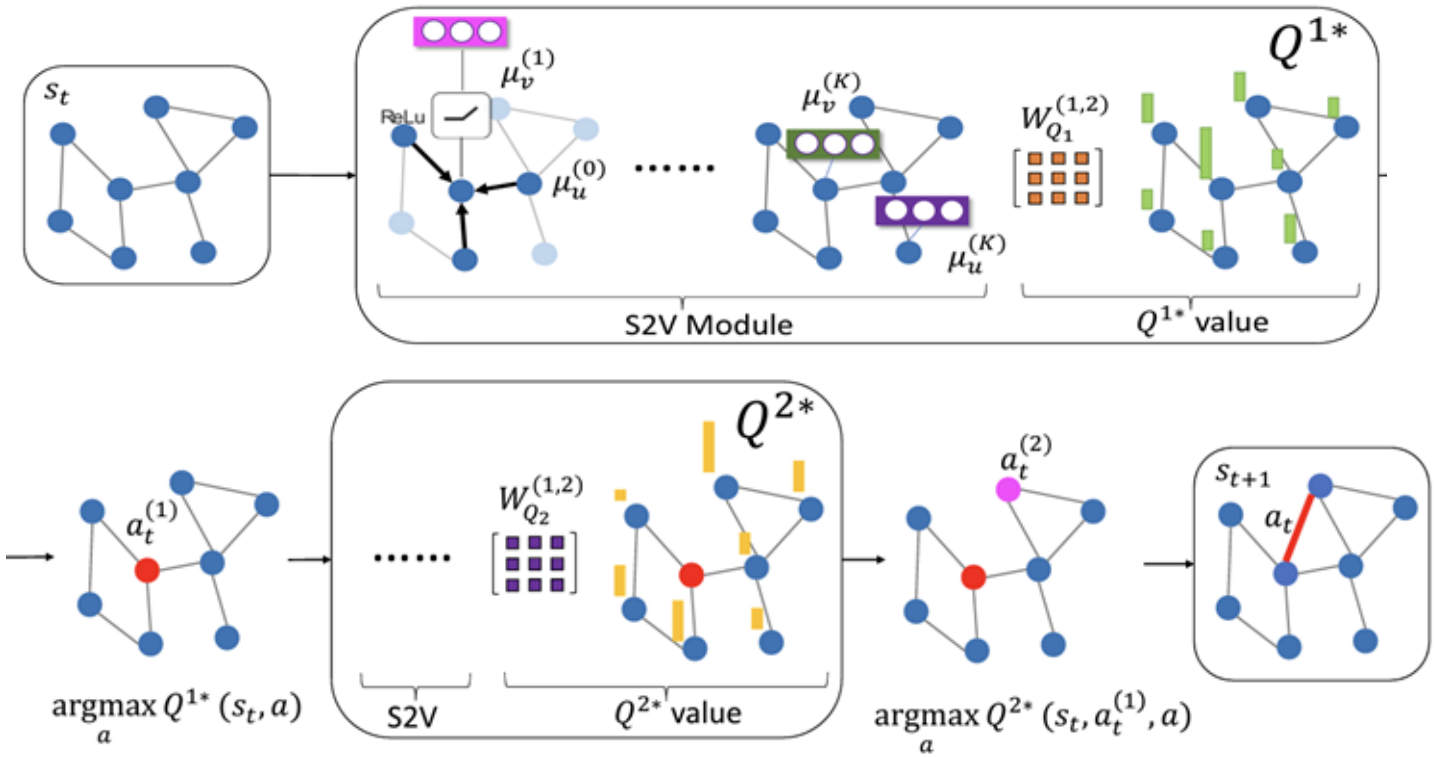


- Rely on queries and observations through the model APIs

Dai, Hanjun, et al. "Adversarial attack on graph structured data. In International conference on machine learning" ICML, 2018.

Black-box Attacks

- **RL-S2V: a reinforcement learning based black-box attacks**
  - Action: add/delete an edge
  - State: Current partially modified graph
  - Reward: If the attack is successful (positive/negative reward)

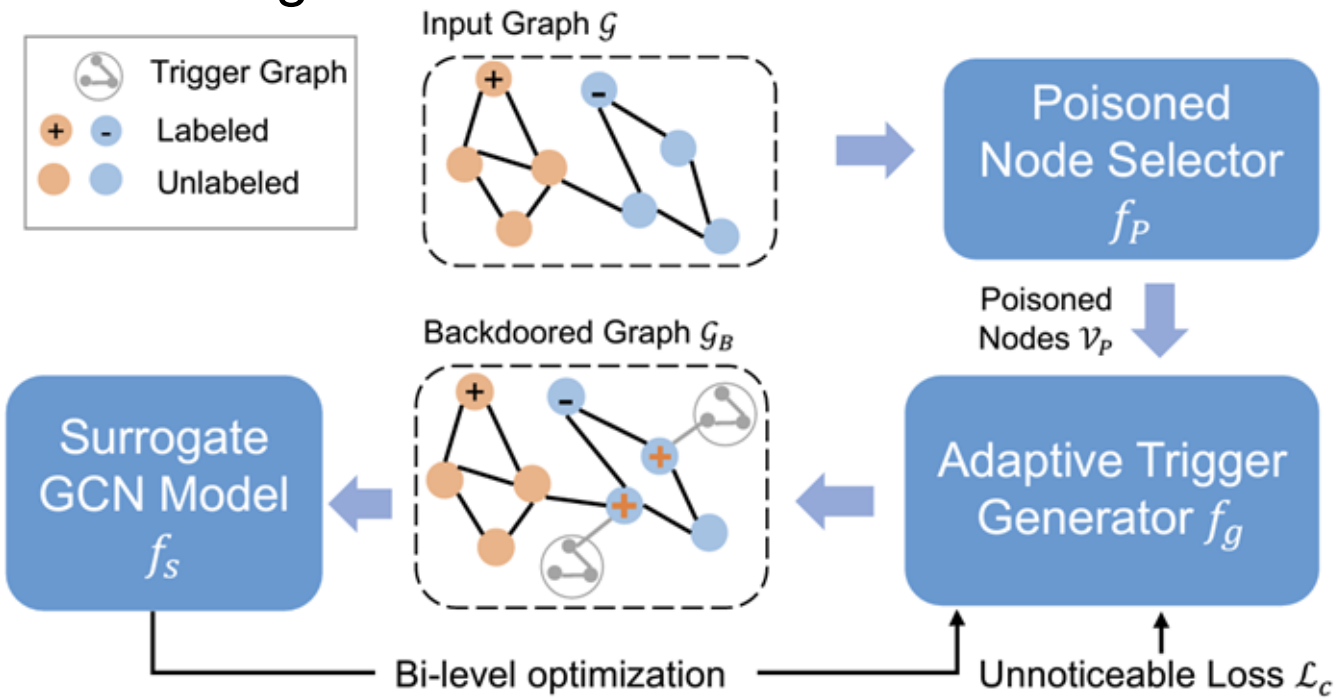


Dai, Hanjun, et al. “Adversarial attack on graph structured data. In International conference on machine learning” ICML, 2018.

Grey-box Attacks

- **Attacker's knowledge:**
  - The **model architecture** is known to the attackers, but the model's parameters remain inaccessible.

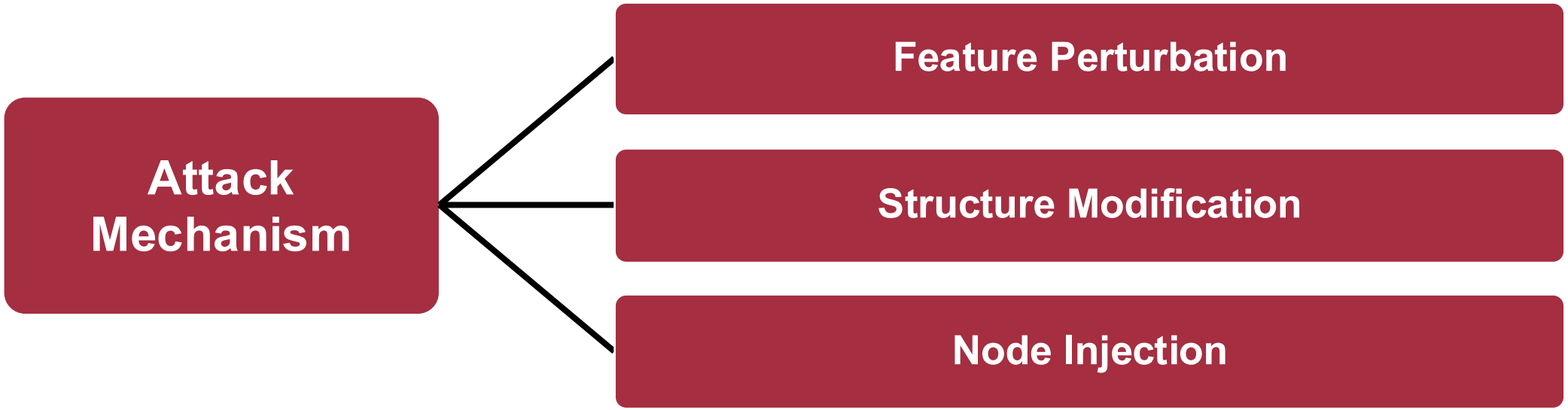
- The attacker usually train a surrogate model to imitate the target model and conduct white-box attack on the surrogate model.



Dai, Enyan, et al. "Unnoticeable Backdoor Attacks on Graph Neural Networks" WWW 2023.

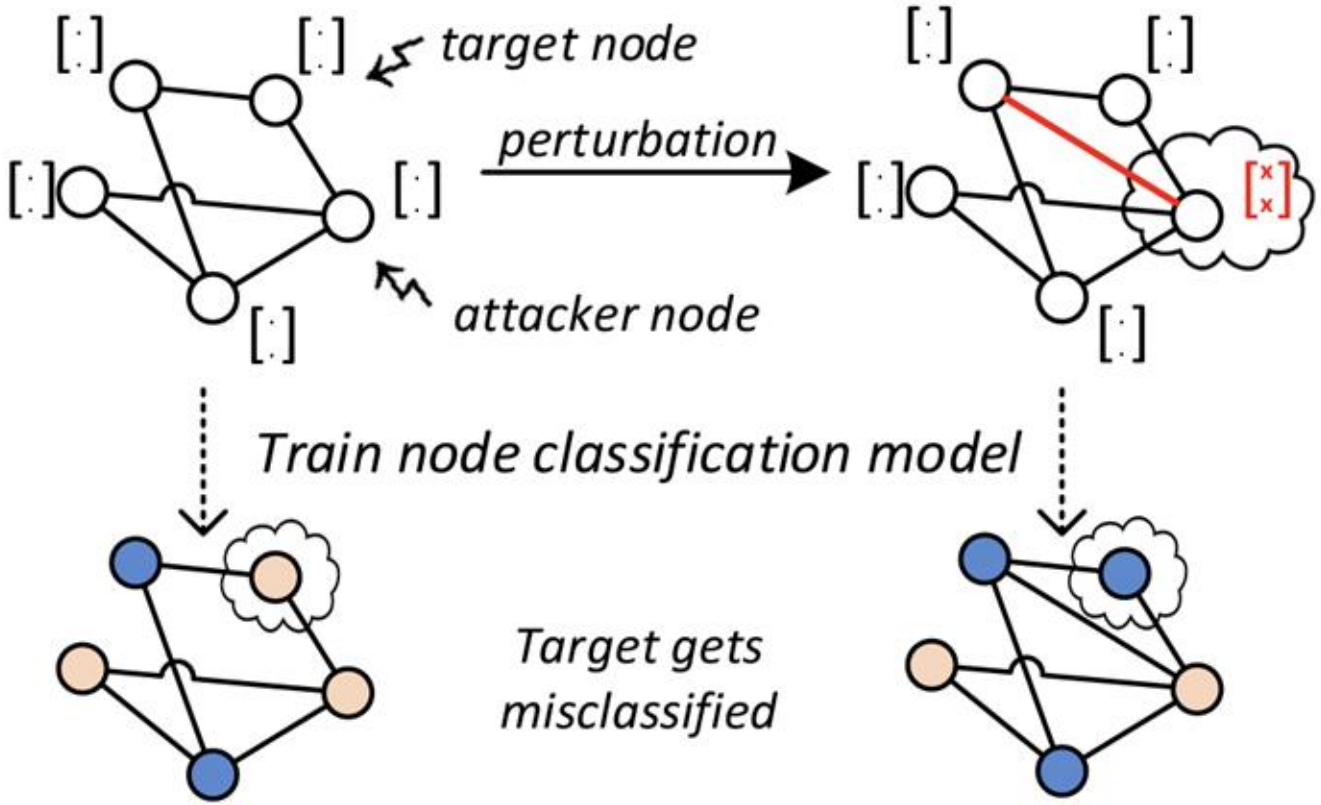
Attack Mechanism

Attention Mechanism: Attacker may perturb the node features, add/delete edges and inject fake nodes



Feature Perturbation

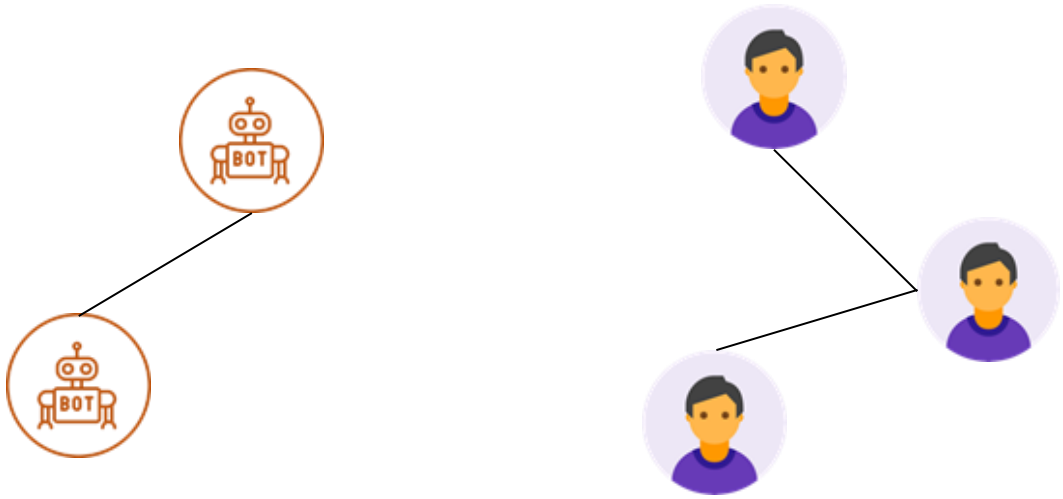
- **Attacker's capability:**
  - Aims to revise node features, which is similar to adversarial attack in vision domains



Zügner, Daniel, et al. "Adversarial Attacks on Neural Networks for Graph Data" KDD, 2018.

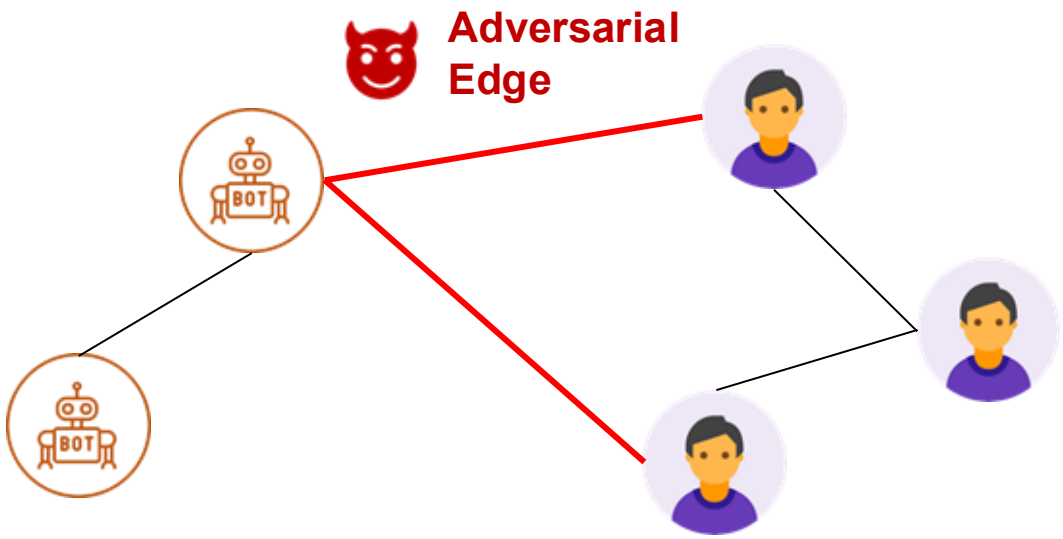
Structure Modification

- **Attacker's capability :**
  - Structure modification is more powerful than feature perturbation in graphs.
  - The attacker can add/delete edges in the original graph



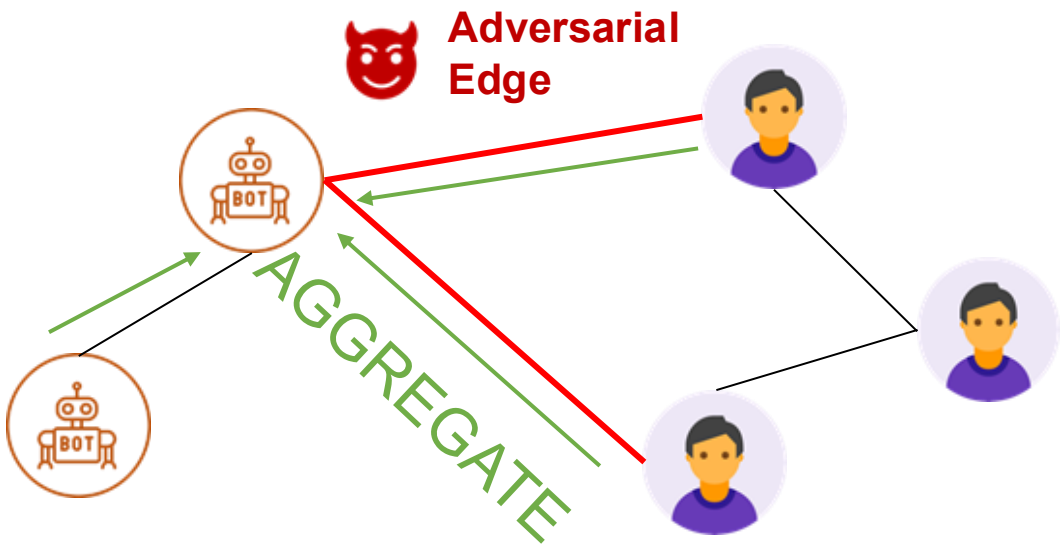
Structure Modification

- **Attacker's capability :**
  - Structure modification is more powerful than feature perturbation in graphs.
  - The attacker can add/delete edges in the original graph



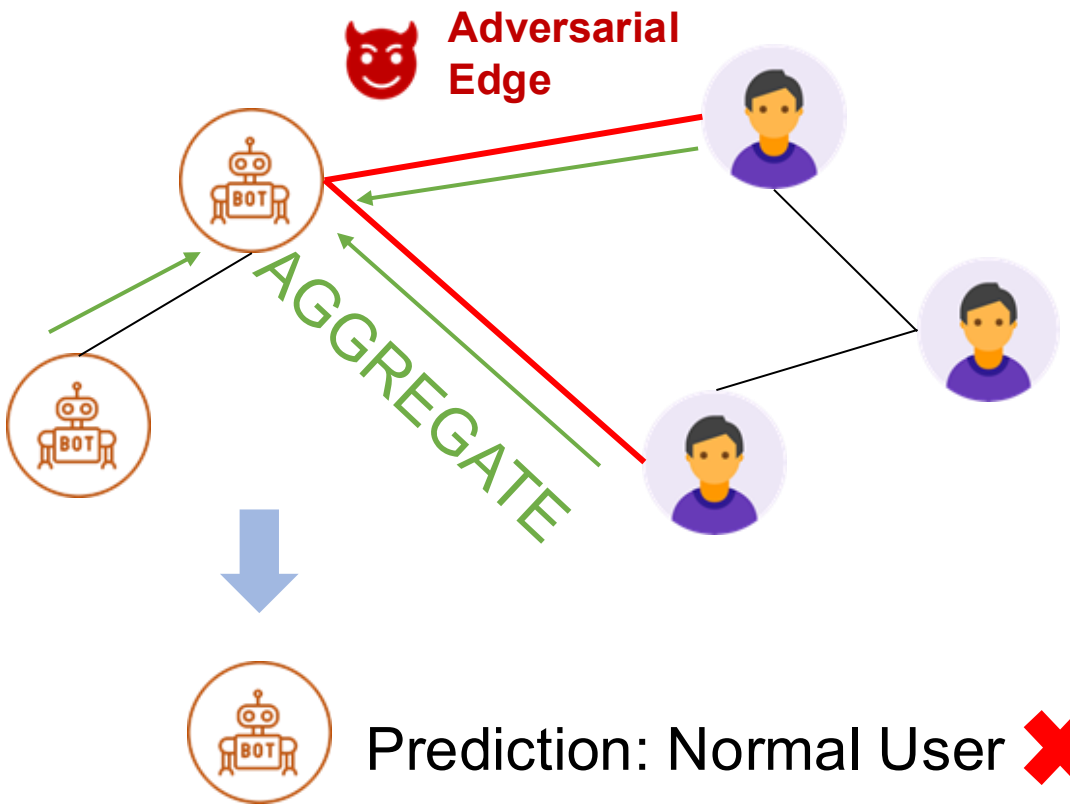
Structure Modification

- **Attacker's capability :**
  - Structure modification is more powerful than feature perturbation in graphs.
  - The attacker can add/delete edges in the original graph



