COMP4222 Machine Learning with Structured Data

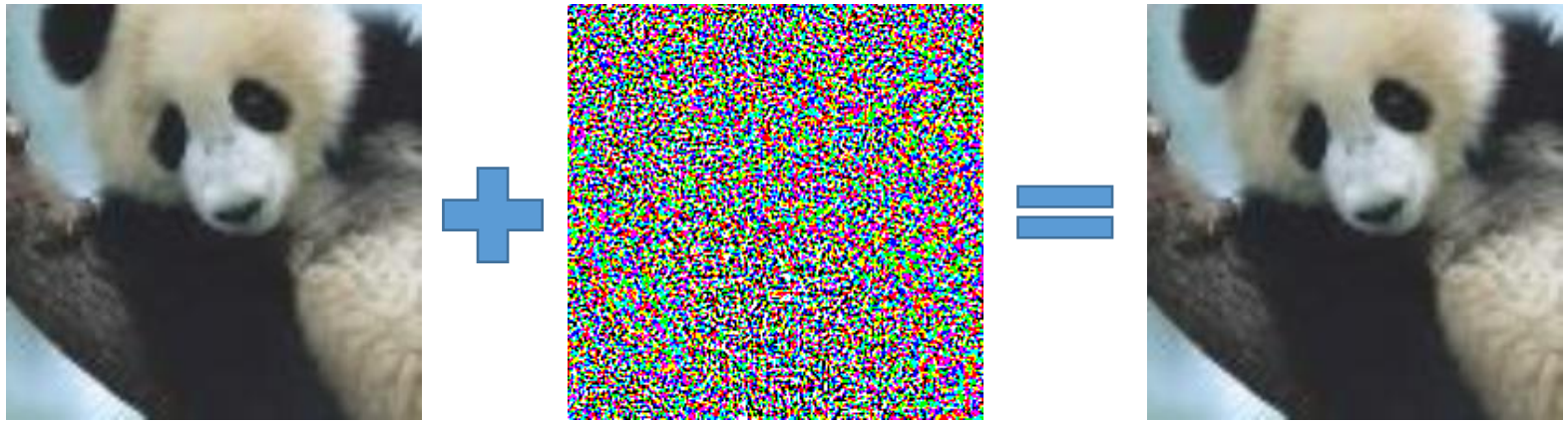
Advanced Topics of GNN

Instructor: Yangqiu Song

Slides credits: Yao Ma and Yiqi Wang, Tyler Derr, Lingfei Wu and Tengfei Ma, Jure Laskovec

Adversarial Attacks on Deep Learning

- Deep convolutional neural networks are vulnerable to **adversarial attacks:**
  - Imperceptible noise changes the prediction.
- Adversarial examples are also reported in natural language processing [Jia & Liang et al. EMNLP 2017] and audio processing [Carlini et al. 2018] domains.



Adversarial example

Classified as panda
x

Small adversarial noise
ϵ

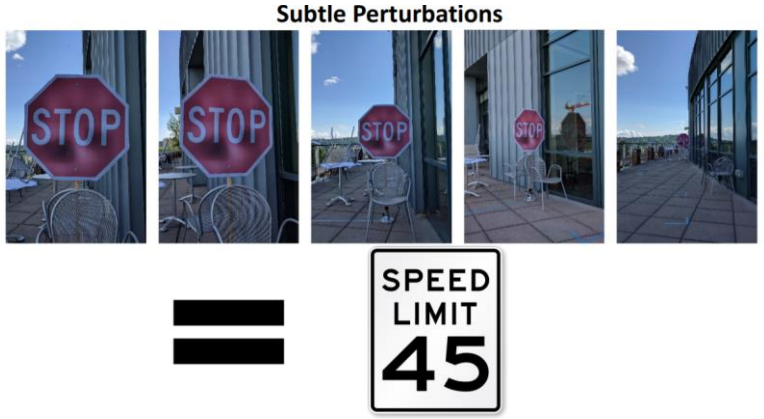
Classified as gibbon
x'

Carefully- calculated noise

Find x' satisfying $||x' - x|| \leq \Delta$
such that $C(x') \neq y$

Implication of Adversarial Examples

- **The existence of adversarial examples prevents the reliable deployment of deep learning models to the real world.**
  - Adversaries may try to actively hack the deep learning models.
  - The model performance can become much worse than we expect.
- **Deep learning models are often not robust.**
  - In fact, it is an active area of research to make these models robust against adversarial examples



Evtimov, Ivan, Kevin Eykholt, Earlence Fernandes, Tadayoshi Kohno, Bo Li, Atul Prakash, Amir Rahmati, and Dawn Song. "Robust Physical-World Attacks on Machine Learning Models." *arXiv preprint arXiv:1707.08945* (2017).

https://seclab.stanford.edu/AdvML2017/slides/dawn-stanford-ai-security-workshop-short-sep-2017.pdf

Do Graph Neural Networks Suffer the Same Problem?

- **Premise**: Common applications of GNNs involve **public platforms** and **monetary interests**.
  - Recommender systems
  - Social networks
  - Search engines
- Adversaries **have the incentive to** manipulate input graphs and hack GNNs' predictions.

Adversarial Attacks on GNN