Structure Modification

- **Attacker's capability :**
  - Structure modification is more powerful than feature perturbation in graphs.
  - The attacker can add/delete edges in the original graph



Noisy edge will affect the information aggregation

Nettack (Feature/Structure Perturbation)

- **Objective functions:**
  - GNN training objective: $$L(\theta; A, X) = -\sum_{v \in \mathcal{V}_L} \ln Z_{v,c_v} \quad , \quad Z = f_\theta(A, X)$$
  - Attack objective:

$$\underset{(A',X') \in \mathcal{P}^{G0}_{\Delta,\mathcal{A}}}{\arg\max} \; \max_{c \neq c_{old}} \ln Z^*_{v_0,c} - \ln Z^*_{v_0,c_{old}}$$

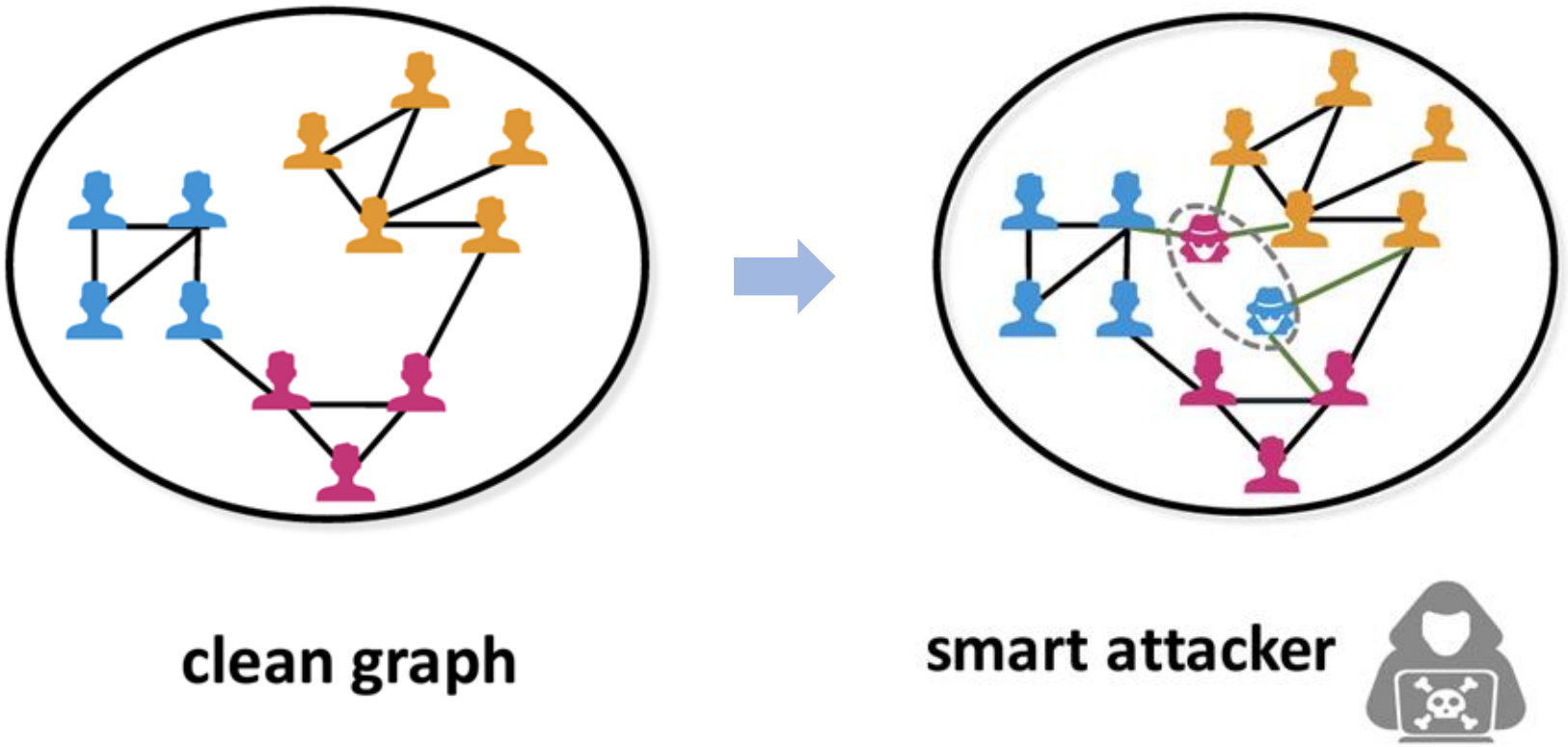
$$subject\ to\ Z^* = f_{\theta^*}(A', X')\ with\ \theta^* = \arg\min_\theta L(\theta; A', X')$$

Find a perturbed graph such that can classify target node as other label!

Zügner, Daniel, et al. “Adversarial Attacks on Neural Networks for Graph Data” KDD, 2018.

Node Injection

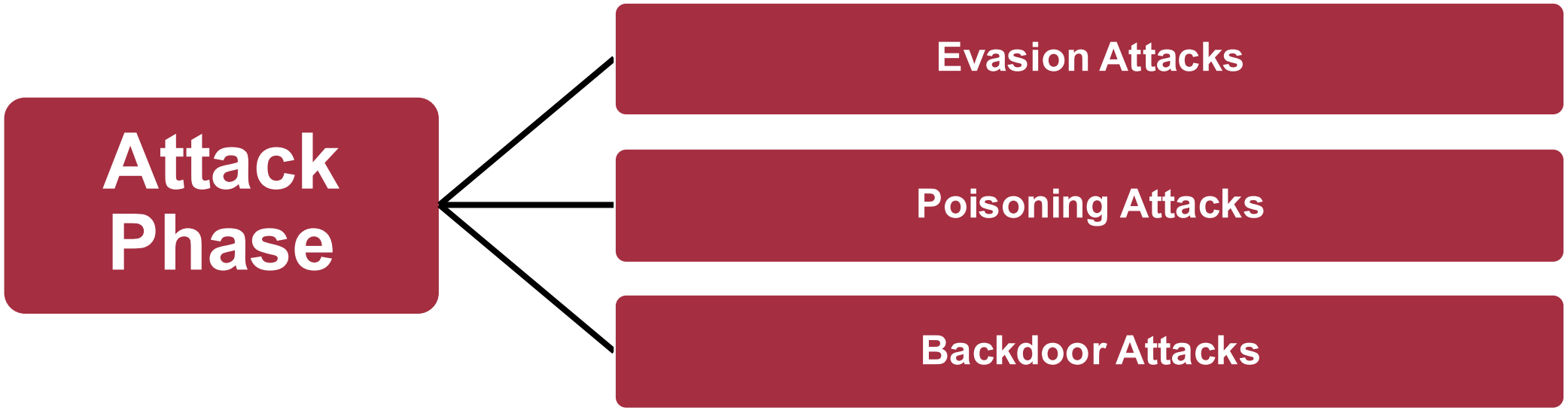
- **Attacker's capability :**
  - The attacker can inject fake nodes without modifying the original graph.



Sun, Yiwei, et al. "Adversarial attacks on graph neural networks via node injections: A hierarchical reinforcement learning approach" WWW, 2020.

Attack Phase

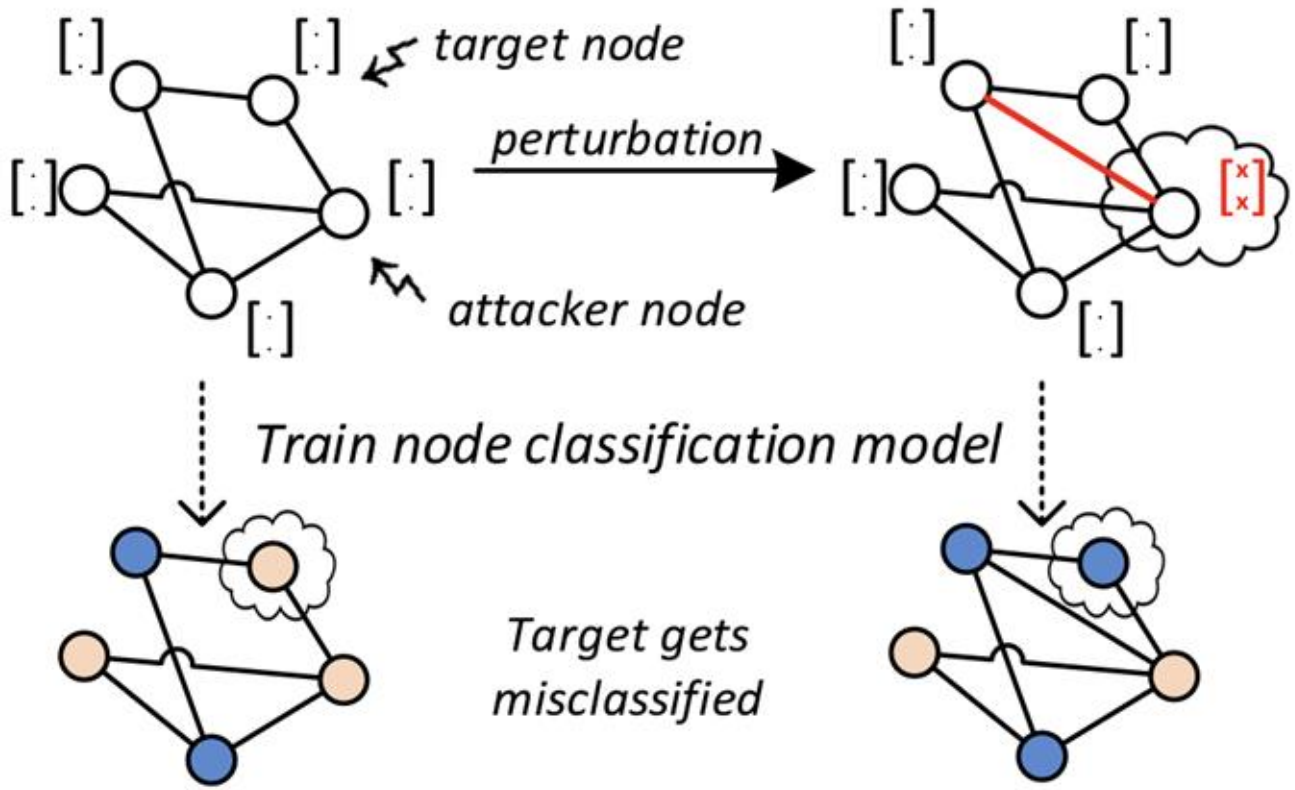
Attack Phase is the stage that the attackers add perturbations into samples.



Evasion Attacks

- **Attacker's goal:**
  - The attacker add perturbation to the test samples to mislead the model to misclassify the attacked sample. (post-training)

- **Representative Works**:
  - Nettack (Zügner et al, 2018)



Zügner, Daniel, et al. "Adversarial Attacks on Neural Networks for Graph Data" KDD, 2018.

Poisoning Attacks

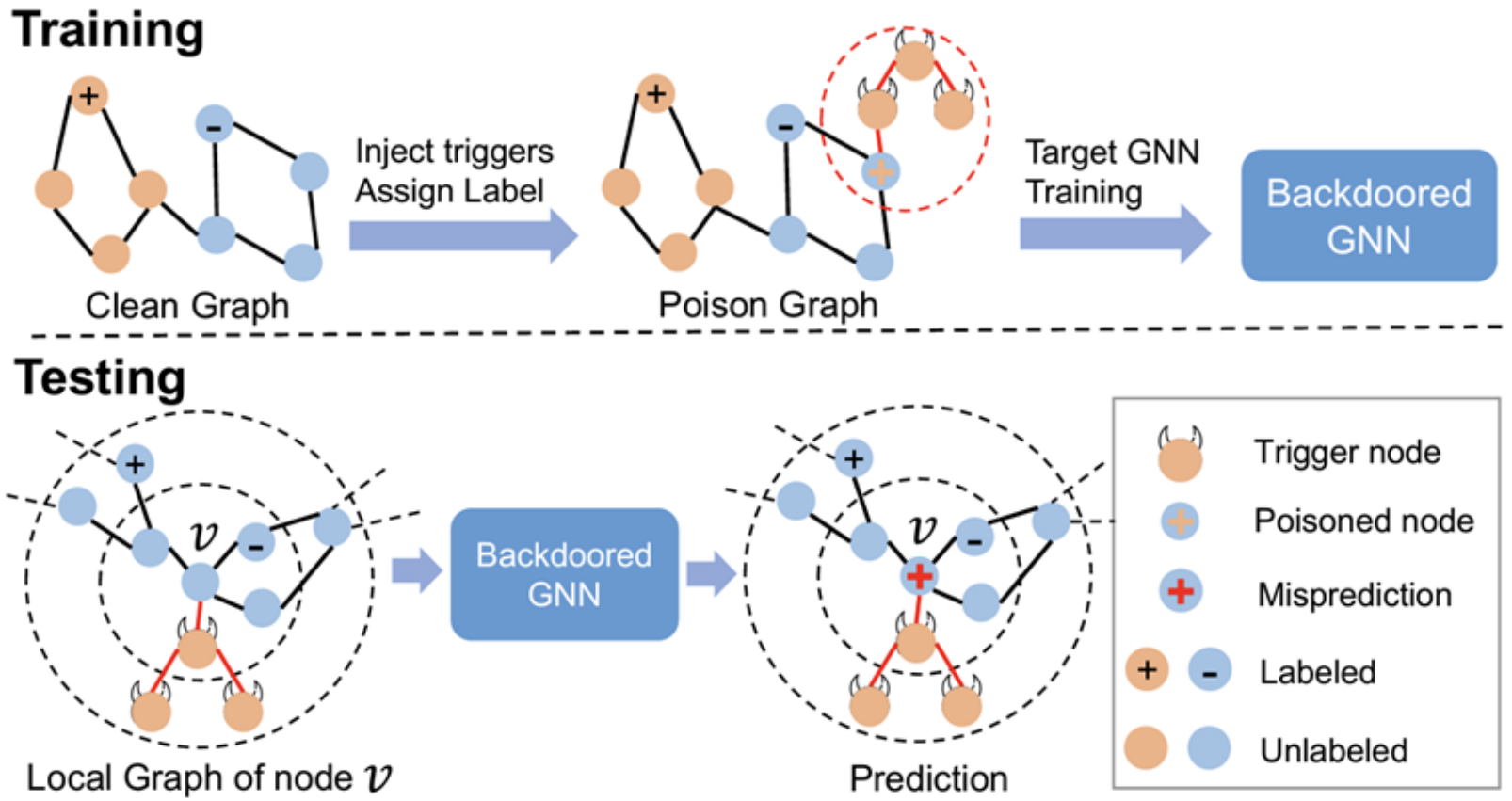
- **Attacker's goal:**
  - The attacker add perturbation to the training samples to poison the model parameters. (during training)
  - These attacks occur during the training phase, where the injected noise or manipulated data disrupts the model's learning process.

- **Representative Works**:
  - NIPA (Sun et al, 2020)
  - Mettack (Zügner et al, 2019)

Dai, Hanjun, et al. "Adversarial attack on graph structured data. In International conference on machine learning" ICML, 2018.

Backdoor Attacks

- **Attacker's goal:**
  - The attacker aims to manipulate the model into misclassifying specific target nodes while maintaining normal behavior for clean nodes.



Dai, Enyan, et al. "Unnoticeable Backdoor Attacks on Graph Neural Networks" WWW 2023.

Backdoor Attacks: UGBA

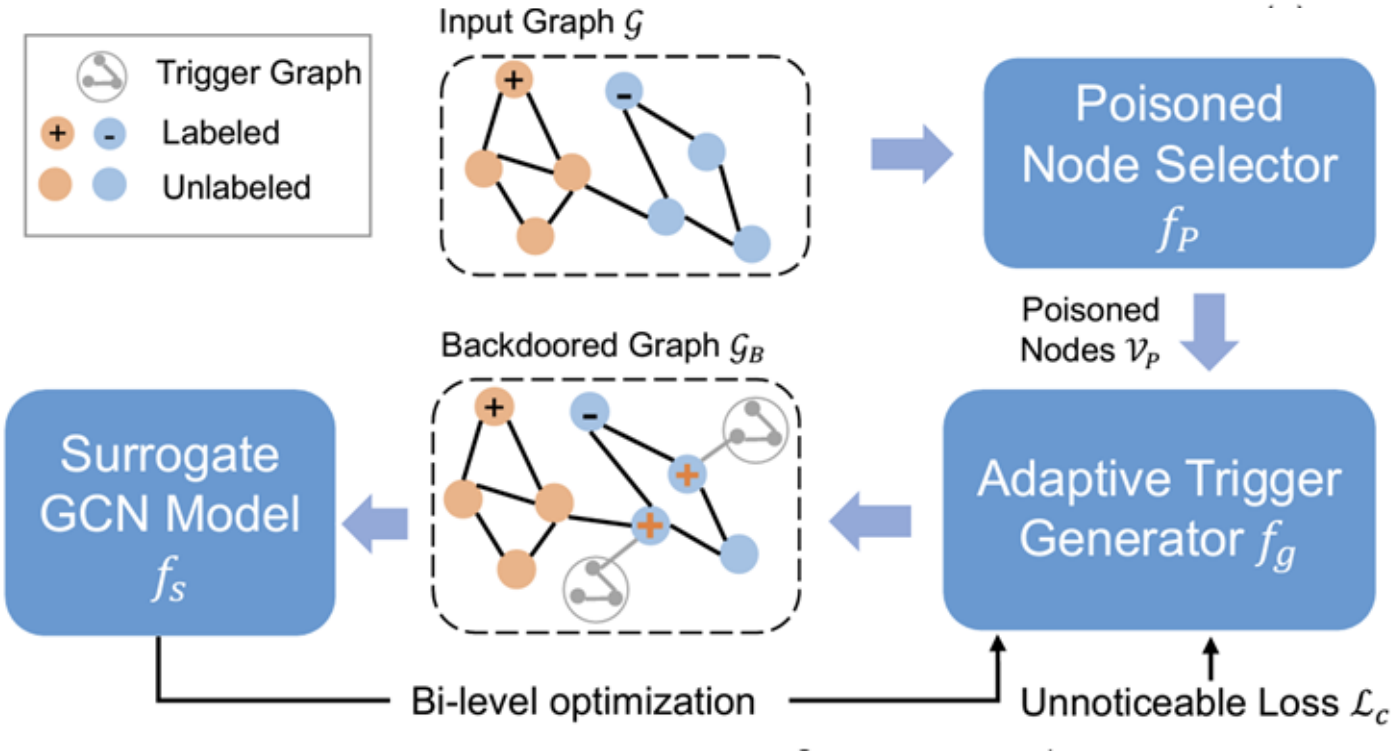
- **Objective:**

$$\min_{\mathcal{V}_P, \theta_g} \sum_{v_i \in \mathcal{V}_U} l(f_{\theta^*}(a(\mathcal{G}_C^i, g_i)), y_t)$$

$$s.t.\ \theta^* = \arg\min_{\theta} \sum_{v_i \in \mathcal{V}_L} l(f_\theta(\mathcal{G}_C^i), y_i) + \sum_{v_i \in \mathcal{V}_P} l(f_\theta(a(\mathcal{G}_C^i, g_i)), y_t),$$

$$\forall v_i \in \mathcal{V}_P \cup \mathcal{V}_T,\ g_i \ meets\ Eq.(1)\ and\ |g_i| < \Delta_g$$

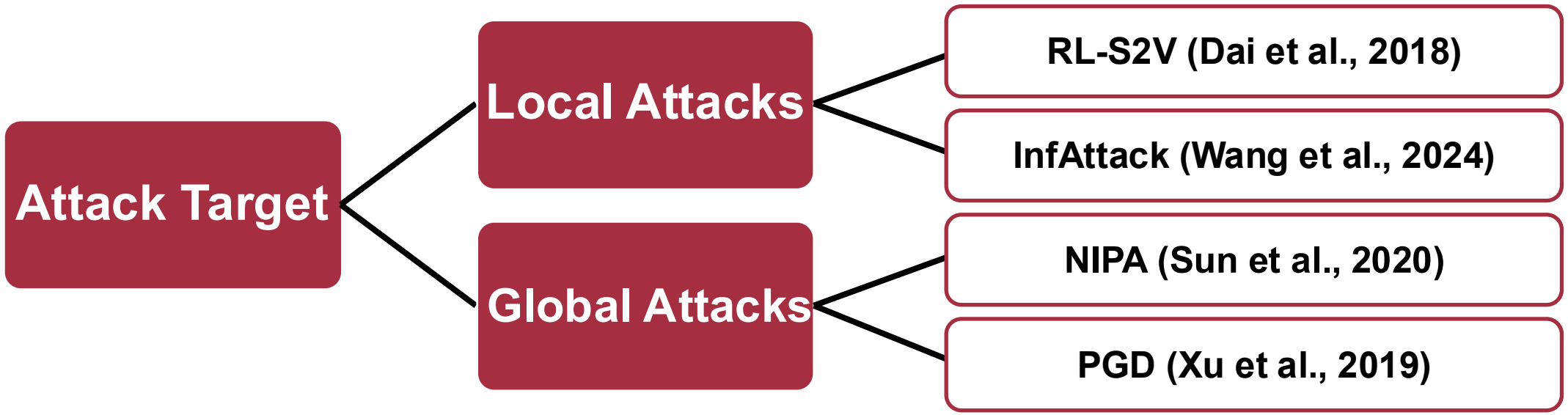
$$|\mathcal{V}_P| \leq \Delta_P$$



Dai, Enyan, et al. "Unnoticeable Backdoor Attacks on Graph Neural Networks" WWW 2023.

Attack Target

- **Attack target** can be categorized as local attacks and global attacks based on whether attackers aim to misclassify a set of target instance or reduce the overall performance.



Sun, Lichao, et al. "Adversarial attack and defense on graph data: A survey." *IEEE Transactions on Knowledge and Data Engineering,* 2022.

Overview

Presenter:
Zhichao Hou

Traditional Approaches

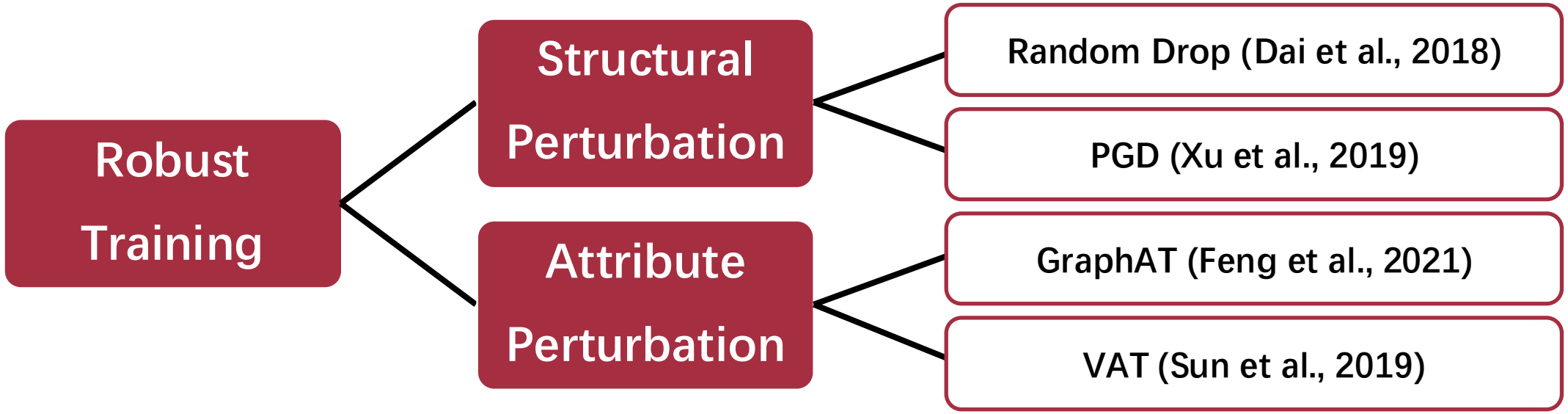
- Robust Training
- Graph Cleaning
- Attention Mechanism

Emerging Techniques

- Implicit Architectures
- Innovative Perspective

Robust Training

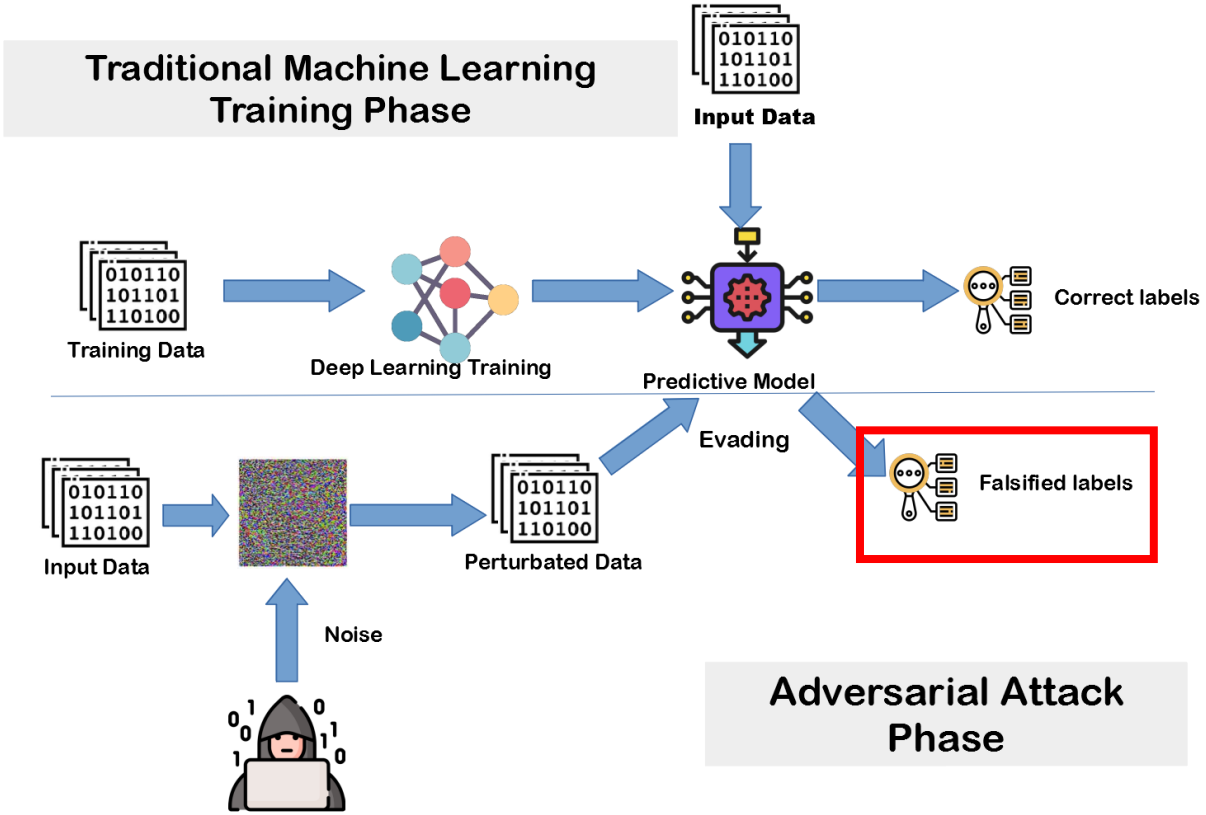
- **Robust Training** is a classical method to improve model robustness by augmenting the training data with either adversarial features or perturbed graph topology.



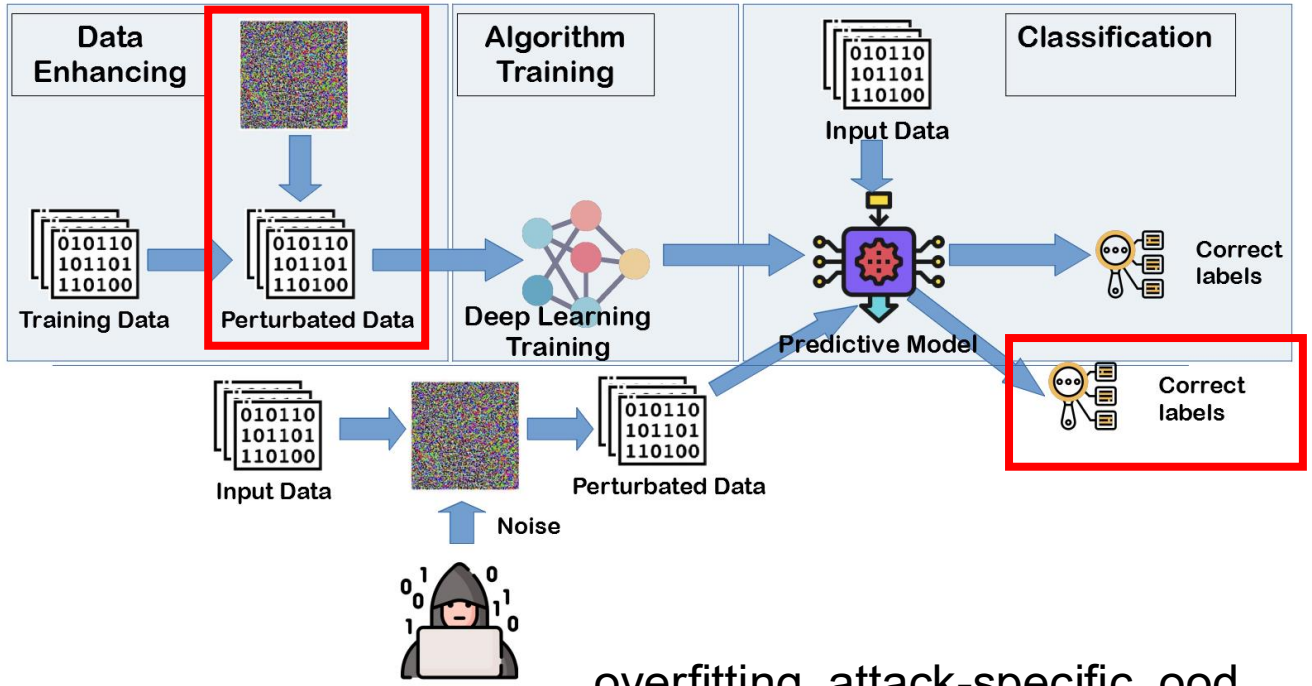
Sun, Lichao, et al. "Adversarial attack and defense on graph data: A survey." *IEEE Transactions on Knowledge and Data Engineering*, 2022.

Robust Training

- **Robust Training** is a classical method to improve model robustness by augmenting the training data with either adversarial features or perturbed graph topology.



Normal Training



overfitting, attack-specific, ood

Robust/Adversarial Training

Sun, Lichao, et al. "Adversarial attack and defense on graph data: A survey." *IEEE Transactions on Knowledge and Data Engineering*, 2022.

PGD-AT

- **PGD Topology Attack:**

$$\begin{aligned} \underset{\mathbf{s}}{\text{minimize}} \quad & \sum_{i \in \mathcal{V}} f_i(\mathbf{s}; \mathbf{W}, \mathbf{A}, \{\mathbf{x}_i\}, y_i) \\ \text{subject to} \quad & \mathbf{1}^\top \mathbf{s} \leq \epsilon, \ \mathbf{s} \in \{0,1\}^n, \end{aligned}$$

discrete

where:

- $\mathbf{S}$ is the Boolean perturbation matrix
- Perturbed graph: $\mathbf{A}' = \mathbf{A} + \mathbf{C} \circ \mathbf{S}, \ \mathbf{C} = \bar{\mathbf{A}} - \mathbf{A}$
- $\mathbf{s}$ is the vector form of $\mathbf{S}$
- f_i is cross-entropy (CE) or Carlili-Wagner (CW) loss

- Challenge: discrete variable → combinatorial optimization problem

K. Xu, H. Chen, S. Liu, P.-Y. Chen, T.-W. Weng, M. Hong, and X. Lin. Topology attack and defense for graph neural networks: An optimization perspective. arXiv preprint, 2019.

PGD-AT

PGD Topology Attack:

$$\begin{aligned} \underset{\mathbf{s}}{\text{minimize}} \quad & \textstyle\sum_{i\in\mathcal{V}} f_i(\mathbf{s}; \mathbf{W}, \mathbf{A}, \{\mathbf{x}_i\}, y_i) \\ \text{subject to} \quad & \mathbf{1}^\top \mathbf{s} \le \epsilon, \ \mathbf{s} \in \{0,1\}^n, \end{aligned}$$ discrete

$$\begin{aligned} \underset{\mathbf{s}}{\text{minimize}} \quad & f(\mathbf{s}) := \textstyle\sum_{i\in\mathcal{V}} f_i(\mathbf{s}; \mathbf{W}, \mathbf{A}, \{\mathbf{x}_i\}, y_i) \\ \text{subject to} \quad & \mathbf{s} \in \mathcal{S}, \end{aligned}$$

$$\mathcal{S} = \{\mathbf{s} \,|\, \mathbf{1}^T s \le \epsilon, \mathbf{s} \in [0,1]^n\}$$

continuous

Step 1: Projected Gradient Descent (PGD)

$$\mathbf{s}^{(t)} = \Pi_{\mathcal{S}}\left[\mathbf{s}^{(t-1)} - \eta_t \hat{\mathbf{g}}_t\right] \quad \hat{\mathbf{g}}_t = \nabla f(\mathbf{s}^{(t-1)})$$

$$\Pi_{\mathcal{S}}(\mathbf{a}) := \arg\min_{\mathbf{s}\in\mathcal{S}} \|\mathbf{s} - \mathbf{a}\|_2^2$$

Step 2: Random Sampling

Algorithm 1 Random sampling from probabilistic to binary topology perturbation

1: Input: probabilistic vector $\mathbf{s}$, K is # of random trials
2: **for** $k = 1, 2, \ldots, K$ **do**
3: draw binary vector $\mathbf{u}^{(k)}$ following

$$u_i^{(k)} = \begin{cases} 1 & \text{with probability } s_i \\ 0 & \text{with probability } 1 - s_i \end{cases}, \forall i \qquad (9)$$

4: **end for**
5: choose a vector $\mathbf{s}^*$ from $\{\mathbf{u}^{(k)}\}$ which yields the smallest attack loss $f(\mathbf{u}^{(k)})$ under $\mathbf{1}^T\mathbf{s} \le \epsilon$.

K. Xu, H. Chen, S. Liu, P.-Y. Chen, T.-W. Weng, M. Hong, and X. Lin. Topology attack and defense for graph neural networks: An optimization perspective. arXiv preprint, 2019.

PGD-AT

- **PGD Topology Attack:**

$$\begin{aligned} \underset{\mathbf{s}}{\text{minimize}} \quad & f(\mathbf{s}) := \textstyle\sum_{i \in \mathcal{V}} f_i(\mathbf{s}; \mathbf{W}, \mathbf{A}, \{\mathbf{x}_i\}, y_i) \\ \text{subject to} \quad & \mathbf{s} \in \mathcal{S}, \end{aligned}$$

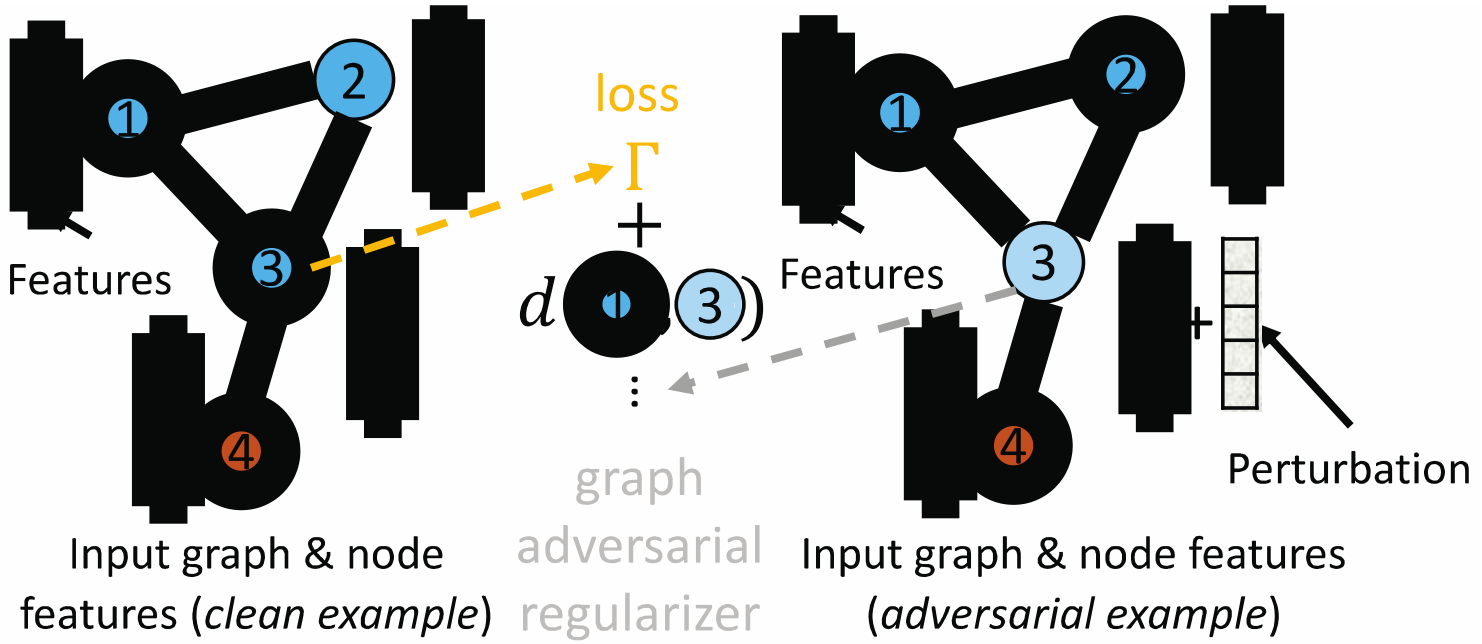
$$\mathcal{S} = \{\mathbf{s} \,|\, \mathbf{1}^T s \leq \epsilon, \mathbf{s} \in [0,1]^n\}$$

- **Adversarial Training for GNNs (PGD-AT):**

$$\underset{\mathbf{W}}{\text{minimize}} \; \underset{\mathbf{s} \in \mathcal{S}}{\text{maximize}} \; -f(\mathbf{s}, \mathbf{W})$$

K. Xu, H. Chen, S. Liu, P.-Y. Chen, T.-W. Weng, M. Hong, and X. Lin. Topology attack and defense for graph neural networks: An optimization perspective. arXiv preprint, 2019.

GraphAT



$$\textbf{min: } \Gamma_{GAT} = \Gamma + \beta \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} d(f(\boldsymbol{x}_i + \boldsymbol{r}_i^g, G|\boldsymbol{\Theta}), f(\boldsymbol{x}_j, G|\boldsymbol{\Theta})),$$

$$\textbf{max: } \boldsymbol{r}_i^g = \arg \max_{\boldsymbol{r}_i, \|\boldsymbol{r}_i\| \leq \epsilon} \sum_{j \in \mathcal{N}_i} d(f(\boldsymbol{x}_i + \boldsymbol{r}_i, G|\hat{\boldsymbol{\Theta}}), f(\boldsymbol{x}_j, G|\hat{\boldsymbol{\Theta}})),$$

$$\Gamma = \Omega + \lambda \Phi$$

Ω classification loss
Φ graph Laplacian term

F. Feng, X. He, J. Tang, and T.-S. Chua. Graph adversarial training: Dynamically regularizing based on graph structure. IEEE Transactions on Knowledge and Data Engineering, 2019.

GraphAT & GraphVAT

GraphAT:

$$\textbf{min: } \Gamma_{GAT} = \Gamma + \beta \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} d(f(\boldsymbol{x}_i + \boldsymbol{r}_i^g, G|\boldsymbol{\Theta}), f(\boldsymbol{x}_j, G|\boldsymbol{\Theta})),$$

$$\textbf{max: } \boldsymbol{r}_i^g = \arg \max_{\boldsymbol{r}_i, \|\boldsymbol{r}_i\| \leq \epsilon} \sum_{j \in \mathcal{N}_i} d(f(\boldsymbol{x}_i + \boldsymbol{r}_i, G|\hat{\boldsymbol{\Theta}}), f(\boldsymbol{x}_j, G|\hat{\boldsymbol{\Theta}})),$$

Linear approximation:

$$\boldsymbol{r}_i^g \approx \epsilon \frac{\boldsymbol{g}}{\|\boldsymbol{g}\|}, \text{ where } \boldsymbol{g} = \nabla_{\boldsymbol{x}_i} \sum_{j \in \mathcal{N}_i} D(f(\boldsymbol{x}_i, G|\hat{\boldsymbol{\Theta}}), f(\boldsymbol{x}_j, G|\hat{\boldsymbol{\Theta}}))$$

GraphVAT:

$$\textbf{min: } \Gamma_{GVAT} = \Gamma + \underbrace{\alpha \sum_{i=1}^{N} d(f(x_i + r_i^v, G|\Theta), \tilde{y}_i)}_{\text{virtual adversarial regularizer}} + \underbrace{\beta \sum_{i=1}^{N} \sum_{j \in N_i} d(f(x_i + r_i^g, G|\Theta), f(x_j, G|\Theta))}_{\text{graph adversarial regularizer}},$$

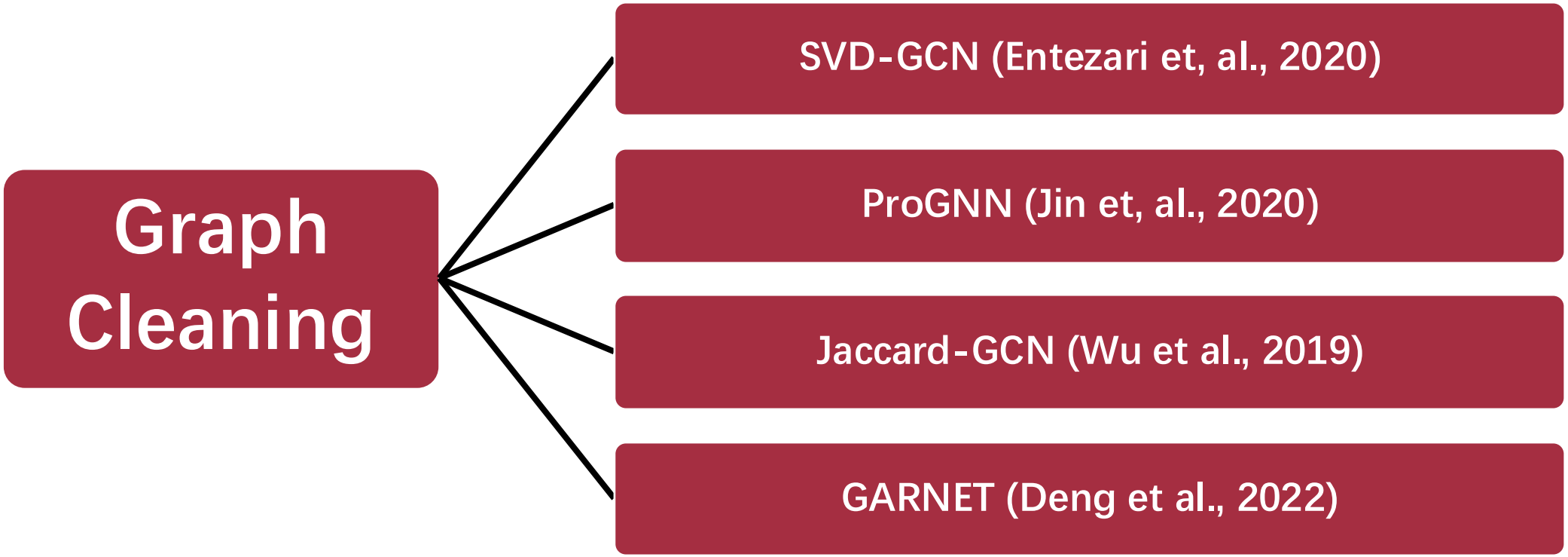
$$\textbf{max: } r_i^v = \arg \max_{r_i', \|r_i'\| \leq \epsilon'} d(f(x_i + r_i', G|\hat{\Theta}), \tilde{y}_i),$$

$$\tilde{y}_i = \begin{cases} \hat{y}_i, & i \leq M \text{ (labeled node)}, \\ f(x_i, G|\hat{\Theta}), & M < i \leq N \text{ (unlabeled node)}. \end{cases}$$

F. Feng, X. He, J. Tang, and T.-S. Chua. Graph adversarial training: Dynamically regularizing based on graph structure. IEEE Transactions on Knowledge and Data Engineering, 2019.

Graph Cleaning

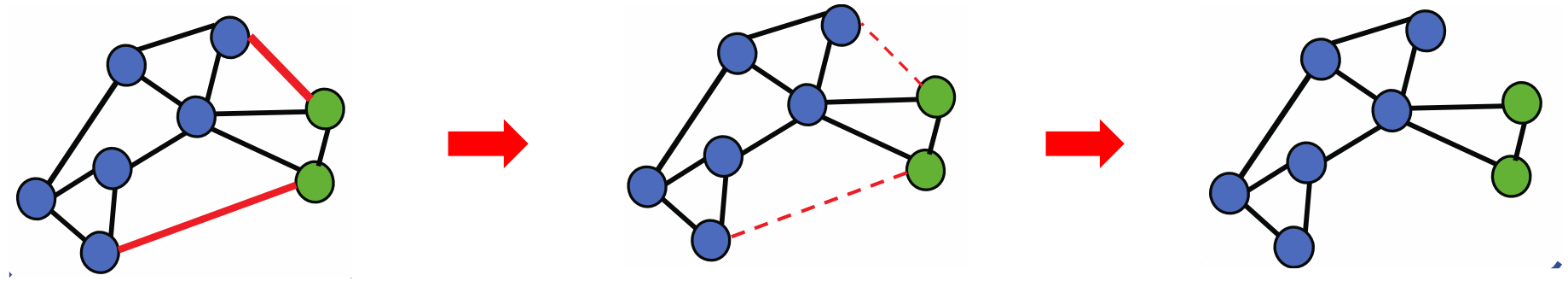
Graph Cleaning aims to recover the clean graph topology from an adversarially manipulated graph using techniques such as pruning or low-rank approximation to mitigate the impact of malicious edges.



Sun, Lichao, et al. "Adversarial attack and defense on graph data: A survey." *IEEE Transactions on Knowledge and Data Engineering,* 2022.

Graph Cleaning

Graph Cleaning aims to recover the clean graph topology from an adversarially manipulated graph using techniques such as pruning or low-rank approximation to mitigate the impact of malicious edges.



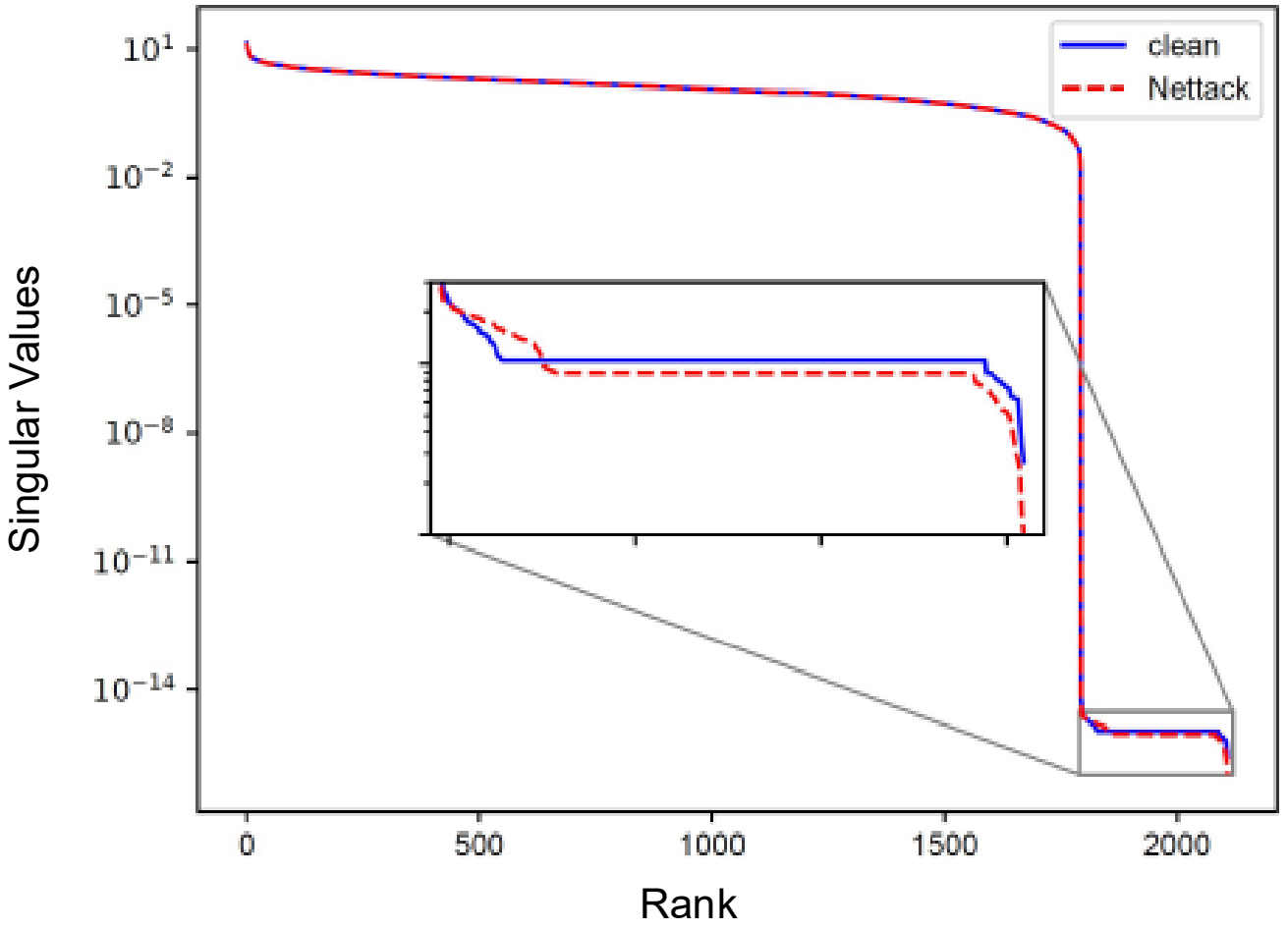
Attacked Graph

Graph Cleaning

Cleaned Graph

Sun, Lichao, et al. "Adversarial attack and defense on graph data: A survey." *IEEE Transactions on Knowledge and Data Engineering,* 2022.

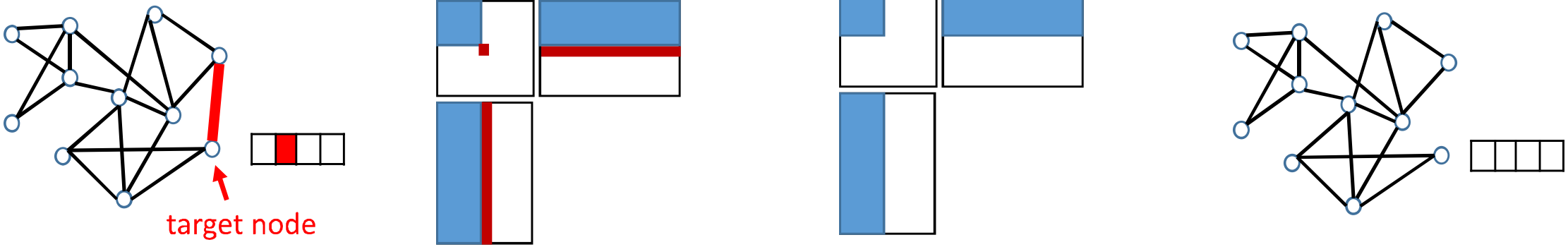
SVD-GCN



Only **high-rank (low-valued)** singular components of the graph are affected.

N. Entezari, S. A. Al-Sayouri, A. Darvishzadeh, and E. E. Papalexakis. All you need is low (rank) defending against adversarial attacks on graphs. WSDM, 2020.

SVD-GCN



Adjacency matrix: $A = U\Sigma V^T$

Rank-r approximation of A: $A_r = U_r \Sigma_r V_r^T = \sum_{i=1}^{r} u_i \sigma_i v_i^T$

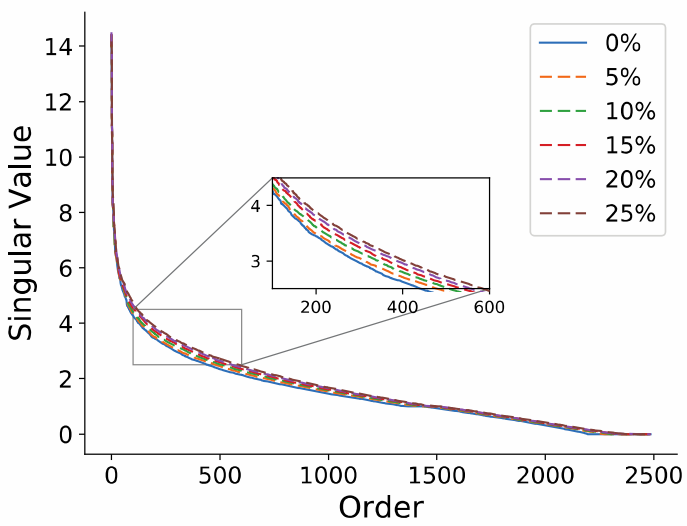
Low-rank approximation could be not differentiable

N. Entezari, S. A. Al-Sayouri, A. Darvishzadeh, and E. E. Papalexakis. All you need is low (rank) defending against adversarial attacks on graphs. WSDM, 2020.

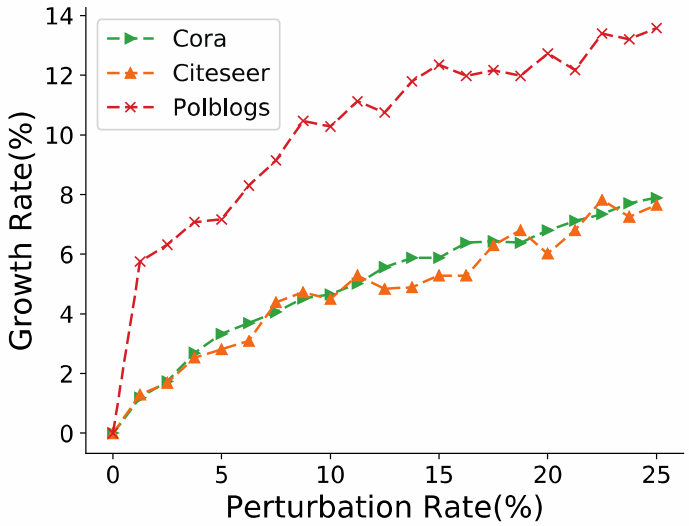
ProGNN

- The properties of real-world clean graphs:
  - Low-rank
  - Sparse
  - The features of two adjacent nodes tend to similar

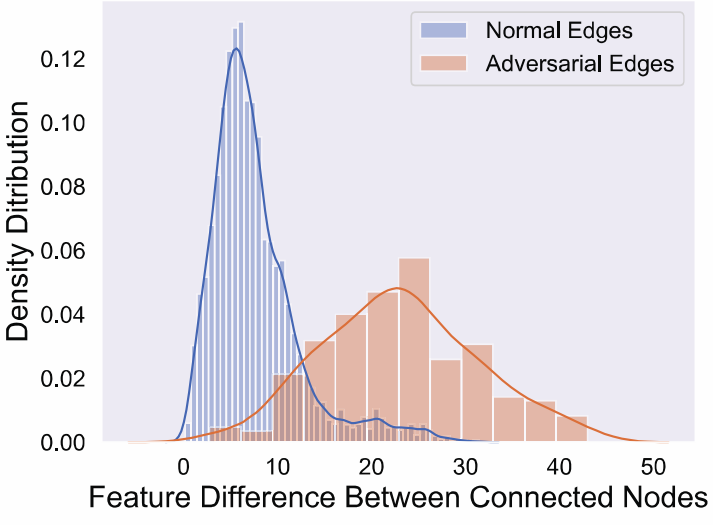
However, adversarial attacks are likely to violate these graph properties.



Singular Values



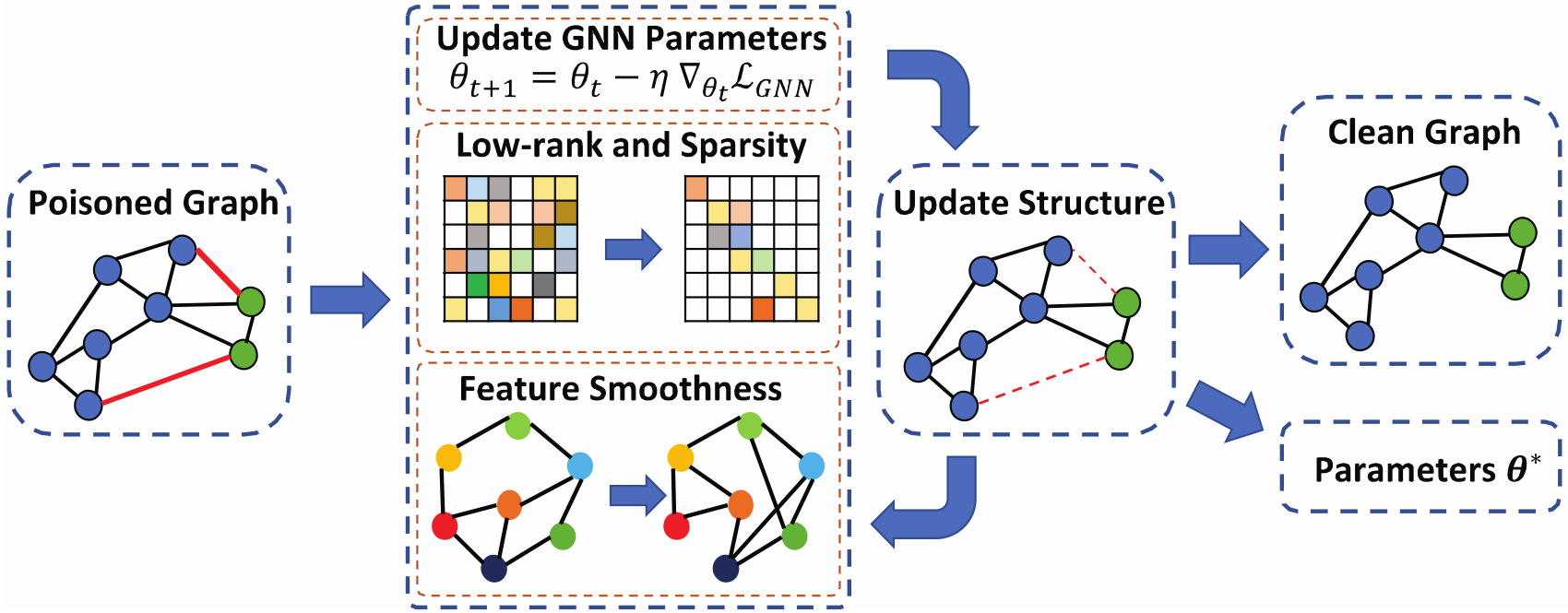
Rank Growth



Feature Smoothness

W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang. Graph structure learning for robust graph neural networks. KDD, 2020.

ProGNN



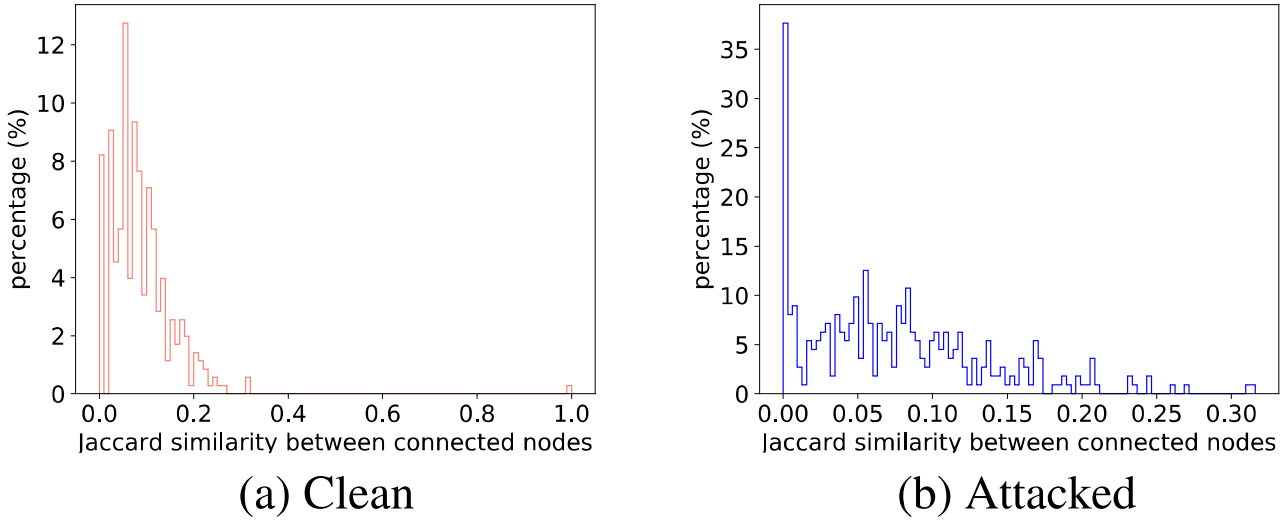
$$\arg\min_{\mathrm{S}\in\mathcal{S},\theta} \mathcal{L} = \mathcal{L}_0 + \lambda\mathcal{L}_s + \gamma\mathcal{L}_{GNN}$$

$$= \|\mathbf{A} - \mathbf{S}\|_F^2 + \alpha\|\mathbf{S}\|_1 + \beta\|\mathbf{S}\|_* + \gamma\mathcal{L}_{GNN}(\theta, \mathbf{S}, \mathbf{X}, \mathcal{Y}_L) + \lambda tr(\mathbf{X}^T\hat{\mathbf{L}}\mathbf{X}) \quad s.t., \mathbf{S} = \mathbf{S}^\top$$

Unnoticeble Sparse Low-rank Supervised Loss Feature smoothing

W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang. Graph structure learning for robust graph neural networks. KDD, 2020.

Jaccard-GCN



(a) Clean (b) Attacked

Figure 1: Histograms for the Jaccard similarities between connected nodes before and after FGSM attack.

Jaccard Similarity:

$$J_{u,v} = \frac{M_{11}}{M_{01} + M_{10} + M_{11}}$$

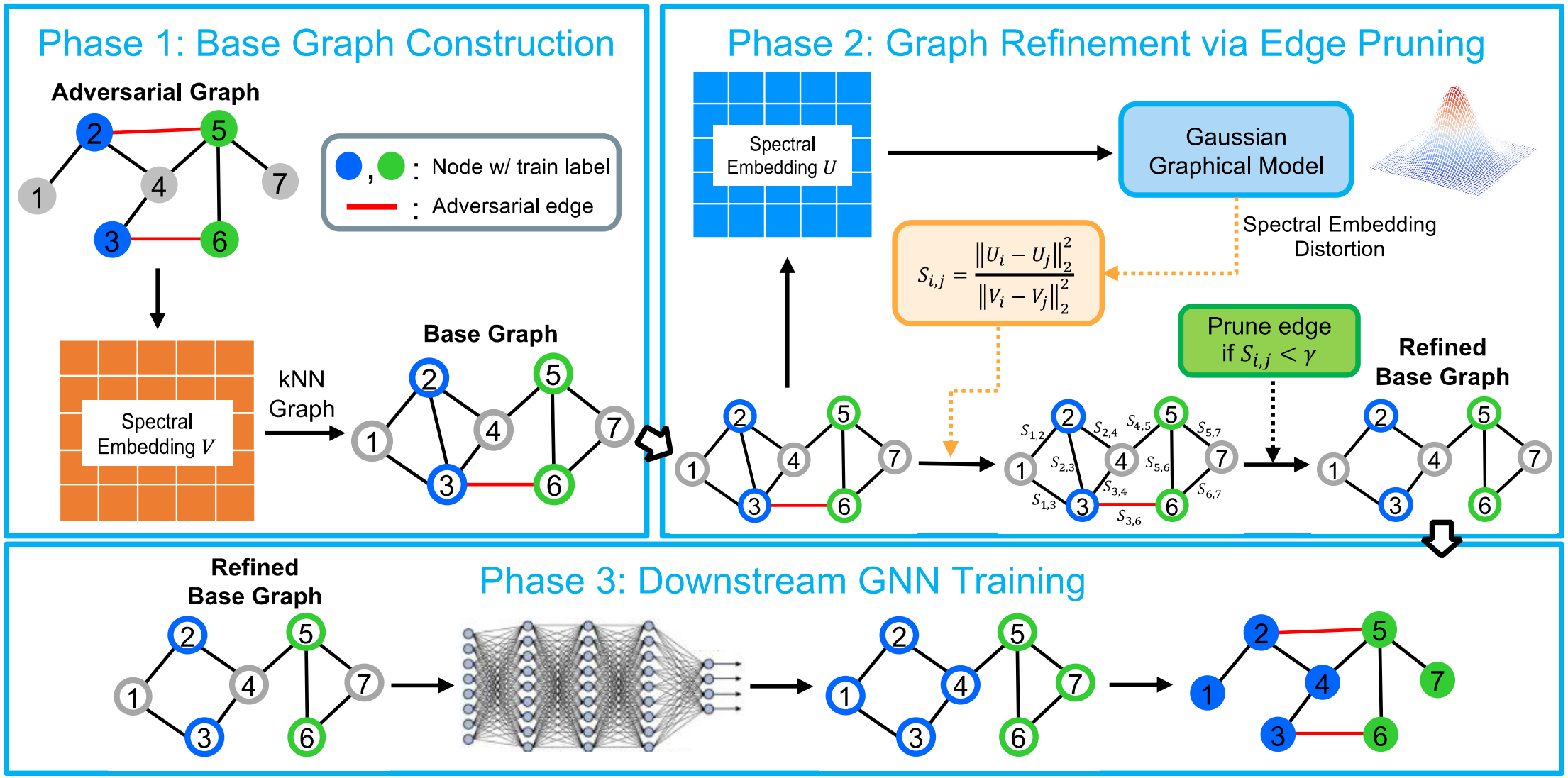
M_{10} represents the number of features where node u has a value of 1, while node v has a value of 0.

The same applies for M_{11}, M_{01}, M_{00}.

Jaccard-GCN assigns lower weights to the edges connecting nodes sharing little feature similarity.

H. Wu, C. Wang, Y. Tyshetskiy, A. Docherty, K. Lu, and L. Zhu. Adversarial examples on graph data: Deep insights into attack and defense. arXiv preprint, 2019.

GARNET



C. Deng, X. Li, Z. Feng, and Z. Zhang. Garnet: Reduced-rank topology learning for robust and scalable graph neural networks. LoG, 2022.

GARNET

- Probabilistic Graphical Models (PGM): to learn the **precision matirx Θ** that corresponds to an undirected graph structure $\mathcal{G}$ for encoding the **conditional dependence between variables**.
  - This work focuses on the attractive Gaussian Markov random fields (GMRFs), which restricts the **precision matrix** to be a Laplacian-like matrix:

$$\Theta = L + \frac{I}{\sigma^2}, \text{where } L = D - A \rightarrow \Theta_{ij} = -w_{ij} \text{ (off diagonal)}$$

  - To estimate a sparse Θ through maximum likelihood estimation (MLE):

$$\max_{\Theta} F := \log\det\Theta - \frac{1}{r} tr(VV^T\Theta) - \alpha\|\Theta\|_1$$

    Where $V \in \mathbb{R}^{n\times r}$ is the selected spectral embedding corresponding to r largest singular values.
  - Then the update of Θ_{ij} is:

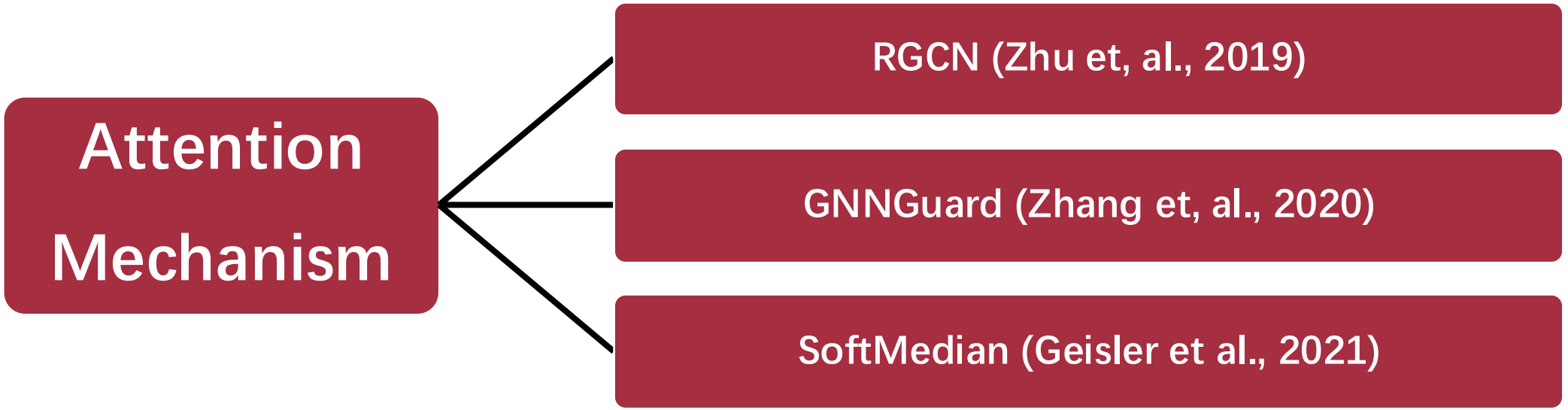
$$\Theta_{ij} \leftarrow \Theta_{ij} + \eta\frac{\partial F}{\partial \Theta_{ij}} = \Theta_{ij} - \eta\frac{\partial F}{\partial w_{ij}}$$

- **GARNET**: $\frac{\partial F}{\partial w_{ij}}$ large and positive → Θ_{ij} more negative → increase the edge weight of (i, j)

C. Deng, X. Li, Z. Feng, and Z. Zhang. Garnet: Reduced-rank topology learning for robust and scalable graph neural networks. LoG, 2022.

Attention Mechanism

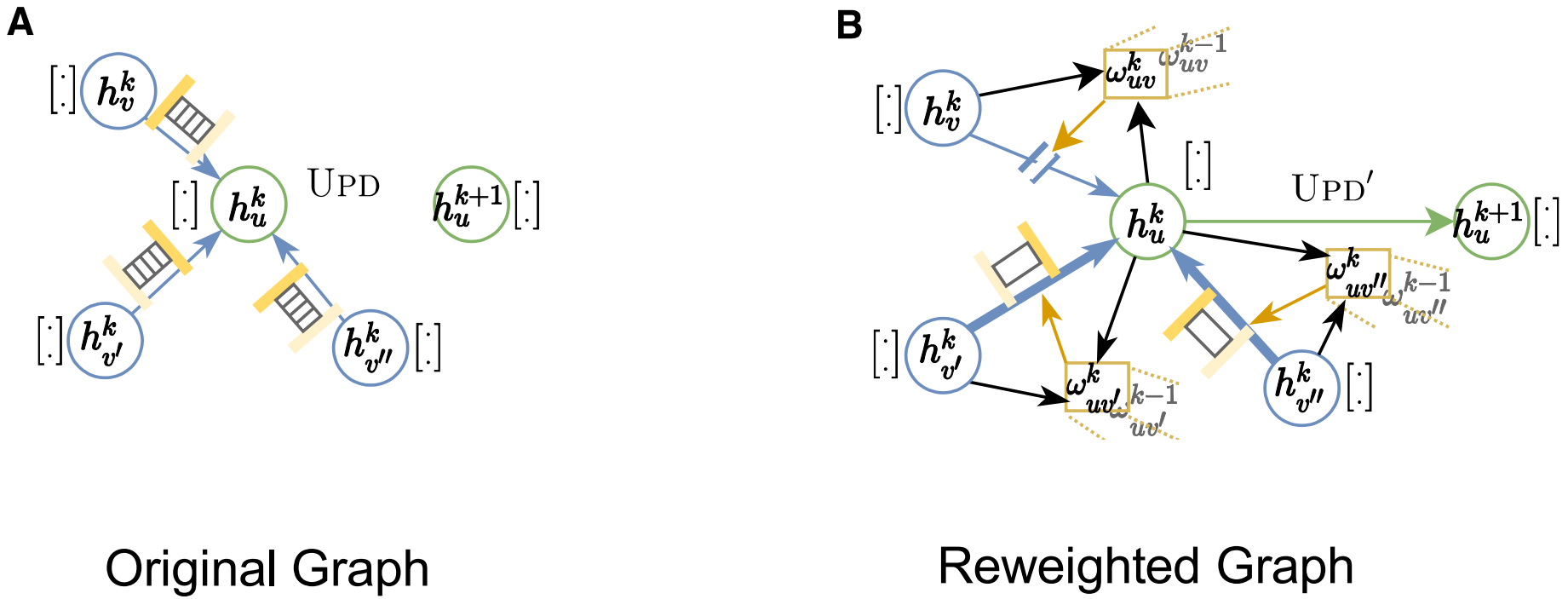
Attention Mechanism seeks to adaptively learn an attention scheme by up-weighting the important edges and down-weighting the suspicious ones.



Sun, Lichao, et al. "Adversarial attack and defense on graph data: A survey." *IEEE Transactions on Knowledge and Data Engineering,* 2022.

Attention Mechanism

Attention Mechanism seeks to adaptively learn an attention scheme by up-weighting the important edges and down-weighting the suspicious ones.



Sun, Lichao, et al. "Adversarial attack and defense on graph data: A survey." *IEEE Transactions on Knowledge and Data Engineering*, 2022.

RGCN

- Gaussian-based Graph Convolution Layer:
  - Instead of vectors, RGCN adopts Gaussian distributions as the hidden representations

$$\mathbf{h}_i^{(l)} = \mathcal{N}(\boldsymbol{\mu}_i^{(l)}, diag(\boldsymbol{\sigma}_i^{(l)}))$$

$$\mathbf{h}_{ne(i)}^{(l)} = \sum_{j \in ne(i)} \frac{1}{\sqrt{\tilde{\mathbf{D}}_{i,i}\tilde{\mathbf{D}}_{j,j}}} \mathbf{h}_j^{(l)} \sim \mathcal{N}\left(\sum_{j \in ne(i)} \frac{1}{\sqrt{\tilde{\mathbf{D}}_{i,i}\tilde{\mathbf{D}}_{j,j}}} \boldsymbol{\mu}_j^{(l)}, diag\left(\sum_{j \in ne(i)} \frac{1}{\tilde{\mathbf{D}}_{i,i}\tilde{\mathbf{D}}_{j,j}} \boldsymbol{\sigma}_j^{(l)} \right) \right)$$

- Variance-based Attention Mechanism
  - Large variance indicates more uncertainties → Down-weight large variance

$$\mathbf{h}_{ne(i)}^{(l)} = \sum_{j \in ne(i)} \frac{1}{\sqrt{\tilde{\mathbf{D}}_{i,i}\tilde{\mathbf{D}}_{j,j}}} (\mathbf{h}_j^{(l)} \odot \boldsymbol{\alpha}_j^{(l)}) \sim \mathcal{N}\left(\sum_{j \in ne(i)} \frac{\boldsymbol{\mu}_j^{(l)} \odot \boldsymbol{\alpha}_j^{(l)}}{\sqrt{\tilde{\mathbf{D}}_{i,i}\tilde{\mathbf{D}}_{j,j}}}, diag\left(\sum_{j \in ne(i)} \frac{\boldsymbol{\sigma}_j^{(l)} \odot \boldsymbol{\alpha}_j^{(l)} \odot \boldsymbol{\alpha}_j^{(l)}}{\tilde{\mathbf{D}}_{i,i}\tilde{\mathbf{D}}_{j,j}} \right) \right)$$

$$\boldsymbol{\alpha}_j^{(l)} = exp(-\gamma \boldsymbol{\sigma}_j^{(l)})$$

D. Zhu, Z. Zhang, P. Cui, and W. Zhu. Robust graph convolutional networks against adversarial attacks. KDD, 2019.

RGCN

- Vanilla GCN:

$$\mathbf{H}^{(l+1)} = \rho\left(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right)$$

- RGCN:

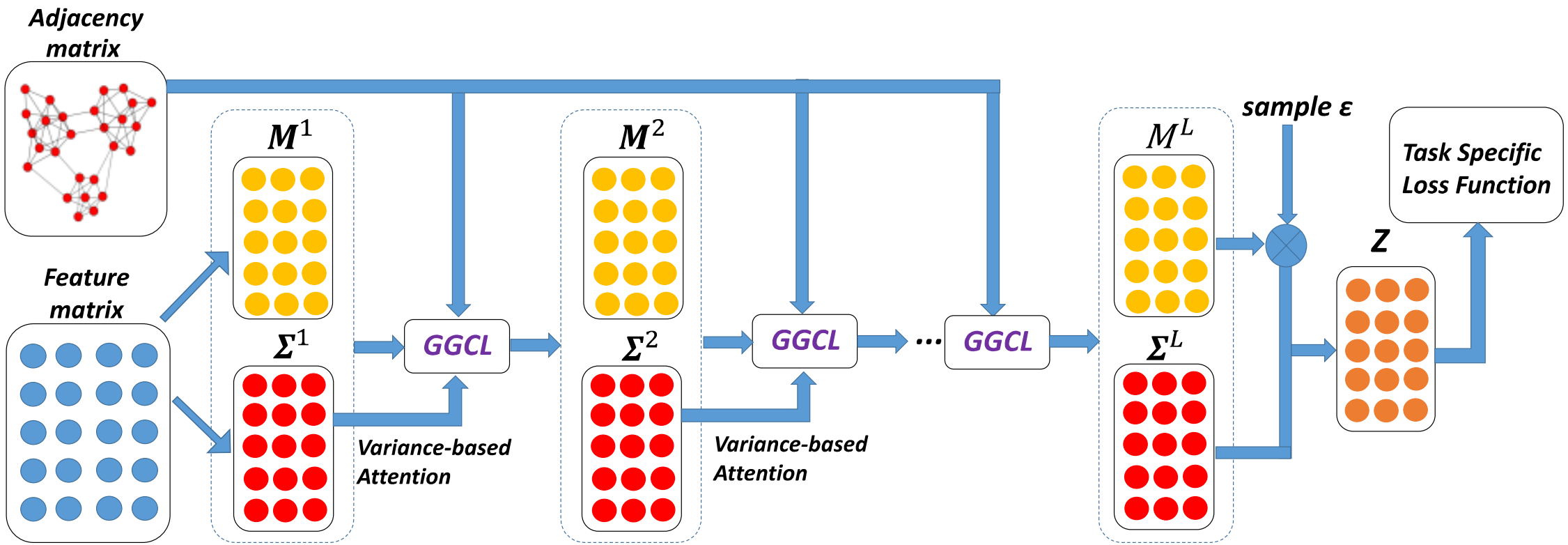
$$\mathbf{M}^{(l+1)} = \rho\left(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\left(\mathbf{M}^{(l)} \odot \mathcal{A}^{(l)}\right)\mathbf{W}_{\mu}^{(l)}\right)$$

$$\Sigma^{(l+1)} = \rho\left(\tilde{\mathbf{D}}^{-1}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1}\left(\Sigma^{(l)} \odot \mathcal{A}^{(l)} \odot \mathcal{A}^{(l)}\right)\mathbf{W}_{\sigma}^{(l)}\right)$$

where: $\mathcal{A}^{(l)} = exp(-\gamma\Sigma^{(l)})$

D. Zhu, Z. Zhang, P. Cui, and W. Zhu. Robust graph convolutional networks against adversarial attacks. KDD, 2019.

RGCN



D. Zhu, Z. Zhang, P. Cui, and W. Zhu. Robust graph convolutional networks against adversarial attacks. KDD, 2019.

GNNGuard

- GNNGuard assigns higher weights to edges connecting similar nodes while pruning edges between unrelated nodes.
  - Quantify similarity s_{uv}^k between u and v in the k-th layer of GNN as:

$$s_{uv}^k = d(\boldsymbol{h}_u^k, \boldsymbol{h}_v^k), \qquad d(\boldsymbol{h}_u^k, \boldsymbol{h}_v^k) = (\boldsymbol{h}_u^k \odot \boldsymbol{h}_v^k)/(||\boldsymbol{h}_u^k||_2 ||\boldsymbol{h}_v^k||_2)$$

  - Normalized node similarities:

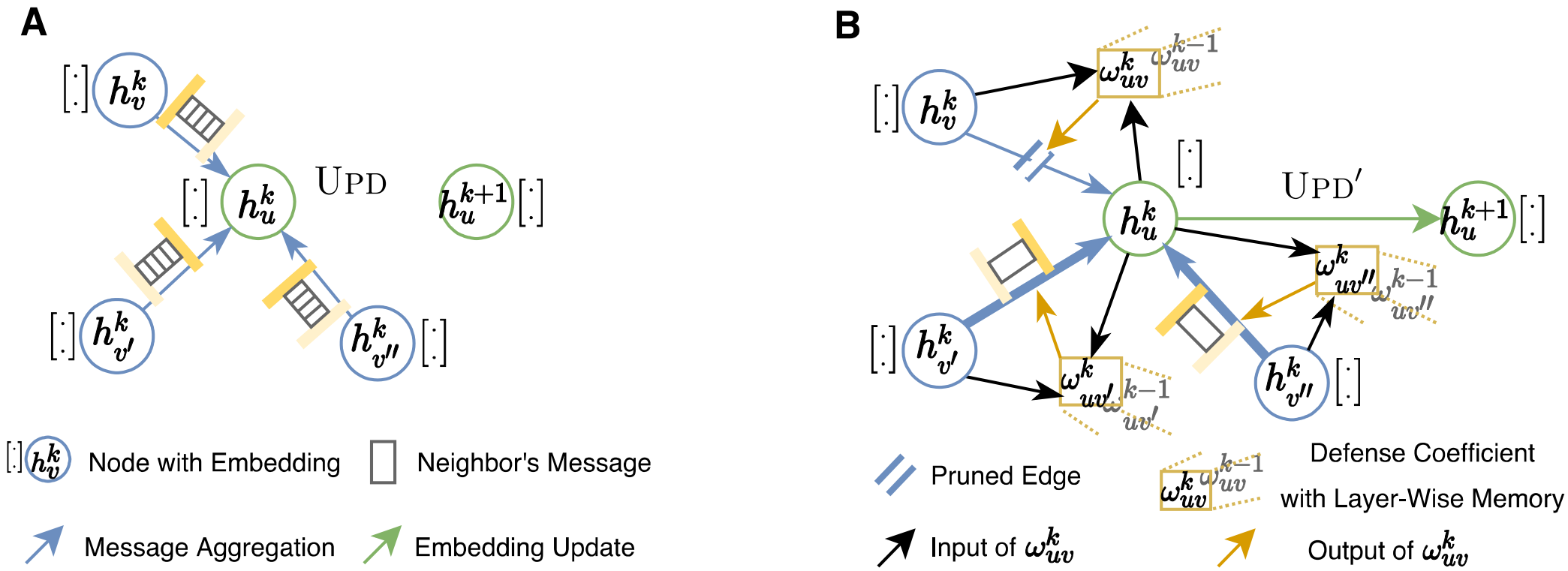
$$\hat{e}_{uv}^k = \begin{cases} s_{uv}^k / \sum_{v \in \hat{N}_u^k} s_{uv}^k \times \hat{N}_u^k / (\hat{N}_u^k + 1) & \text{if } u \neq v \\ 1/(\hat{N}_u^k + 1) & \text{if } u = v \end{cases}$$

  - Updated importance weight:

$$\alpha_{uv}^k = \beta \alpha_{uv}^{k-1} + (1-\beta)\hat{e}_{uv}^k$$

X. Zhang and M. Zitnik. Gnnguard: Defending graph neural networks against adversarial attacks. NeurIPS, 2020.

GNNGuard



X. Zhang and M. Zitnik. Gnnguard: Defending graph neural networks against adversarial attacks. NeurIPS, 2020.

SoftMedian

- **Vanilla Message Passing:** mean or sum → not robust

$$\mathbf{h}_v^{(l)} = \sigma^{(l)}\left[\mathrm{AGG}^{(l)}\left\{\left(\boldsymbol{A}_{vu}, \mathbf{h}_u^{(l-1)}\boldsymbol{W}^{(l)}\right), \forall\, u \in \mathbb{N}'(v)\right\}\right]$$

- **Dimension-wise Median:** $\bar{\mathbf{x}}$ robust but not differentiable

- **Soft Median:**

$$\mu_{\text{SoftMedian}}(X) = \text{softmax}(-c/T^{p}\,\bar{d})^{\top} X = s^{\top} X \;\dot{\in}\; \arg\min_{x_o \in X} k\bar{x} - x_o k$$

  - Distance: $c_v = k\bar{x} - X_{v,:}k$
  - Temperature T:
    - $T \to 0$: recover dimension-wise Median
    - $T \to \infty$: equivalent to sample mean

S. Geisler, T. Schmidt, H. S¸ irin, D. Z¨ugner, A. Bojchevski, and S. G¨unnemann. Robustness of graph neural networks at scale. NeurIPS, 2021.

Overview

Presenter:
Zhichao Hou

Traditional Approaches

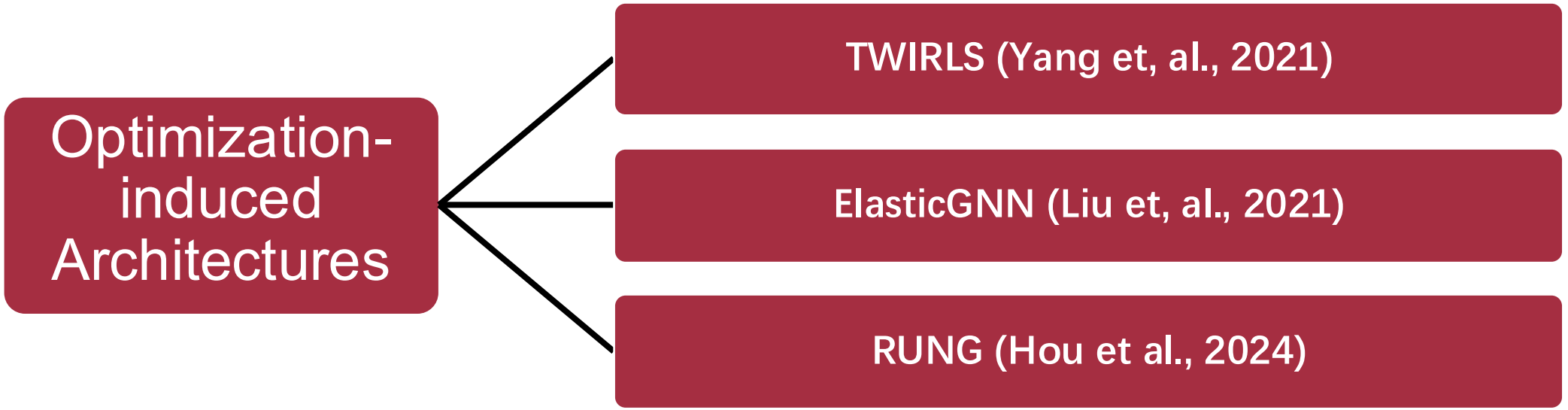
- Robust Training
- Graph Cleaning
- Attention Mechanism

Emerging Techniques

- Optimization-induced Architectures
- Innovative Perspective

Optimization-induced Architectures

Optimization-induced Architectures refers to a series of GNNs that are unrolled from optimization algorithms designed to solve specific graph smoothing estimators.



Sun, Lichao, et al. "Adversarial attack and defense on graph data: A survey." *IEEE Transactions on Knowledge and Data Engineering,* 2022.

Optimization-induced Architecture

Optimization-induced Architectures refers to a series of GNNs that are unrolled from optimization algorithms designed to solve specific graph smoothing estimators.

E.g., Attention Layer

	Vanilla Architecture	Optimization-induced Architecture
Underlying Problem	$\operatorname{argmin}_{\mathbf{z}} \mathcal{L}(\mathbf{z}) = \sum_{i=1}^{N} a_i \cdot \|\mathbf{z} - \mathbf{x}_i\|^2$	
Architecture	$\mathbf{y} = \sum_{i=1}^{N} a_i \cdot \mathbf{x}_i$	$\mathbf{z}^{(k+1)} = \mathbf{z}^{(k)} - \alpha \nabla_{\mathbf{z}} \mathcal{L}(\mathbf{z}^{(k)})$ $= \mathbf{z}^{(k)} - \alpha \sum_{i=1}^{N} 2a_i \cdot (\mathbf{z}^{(k)} - \mathbf{x}_i)$
Model	Direct	Indirect
Formulation	Closed-form	Iterative

Sun, Lichao, et al. "Adversarial attack and defense on graph data: A survey." *IEEE Transactions on Knowledge and Data Engineering*, 2022.

ElasticGNN

- Popular GNNs such as GCN, GAT, PPNP, and APPNP → ℓ_2-based graph smoothing

$$\arg\min_{\mathbf{F}} \mathcal{L}(\mathbf{F}) := \|\mathbf{F} - \mathbf{X}_{\text{in}}\|_F^2 + \lambda \operatorname{tr}(\mathbf{F}^\top \mathbf{L}\mathbf{F})$$

where $\mathbf{X}_{\text{in}} \in \mathbb{R}^{n\times d}$ is the input signal and $\mathbf{L} \in \mathbb{R}^{n\times n}$ is the graph Laplacian matrix encoding the graph structure.

GCN. The message passing scheme in Graph Convolutional Networks (GCN) (Kipf & Welling, 2016),

$$\mathbf{X}_{\text{out}} = \tilde{\mathbf{A}}\mathbf{X}_{\text{in}},$$

is equivalent to one gradient descent step to minimize $\operatorname{tr}(\mathbf{F}^\top(\mathbf{I} - \tilde{\mathbf{A}})\mathbf{F})$ with the initial $\mathbf{F} = \mathbf{X}_{\text{in}}$ and stepsize $1/2$. Here $\tilde{\mathbf{A}} = \hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-\frac{1}{2}}$ with $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ being the adjacent matrix with self-loop, whose degree matrix is $\hat{\mathbf{D}}$.

PPNP & APPNP. The message passing scheme in PPNP and APPNP (Klicpera et al., 2018) follow the aggregation rules

$$\mathbf{X}_{\text{out}} = \alpha\big(\mathbf{I} - (1-\alpha)\tilde{\mathbf{A}}\big)^{-1}\mathbf{X}_{\text{in}},$$

and

$$\mathbf{X}^{(k+1)} = (1-\alpha)\tilde{\mathbf{A}}\mathbf{X}^{(k)} + \alpha\mathbf{X}_{\text{in}}.$$

They are shown to be the exact solution and one gradient descent step with stepsize $\alpha/2$ for the following problem

$$\min_{\mathbf{F}} \|\mathbf{F} - \mathbf{X}_{\text{in}}\|_F^2 + (1/\alpha - 1)\operatorname{tr}(\mathbf{F}^\top(\mathbf{I} - \tilde{\mathbf{A}})\mathbf{F}). \quad (3)$$

X. Liu, W. Jin, Y. Ma, Y. Li, H. Liu, Y. Wang, M. Yan, and J. Tang. Elastic graph neural networks. ICML, 2021.

ElasticGNN

- Popular GNNs such as GCN, GAT, PPNP, and APPNP → ℓ_2-based graph smoothing

$$\arg\min_{\mathbf{F}} \mathcal{L}(\mathbf{F}) := \|\mathbf{F} - \mathbf{X}_{\text{in}}\|_F^2 + \lambda \operatorname{tr}(\mathbf{F}^\top \mathbf{L}\mathbf{F})$$

where $\mathbf{X}_{\text{in}} \in \mathbb{R}^{n\times d}$ is the input signal and $\mathbf{L} \in \mathbb{R}^{n\times n}$ is the graph Laplacian matrix encoding the graph structure.

- ℓ_2-based graph smoothing: $\operatorname{tr}(\mathbf{F}^\top \tilde{\mathbf{L}}\mathbf{F}) = \sum_{(v_i,v_j)\in\mathcal{E}} \left\| \frac{\mathbf{F}_i}{\sqrt{d_i+1}} - \frac{\mathbf{F}_j}{\sqrt{d_j+1}} \right\|_2^2$
  - Quadratic penalty on large feature difference → more sensitive to malicious edge
- ℓ_1-based graph smoothing: $\|\tilde{\Delta}\mathbf{F}\|_1 = \sum_{(v_i,v_j)\in\mathcal{E}} \left\| \frac{\mathbf{F}_i}{\sqrt{d_i+1}} - \frac{\mathbf{F}_j}{\sqrt{d_j+1}} \right\|_1$
  - Linear penalty on large feature difference → less sensitive to malicious edge

X. Liu, W. Jin, Y. Ma, Y. Li, H. Liu, Y. Wang, M. Yan, and J. Tang. Elastic graph neural networks. ICML, 2021.

ElasticGNN

❑ Elastic Graph Signal Estimator:

$$\underset{\mathbf{F}\in\mathbb{R}^{n\times d}}{\arg\min} \underbrace{\lambda_1\|\tilde{\Delta}\mathbf{F}\|_1}_{g_1(\tilde{\Delta}\mathbf{F})} + \underbrace{\frac{\lambda_2}{2}\mathrm{tr}(\mathbf{F}^\top\tilde{\mathbf{L}}\mathbf{F}) + \frac{1}{2}\|\mathbf{F}-\mathbf{X}_{\text{in}}\|_F^2}_{f(\mathbf{F})}$$

- Non-smoothness objective → equivalent saddle point problem & primal-dual algorithm.

❑ Elastic Message Passing:

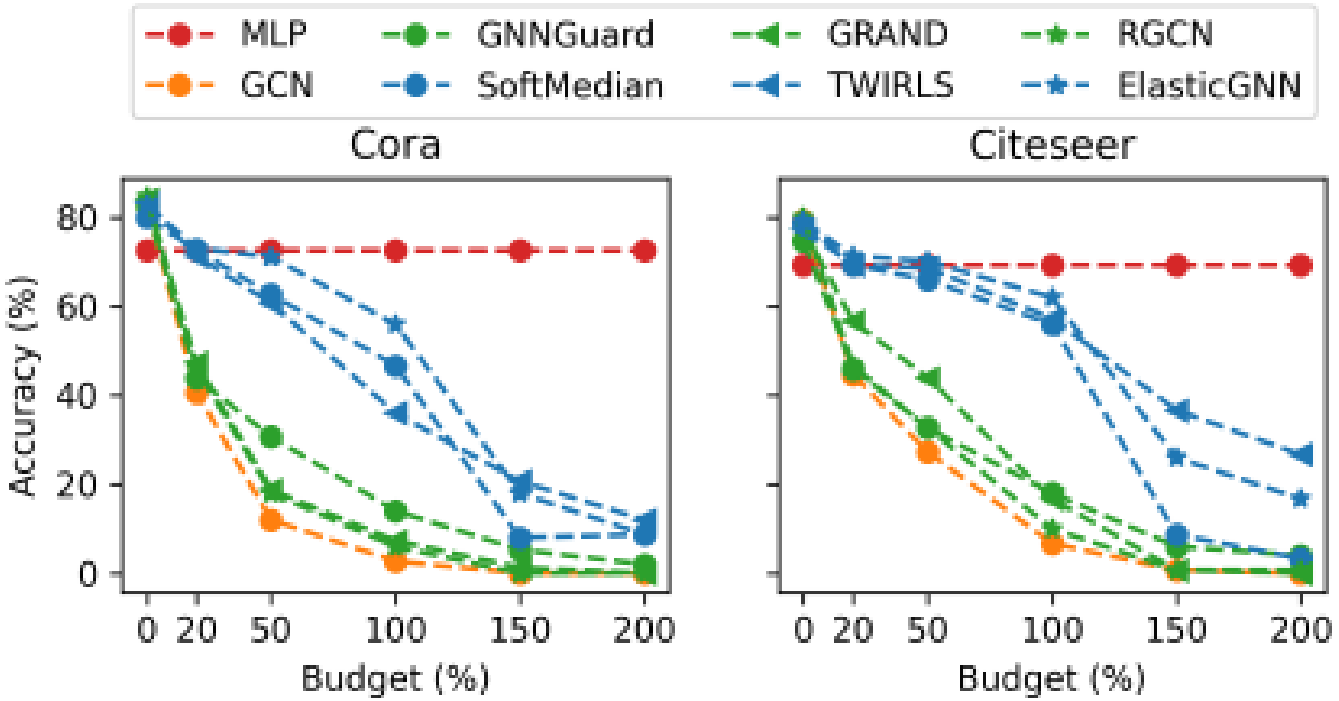
$$\begin{cases} \mathbf{Y}^{k+1} = \gamma\mathbf{X}_{\text{in}} + (1-\gamma)\tilde{\mathbf{A}}\mathbf{F}^k \\ \bar{\mathbf{F}}^{k+1} = \mathbf{Y}^k - \gamma\tilde{\Delta}^\top\mathbf{Z}^k \\ \bar{\mathbf{Z}}^{k+1} = \mathbf{Z}^k + \beta\tilde{\Delta}\bar{\mathbf{F}}^{k+1} \\ \begin{cases} \mathbf{Z}^{k+1} = \min(|\bar{\mathbf{Z}}^{k+1}|, \lambda_1)\cdot\mathrm{sign}(\bar{\mathbf{Z}}^{k+1}) & \textbf{(Option I: } \ell_1 \text{ norm)} \\ \mathbf{Z}_i^{k+1} = \min(\|\bar{\mathbf{Z}}_i^{k+1}\|_2, \lambda_1)\cdot\frac{\bar{\mathbf{Z}}_i^{k+1}}{\|\bar{\mathbf{Z}}_i^{k+1}\|_2}, \forall i\in[m] & \textbf{(Option II: } \ell_{21} \text{ norm)} \end{cases} \\ \mathbf{F}^{k+1} = \mathbf{Y}^k - \gamma\tilde{\Delta}^\top\mathbf{Z}^{k+1} \end{cases}$$

Figure 1. Elastic Message Passing (EMP). $\mathbf{F}^0 = \mathbf{X}_{\text{in}}$ and $\mathbf{Z}^0 = \mathbf{0}^{m\times d}$.

X. Liu, W. Jin, Y. Ma, Y. Li, H. Liu, Y. Wang, M. Yan, and J. Tang. Elastic graph neural networks. ICML, 2021.

RUNG

Robustness Analysis of Existing Methods:



- All defenses encounter a catastrophic performance degradation as the attack budget scales up.
- SoftMedian, TWIRLS, and ElasticGNN exhibit notable and closely aligned improvements.

Z. Hou, R. Feng, T. Derr, and X. Liu. Robust graph neural networks via unbiased aggregation, NeurIPS, 2024.

RUNG

Unified View of Robust Estimation

SoftMedian, TWIRLS, and ElasticGNN are ℓ_1-graph smoothing:

SoftMedian [4] substitutes the GCN aggregation for enhanced robustness with the dimension-wise median $\boldsymbol{m}_i \in \mathbb{R}^d$ for all neighbors of each node $i \in \mathcal{V}$. However, computing the median involves operations like ranking and selection, which is not compatible with the back-propagation training of GNNs. Therefore, the median is approximated as a differentiable weighted sum $\tilde{\boldsymbol{m}}_i = \frac{1}{Z}\sum_{j\in\mathcal{N}(i)} w(\boldsymbol{f}_j, \boldsymbol{m}_i)\boldsymbol{f}_j, \forall i \in \mathcal{V}$, where $\boldsymbol{m}_i$ is the exact non-differentiable dimension-wise median, $\boldsymbol{f}_j$ is the feature vector of the j-th neighbor, $w(\boldsymbol{x}, \boldsymbol{y}) = e^{-\beta\|\boldsymbol{x}-\boldsymbol{y}\|_2}$, and $Z = \sum_k w(\boldsymbol{f}_k, \boldsymbol{m}_k)$ is a normalization factor. In this way, the aggregation assigns the largest weights to the neighbors closest to the actual median.

TWIRLS [14] utilizes the iteratively reweighted least squares (IRLS) algorithm to optimize the objective with parameter λ, and $\rho(y) = y$ is the default:

$$2\lambda \sum_{(i,j)\in\mathcal{E}} \rho(\|\tilde{\boldsymbol{f}}_i - \tilde{\boldsymbol{f}}_j\|_2) + \sum_{i\in\mathcal{V}} \|\tilde{\boldsymbol{f}}_i - \tilde{\boldsymbol{f}}^{(0)}\|_2^2, \tilde{\boldsymbol{f}}_i = (1+\lambda d_i)^{-\frac{1}{2}}\boldsymbol{f}_i \tag{1}$$

ElasticGNN [15] proposes the elastic message passing which unfolds the proximal alternating predictor-corrector (PAPC) algorithm to minimize the objective with parameter $\lambda_{\{1,2\}}$:

$$\frac{1}{2}\sum_{i\in\mathcal{V}} \|\boldsymbol{f}_i - \boldsymbol{f}_i^{(0)}\|_2^2 + \lambda_1 \sum_{(i,j)\in\mathcal{E}} \left\| \frac{\boldsymbol{f}_i}{\sqrt{d_i}} - \frac{\boldsymbol{f}_j}{\sqrt{d_j}} \right\|_p + \lambda_2 \sum_{(i,j)\in\mathcal{E}} \left\| \frac{\boldsymbol{f}_i}{\sqrt{d_i}} - \frac{\boldsymbol{f}_j}{\sqrt{d_j}} \right\|_2^2, \text{where } p \in \{1,2\}. \tag{2}$$

Z. Hou, R. Feng, T. Derr, and X. Liu. Robust graph neural networks via unbiased aggregation, NeurIPS, 2024.

RUNG

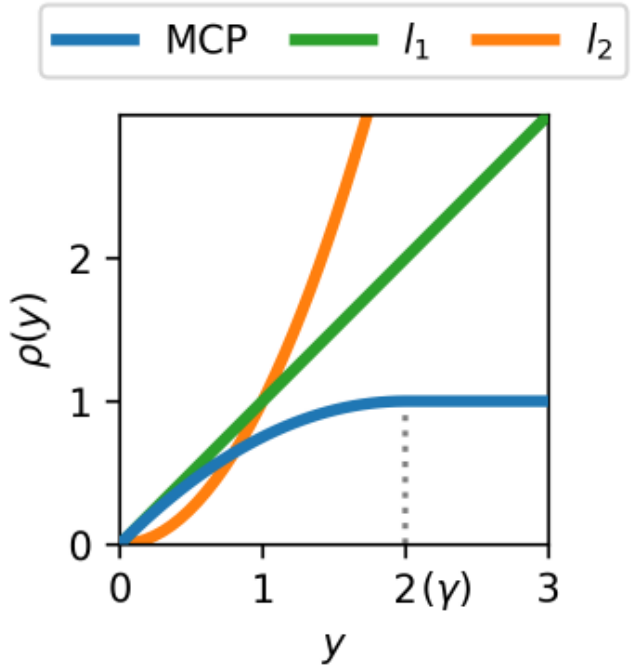
Unbiased Graph Signal Estimator (RUGE):

$$\arg\min_F H(F) = \sum_{(i,j)\in E} \rho_\gamma\left(\left\| \frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right\|_2\right) + \lambda \sum_{i\in V} \|f_i - f_i^{(0)}\|_2^2,$$

where $\rho_\gamma(y)$ denotes the function that penalizes the feature differences on edges by MCP:

$$\rho_\gamma(y) = \begin{cases} y - \frac{y^2}{2\gamma} & \text{if } y < \gamma \\ \frac{\gamma}{2} & \text{if } y \geq \gamma \end{cases}.$$

Non-convex and non-smooth



Alternative Objective - Localized Upper Bound:

$$\hat{H}(F) = \sum_{(i,j)\in E} W_{ij}^{(k)} \left\| \frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right\|_2^2 + \lambda \sum_{i\in V} \|f_i - f_i^{(0)}\|_2^2,$$

where $W_{ij}^{(k)} = 1_{i\neq j} \frac{d\rho_\gamma(y_{ij})}{dy_{ij}^2}\Big|_{y_{ij}=y_{ij}^{(k)}}$, where $y_{ij}^{(k)} = \left\| f_i^{(k)}/\sqrt{d_i} - f_j^{(k)}/\sqrt{d_j} \right\|_2$

Locally quadratic and smooth

Z. Hou, R. Feng, T. Derr, and X. Liu. Robust graph neural networks via unbiased aggregation, NeurIPS, 2024.

RUNG

- **Unbiased Graph Signal Estimator (RUGE):**

$$\arg\min_F H(F) = \sum_{(i,j)\in E} \rho_\gamma\left(\left\| \frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right\|_2\right) + \lambda \sum_{i\in V} \|f_i - f_i^{(0)}\|_2^2,$$

where $\rho_\gamma(y)$ denotes the function that penalizes the feature differences on edges by MCP:

$$\rho_\gamma(y) = \begin{cases} y - \frac{y^2}{2\gamma} & \text{if } y < \gamma \\ \frac{\gamma}{2} & \text{if } y \geq \gamma \end{cases}.$$

- **Robust Unbiased Aggregation (RUNG)** by Quasi-Newton IRLS Algorithm:

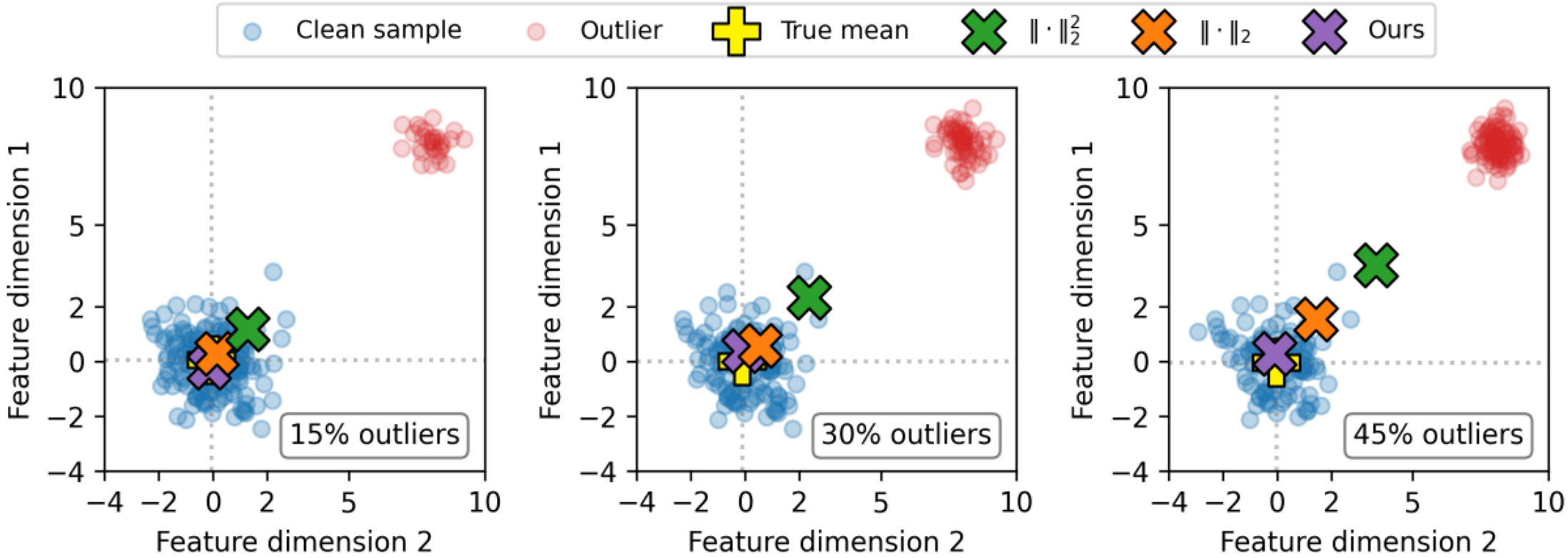
$$F^{(k+1)} = (\mathrm{diag}(q^{(k)}) + \lambda I)^{-1} \quad (W^{(k)} \odot \tilde{A})F^{(k)} + \lambda F^{(0)}$$

where $q_m^{(k)} = \sum_j W_{mj}^{(k)} A_{mj}/d_m$ as in Eq. (6), $W_{ij}^{(k)} = 1_{i\sim j} \max(0, \frac{1}{y_{ij}^{(k)}} - \frac{1}{\gamma})$ and $y_{ij}^{(k)} = \left\| \frac{f_i^{(k)}}{\sqrt{d_i}} - \frac{f_j^{(k)}}{\sqrt{d_j}} \right\|_2$.

Z. Hou, R. Feng, T. Derr, and X. Liu. Robust graph neural networks via unbiased aggregation, NeurIPS, 2024.

RUNG

- Simulation Experiment Visualization



Z. Hou, R. Feng, T. Derr, and X. Liu. Robust graph neural networks via unbiased aggregation, NeurIPS, 2024.

RUNG

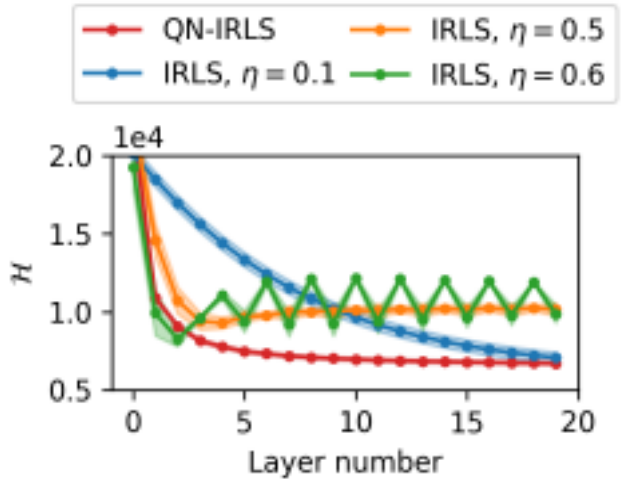


Figure 5: Convergence of our QN-IRLS compared to IRLS.

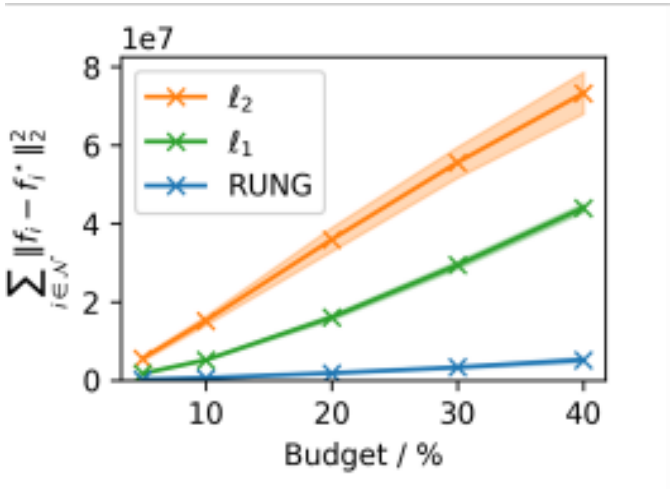


Figure 6: Bias induced by different attack budgets.

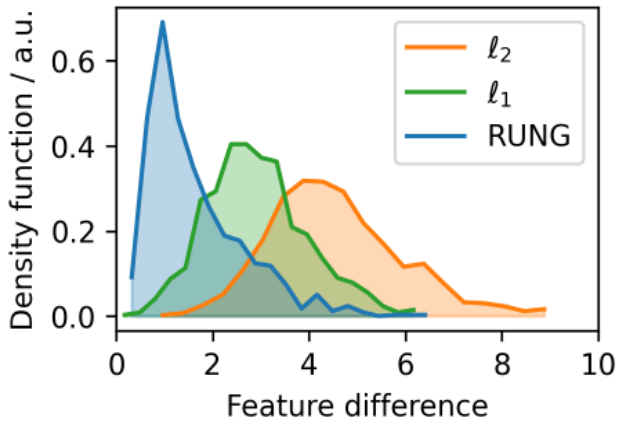
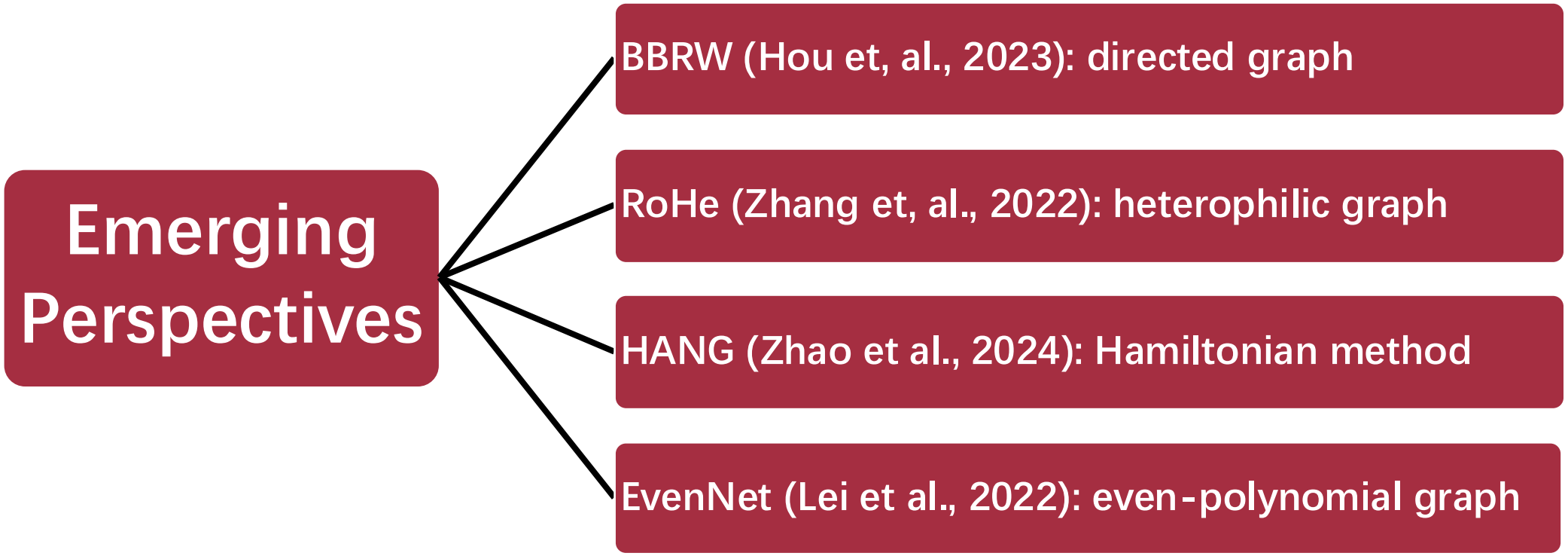


Figure 7: Distribution of feature difference on attacked edges.

Z. Hou, R. Feng, T. Derr, and X. Liu. Robust graph neural networks via unbiased aggregation, NeurIPS, 2024.

Emerging Perspectives

Emerging Perspectives include several new advancements in GNN robustness, including orthogonal directions to existing paradigms and promising foundational theories.



Sun, Lichao, et al. "Adversarial attack and defense on graph data: A survey." *IEEE Transactions on Knowledge and Data Engineering,* 2022.

Biased Bidirectional Randm Walk

- Existing attack algorithms are **undirected** by flipping both out-link and in-link.

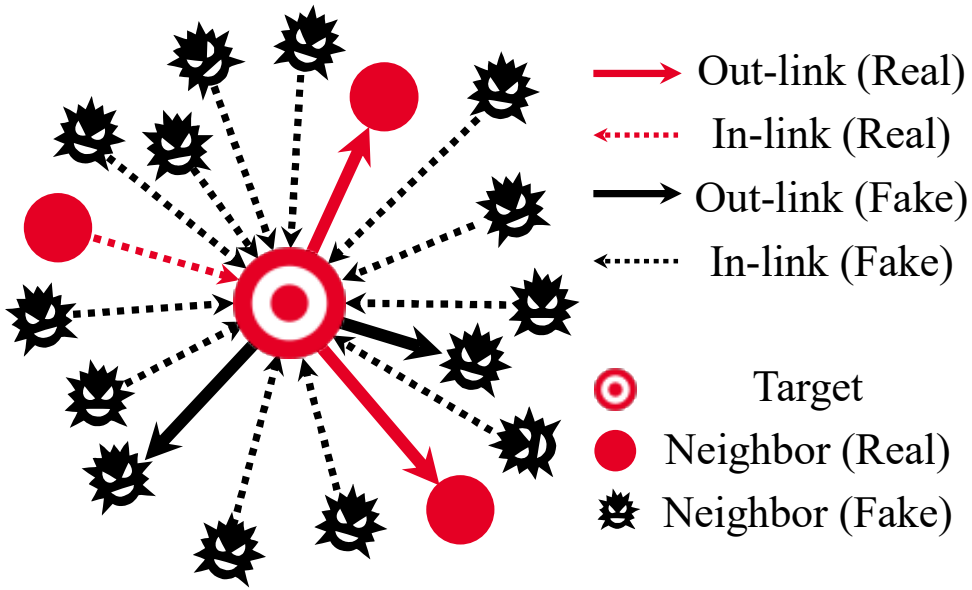


Figure 1: Large-scale link spam.

- Easy to create many fake users to follow the target user. (**in-links**)
- Difficult to hack into the account and manilpulate their following behaviors. (**out-links**)

→ Trust more on out-links will contribute to better robustness

Z. Hou, X. Zhang, W. Wang, C. C. Aggarwal, and X. Liu. Can directed graph neural networks be adversarially robust? arXiv preprint, 2023.

Biased Bidirectional Randm Walk

- Catastrophic Failures of Directed Random Walk due to **Indirect Attack**

Table 1: Accuracy (%) on Cora-ML under RDGA.

Budget	0%	25%		50%	
	Clean	Transfer	Adaptive	Transfer	Adaptive
MLP	73.5±7.4	73.5±7.4	73.5±7.4	73.5±7.4	73.5±7.4
GCN	89.5±6.1	66.0±9.7	66.0±9.7	40.5±8.5	40.5±8.5
GCN-RW	86.5±6.3	86.5±6.3	52.0±8.1	86.5±6.3	28.0±4.6
APPNP	90.5±4.7	81.5±9.5	80.5±10.4	66.5±8.7	68.0±12.1
APPNP-RW	85.5±6.5	85.5±6.5	30.0±7.7	85.5±6.5	15.0±3.9

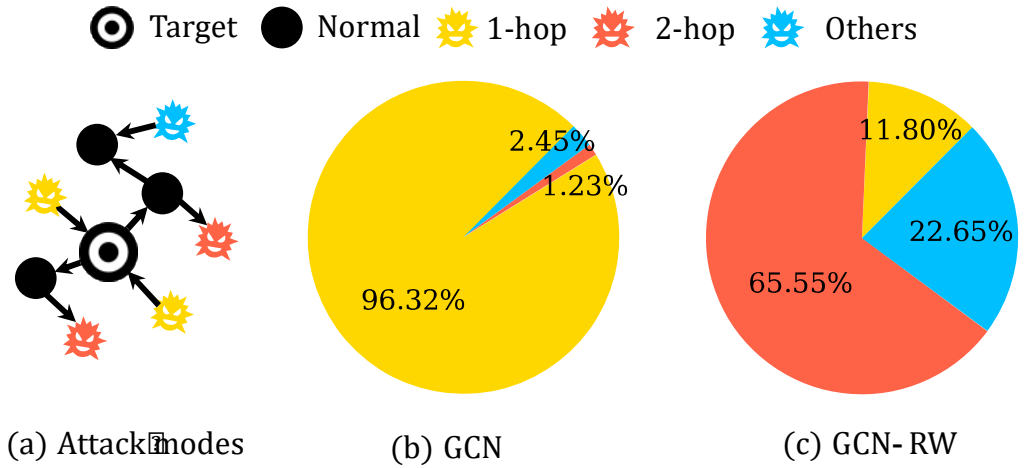


Figure 2: Adversary behaviors.

Z. Hou, X. Zhang, W. Wang, C. C. Aggarwal, and X. Liu. Can directed graph neural networks be adversarially robust? arXiv preprint, 2023.

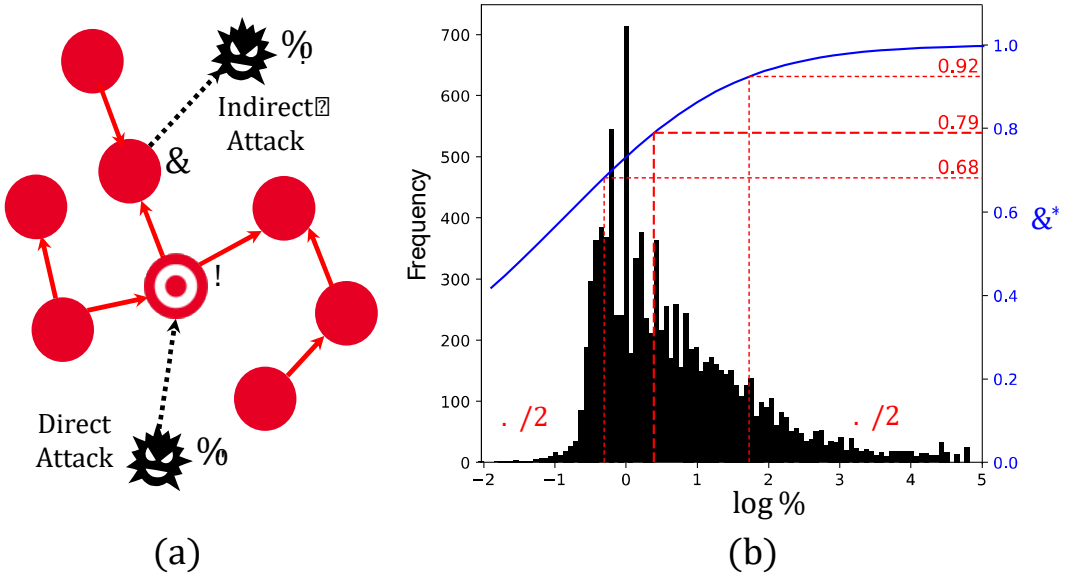
Biased Bidirectional Randm Walk

$$\beta^{\leftarrow} = \arg\min_{\beta 2 [0,1]} \max\{\tilde{A}^{2}_{\beta}(t, x_1), \tilde{A}^{2}_{\beta}(t, x_2)\}. \quad ($$

- Biased Bidirectional Randm Walk (BBRW):

$$\tilde{A}_{\beta} = D_{\beta}^{-1} A_{\beta},$$

$$\text{where } D_{\beta} = A_{\beta} 1, A_{\beta} = \beta A + (1-\beta) A^{>}.$$

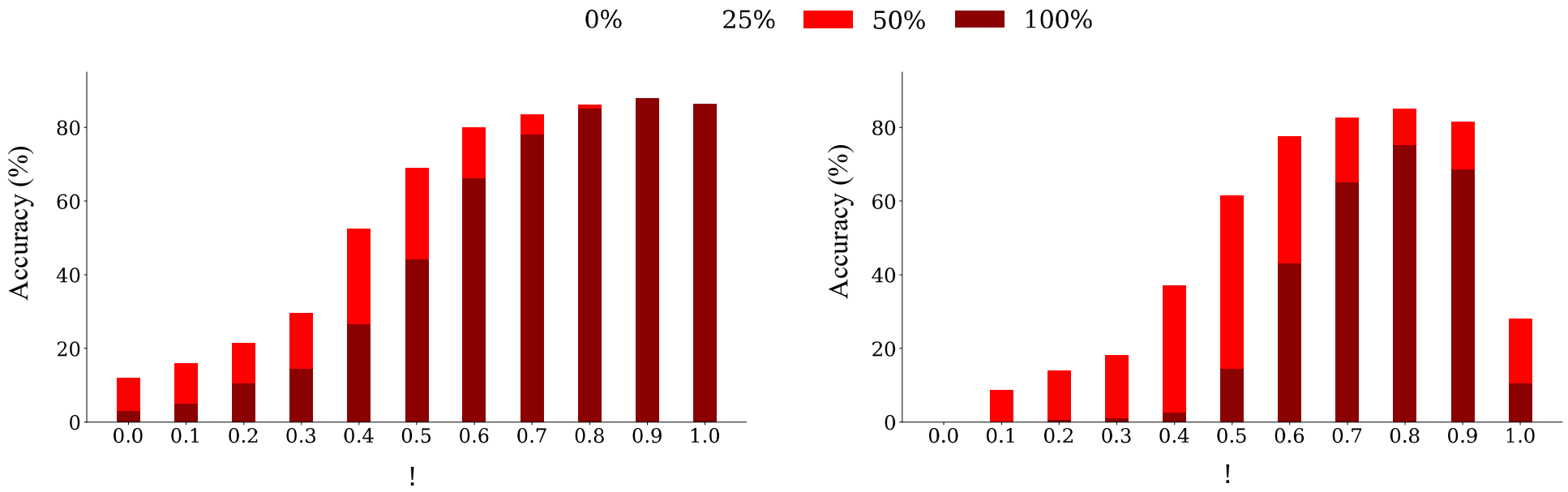


(a) (b)

Figure 3: Theoretical analysis of BBRW.

Z. Hou, X. Zhang, W. Wang, C. C. Aggarwal, and X. Liu. Can directed graph neural networks be adversarially robust? arXiv preprint, 2023.

Biased Bidirectional Randm Walk



(a) Transfer

(b) Adaptive

Z. Hou, X. Zhang, W. Wang, C. C. Aggarwal, and X. Liu. Can directed graph neural networks be adversarially robust? arXiv preprint, 2023.

Biased Bidirectional Randm Walk

Method	Clean (total)	0%	25%		50%		100%	
		Target	Transfer	Adaptive	Transfer	Adaptive	Transfer	Adaptive
MLP	64.6±2.2	73.5±7.4	73.5±7.4	73.5±7.4	73.5±7.4	73.5±7.4	73.5±7.4	73.5±7.4
DGCN	75.0±3.1	89.5±7.6	76.5±13.0	\	54.5±7.9	\	38.0±14.2	\
DiGCN	75.5±2.2	85.0±7.4	50.0±6.7	\	40.5±9.1	\	29.0±6.2	\
Directed-MagNet	57.1±5.2	69.5±10.4	65.0±9.7	\	59.5±10.6	\	54.0±7.0	\
Undirected-MagNet	79.6±2.1	88.5±3.2	70.5±10.6	\	55.5±6.9	\	35.5±6.1	\
Jaccard-GCN	81.0±1.6	90.5±6.5	69.5±7.9	65.5±7.9	44.0±6.2	34.0±7.0	21.0±7.0	8.0±4.6
RGCN	81.4±1.5	88.0±6.0	72.5±8.4	66.0±7.7	44.0±8.9	36.0±5.4	17.5±8.7	7.0±4.6
GRAND	81.2±0.9	85.5±6.1	74.0±7.0	65.0±7.4	64.0±9.2	51.0±8.6	45.0±7.1	24.0±7.7
ElasticGNN	79.0±0.7	89.0±6.2	86.0±5.4	\	74.0±5.8	\	50.0±9.7	\
GCN	81.8±1.5	89.5±6.1	66.0±9.7	66.0±9.7	40.5±8.5	40.5±8.5	12.0±6.4	12.0±6.4
BBRW-GCN	80.5±1.3	90.0±5.5	89.5±6.1	89.0±6.2	86.0±5.4	85.0±6.3	85.0±7.1	75.0±10.2
APPNP	**82.5±1.6**	90.5±4.7	81.5±9.5	80.5±10.4	66.5±8.7	68.0±12.1	44.0±9.2	46.0±7.3
BBRW-APPNP	**82.5±1.2**	91.0±4.9	89.0±5.4	87.5±5.6	85.0±7.1	83.0±6.4	83.5±6.3	69.0±9.7
GCN-Soft-Median	81.6±1.3	91.5±5.5	86.0±7.0	83.0±7.1	75.0±8.4	73.0±7.1	48.5±11.4	47.5±9.3
BBRW-GCN-Soft-Median	82.4±1.3	**92.0±4.6**	**91.5±5.0**	**92.0±4.6**	**89.5±6.9**	**88.0±5.1**	**87.0±8.4**	**84.5±8.8**

- BBRW is plug-in layer, which is **orthogonal** to and can be combined with existing defenses.

Z. Hou, X. Zhang, W. Wang, C. C. Aggarwal, and X. Liu. Can directed graph neural networks be adversarially robust? arXiv preprint, 2023.

HANG

- GNNs with **conservative Hamitonian neural flows** are robust to adversarial attacks.

Definition 4 (Conservative stability). *In a dynamical system inspired by physical principles, such as* (2)*, a conserved quantity might be present. This quantity, which frequently embodies the notion of the system's energy, remains invariant along the system's evolution trajectory* $\mathbf{z}(t)$.

Definition 5 (Dirichlet energy [35]). *The Dirichlet energy is defined on node features* $q(t)$ *at time* t *of an undirected graph* $\mathcal{G}$ *as*

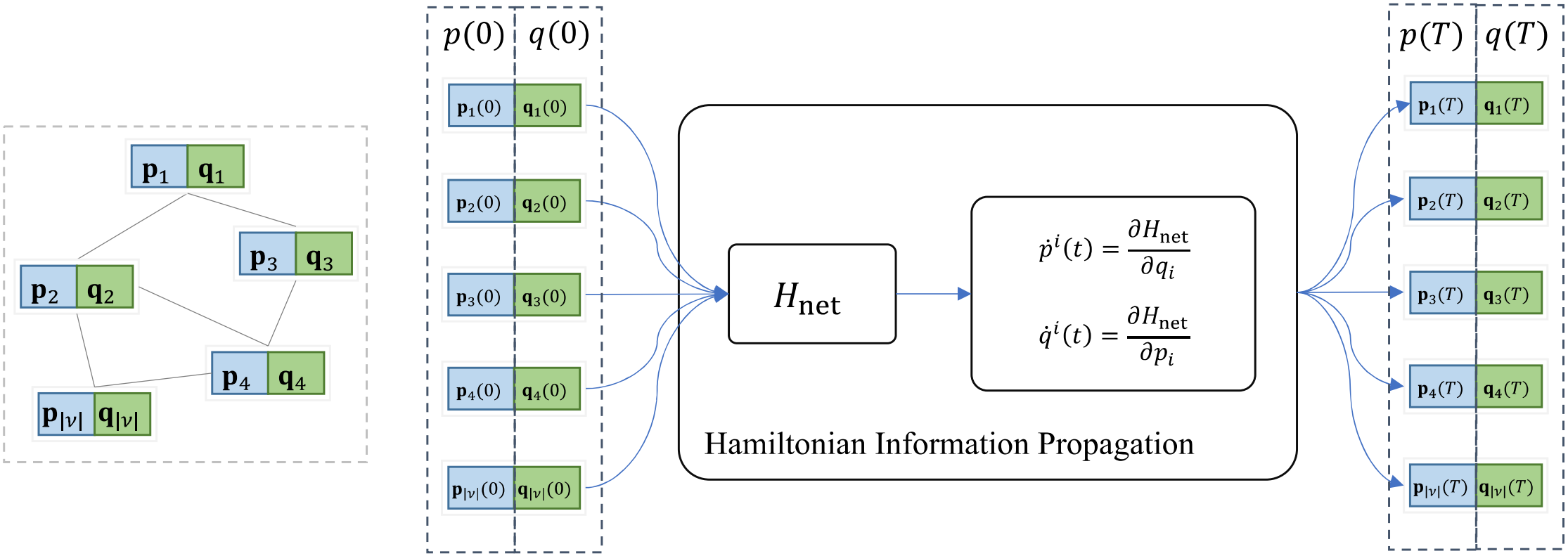
$$\mathcal{E}(q(t)) = \frac{1}{|\mathcal{V}|} \sum_i \sum_{j \in \mathcal{N}(i)} \|\mathbf{q}_i(t) - \mathbf{q}_j(t)\|^2, \tag{11}$$

where $\mathcal{N}(i) = \{j : [i, j] \in \mathcal{E}\}$ *is the set of neighbors adjacent to node* i *in the graph.*

→ Conserving Dirichlet energy is equivalent to preserving connecting nodes' similiarty

K. Zhao, Q. Kang, Y. Song, R. She, S. Wang, and W. P. Tay. Adversarial robustness in graph neural networks: A hamiltonian approach. NeurIPS, 2024.

HANG



$$p(t) = \left(p_1(t), \ldots p_{r|\mathcal{V}|}(t)\right) = \left(\mathbf{p}_1(t), \ldots, \mathbf{p}_{|\mathcal{V}|}(t)\right)$$

Momentum

$$q(t) = \left(q^1(t), \ldots q^{r|\mathcal{V}|}(t)\right) = \left(\mathbf{q}_1(t), \ldots, \mathbf{q}_{|\mathcal{V}|}(t)\right)$$

Position

K. Zhao, Q. Kang, Y. Song, R. She, S. Wang, and W. P. Tay. Adversarial robustness in graph neural networks: A hamiltonian approach. NeurIPS, 2024.

EvenNet

Balance Theory:

"The enemy of my enemy is my friend, and the friend of my friend is also my friend."

EvenNet:

$$Z = f(X, P) = \left(\sum_{k=0}^{\lfloor K/2 \rfloor} w_k P^{2k} t(X) \right)$$

R. Lei, Z. Wang, Y. Li, B. Ding, and Z. Wei. Evennet: Ignoring odd-hop neighbors improves robustness of graph neural networks. NeurIPS, 2022.

Overview

Presenter:
Zhichao Hou

Evaluation

- Empirical Robustness
- Adaptive Attack
- Provable/Certified Robustness

Empirical Robustness

- Empirical Robustness Evaluation
  - Evaluation metric:
    (1) *Accuracy*: drop in performance;
    (2) *Attack success rate*;
    (3) *Classification margin*: largest probability minus second-largest probability

  - Limitations:
    - Depend on a specific attack
    - Optimistic evaluation
    - False sense of security

  - Reliable evaluation:
    - Multiple different and powerful attacks
    - **Adaptive attack**

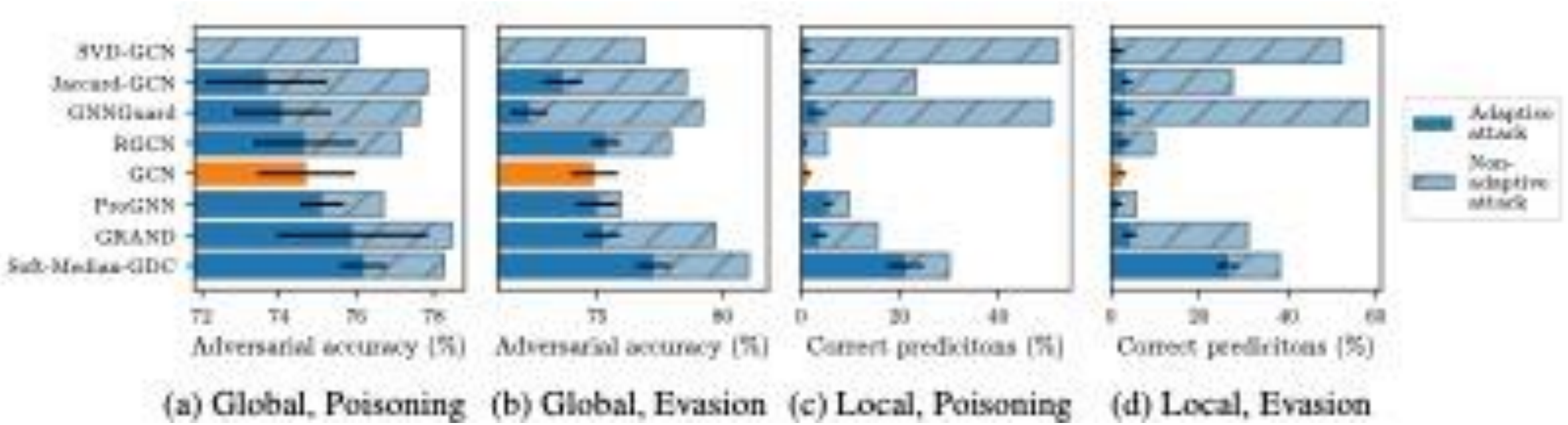
Adaptive Attack

- Non-adaptive Attacks
  - Do not directly attack the victim model,
    e.g., transfer attacks from surrogate model
  - Obfuscated gradients → False sense of security
    e.g., non-differentiable operator, such as low-rank approximation
- Adaptive Attacks
  - Directly attack the victim model and identify the malicious edges
    e.g., typically gradient-based attacks like PGD
  - Avoid obfuscated gradients
    e.g., SVD-GCN, approximate the non-differentiable operator by

$$\mathrm{LRA}(\mathbf{A}+\delta\mathbf{A}) \approx \mathrm{LRA}(\mathbf{A}) + \delta\mathbf{A} \circ \mathrm{Weight}(\mathbf{A})$$

Mujkanovic, Felix, et al. "Are defenses for graph neural networks robust?." NeurIPS, 2022.

Adaptive Attack



- Most of the defenses are evaluated against **non-adaptive attacks** leading to overly optimistic robustness estimates.
- All defenses are less robust than reported under **adaptive attacks**, with an undefended GCN outperforming some.

Mujkanovic, Felix, et al. "Are defenses for graph neural networks robust?." NeurIPS, 2022.

Provable/Certifiable Evaluation

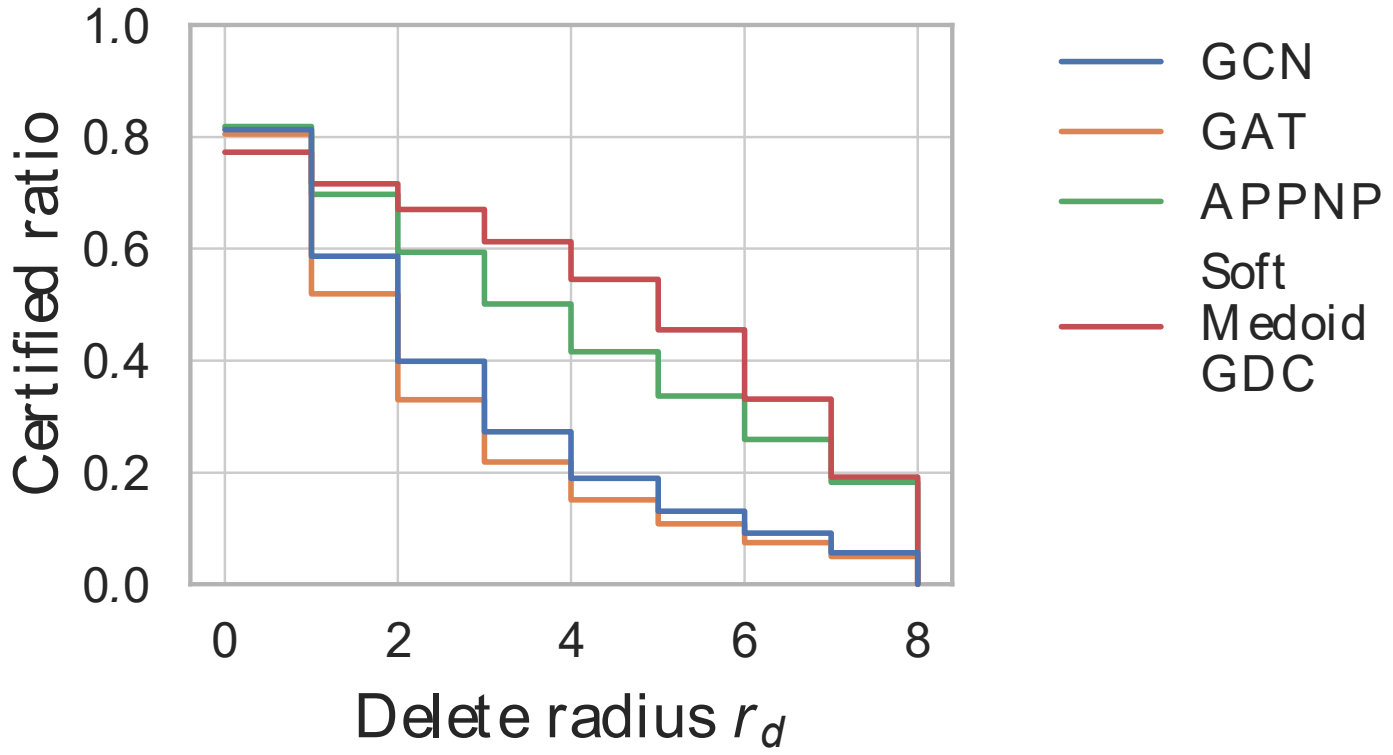
- Typical method: randomized smoothing
  - Base classifier: $f(x)$
  - Smoothed classifier: $g(x) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}(f(x+\varepsilon) = c)$ where $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$

For a specific prediction, randomized smoothing can provide theoretical gurantee that there is no attack within specific radius. (certified)

- Provable Robustness Evaluation
  - Evaluation metric:
    - (1) *Certified ratio*: ratio of sample predictions can be certified as robust given a specific $\boldsymbol{r}$
    - (2) *Certified correctness*: the prediction is both correct and certified
    - (3) *Certified performance/accuracy*: the ratio of "**certified correct**" sample
    - (4) *Certified radius*: for a specific prediction, the **largest** radius r^* for which this prediction can still be certified as robust.

Provable/Certifiable Evaluation

- Provable Robustness Evaluation



Performance

Table 2: Adaptive global attack on Cora ML. The best and second are marked.

Model	0% (Clean)	5%	10%	20%	30%	40%
MLP	65.0 ± 1.0	65.0 ± 1.0	65.0 ± 1.0	65.0 ± 1.0	65.0 ± 1.0	65.0 ± 1.0
GCN	85.0 ± 0.4	75.3 ± 0.5	69.6 ± 0.5	60.9 ± 0.7	54.2 ± 0.6	48.4 ± 0.5
APPNP	86.3 ± 0.4	75.8 ± 0.5	69.7 ± 0.7	60.3 ± 0.9	53.8 ± 1.2	49.0 ± 1.6
GAT	83.5 ± 0.5	75.8 ± 0.8	71.2 ± 1.2	65.0 ± 0.9	60.5 ± 0.9	56.7 ± 0.9
GNNGuard	83.1 ± 0.7	74.6 ± 0.7	70.2 ± 1.0	63.1 ± 1.1	57.5 ± 1.6	51.0 ± 1.2
RGCN	85.7 ± 0.4	75.0 ± 0.8	69.1 ± 0.4	59.8 ± 0.7	52.8 ± 0.7	46.1 ± 0.7
GRAND	86.1 ± 0.7	76.2 ± 0.8	70.7 ± 0.7	61.6 ± 0.7	56.7 ± 0.8	51.9 ± 0.9
ProGNN	85.6 ± 0.5	76.5 ± 0.7	71.0 ± 0.5	63.0 ± 0.7	56.8 ± 0.7	51.3 ± 0.6
Jaccard-GCN	83.7 ± 0.7	73.9 ± 0.5	68.3 ± 0.7	60.0 ± 1.1	54.0 ± 1.7	49.1 ± 2.4
GARNET	84.0 ± 0.5	76.5 ± 0.4	72.1 ± 0.1	66.4 ± 0.7	62.1 ± 1.3	58.7 ± 1.5
HANG	84.5 ± 0.2	75.7 ± 0.6	74.2 ± 0.3	69.5 ± 0.4	64.7 ± 0.9	58.4 ± 0.1
NoisyGNN	83.9 ± 0.5	76.7 ± 0.1	72.1 ± 0.3	64.7 ± 0.7	58.0 ± 0.2	53.3 ± 0.6
EvenNet	84.8 ± 0.9	75.8 ± 0.9	70.7 ± 0.6	63.9 ± 0.5	58.7 ± 0.7	54.7 ± 0.8
GraphCON	83.7 ± 0.6	75.8 ± 0.3	70.9 ± 0.8	65.9 ± 0.7	62.2 ± 0.9	56.6 ± 0.3
SoftMedian	85.0 ± 0.7	78.6 ± 0.3	75.5 ± 0.9	69.5 ± 0.5	62.8 ± 0.8	58.1 ± 0.7
TWIRLS	84.2 ± 0.6	77.3 ± 0.8	72.9 ± 0.3	66.9 ± 0.2	62.4 ± 0.6	58.7 ± 1.1
TWIRLS-T	82.8 ± 0.5	76.8 ± 0.6	73.2 ± 0.4	67.7 ± 0.4	63.8 ± 0.2	60.8 ± 0.3
RUNG-`$_1$ (Ours)	85.8 ± 0.5	78.4 ± 0.4	74.3 ± 0.3	68.1 ± 0.6	63.5 ± 0.7	59.8 ± 0.8
RUNG (Ours)	84.6 ± 0.5	78.9 ± 0.4	75.7 ± 0.2	71.8 ± 0.4	67.8 ± 1.3	65.1 ± 1.2

Z. Hou, R. Feng, T. Derr, and X. Liu. Robust graph neural networks via unbiased aggregation, NeurIPS, 2024.

Performance

Table 1: Adaptive local attack on Cora ML. The **best** and second are marked.

Model	0% (Clean)	20%	50%	100%	150%	200%
MLP	72.6 ± 6.4	72.6 ± 6.4	72.6 ± 6.4	$\mathbf{72.6 \pm 6.4}$	$\mathbf{72.6 \pm 6.4}$	$\mathbf{72.6 \pm 6.4}$
GCN	82.7 ± 4.9	40.7 ± 10.2	12.0 ± 6.2	2.7 ± 2.5	0.0 ± 0.0	0.0 ± 0.0
APPNP	$\mathbf{84.7 \pm 6.8}$	50.0 ± 13.0	27.3 ± 6.5	14.0 ± 5.3	3.3 ± 3.0	0.7 ± 1.3
GAT	80.7 ± 10.0	30.7 ± 16.1	16.0 ± 12.2	11.3 ± 4.5	1.3 ± 1.6	2.0 ± 1.6
GNNGuard	82.7 ± 6.7	44.0 ± 11.6	30.7 ± 11.6	14.0 ± 6.8	5.3 ± 3.4	2.0 ± 2.7
RGCN	84.6 ± 4.0	46.0 ± 9.3	18.0 ± 8.1	6.0 ± 3.9	0.0 ± 0.0	0.0 ± 0.0
GRAND	84.0 ± 6.8	47.3 ± 9.0	18.7 ± 9.1	7.3 ± 4.9	1.3 ± 1.6	0.0 ± 0.0
ProGNN	$\mathbf{84.7 \pm 6.2}$	47.3 ± 10.4	21.3 ± 7.8	4.0 ± 2.5	0.0 ± 0.0	0.0 ± 0.0
Jaccard-GCN	81.3 ± 5.0	46.0 ± 6.8	17.3 ± 4.9	4.7 ± 3.4	0.7 ± 1.3	0.7 ± 1.3
GARNET	82.4 ± 6.8	70.9 ± 7.5	61.9 ± 7.9	42.7 ± 9.3	11.6 ± 3.4	9.6 ± 3.5
HANG	83.1 ± 7.4	71.2 ± 7.8	60.1 ± 6.3	39.5 ± 3.4	9.4 ± 2.3	5.6 ± 2.3
NoisyGNN	82.5 ± 5.6	57.4 ± 5.7	47.8 ± 6.2	36.1 ± 4.5	5.8 ± 3.4	4.1 ± 1.2
EvenNet	83.4 ± 8.1	64.8 ± 6.9	56.1 ± 5.6	29.5 ± 5.3	3.1 ± 1.1	1.1 ± 1.3
GraphCON	81.3 ± 7.1	67.3 ± 7.3	54.7 ± 7.1	41.3 ± 6.2	4.1 ± 2.1	2.3 ± 1.2
SoftMedian	80.0 ± 10.2	72.7 ± 13.7	62.7 ± 12.7	46.7 ± 11.0	8.0 ± 4.5	8.7 ± 3.4
TWIRLS	83.3 ± 7.3	71.3 ± 8.6	60.7 ± 11.0	36.0 ± 8.8	20.7 ± 10.4	12.0 ± 6.9
TWIRLS-T	82.0 ± 4.5	70.7 ± 4.4	62.7 ± 7.4	54.7 ± 6.2	44.0 ± 11.2	40.7 ± 11.8
RUNG-ℓ_1 (Ours)	84.0 ± 6.8	72.7 ± 7.1	62.7 ± 11.2	53.3 ± 8.2	22.0 ± 9.3	14.0 ± 7.4
RUNG (Ours)	84.0 ± 5.3	$\mathbf{75.3 \pm 6.9}$	$\mathbf{72.7 \pm 8.5}$	70.7 ± 10.6	69.3 ± 9.8	69.3 ± 9.0

- Under adaptive attacks, many existing defenses are not significantly more robust than undefended models.
- The robustness improvement of RUNG appears to be more remarkable in **local attacks**. Local attacks are stronger than global attacks since local attacks concentrate on targeted nodes.

Z. Hou, R. Feng, T. Derr, and X. Liu. Robust graph neural networks via unbiased aggregation, NeurIPS, 2024.

Summary



- **Introduction** of GNNs Robustness
- Adversarial **Attacks** on Graph
- Adversarial **Defenses** on Graph
- **Evaluation** of Robust GNNs
- **Summary** and Discussion

Thanks for listening!

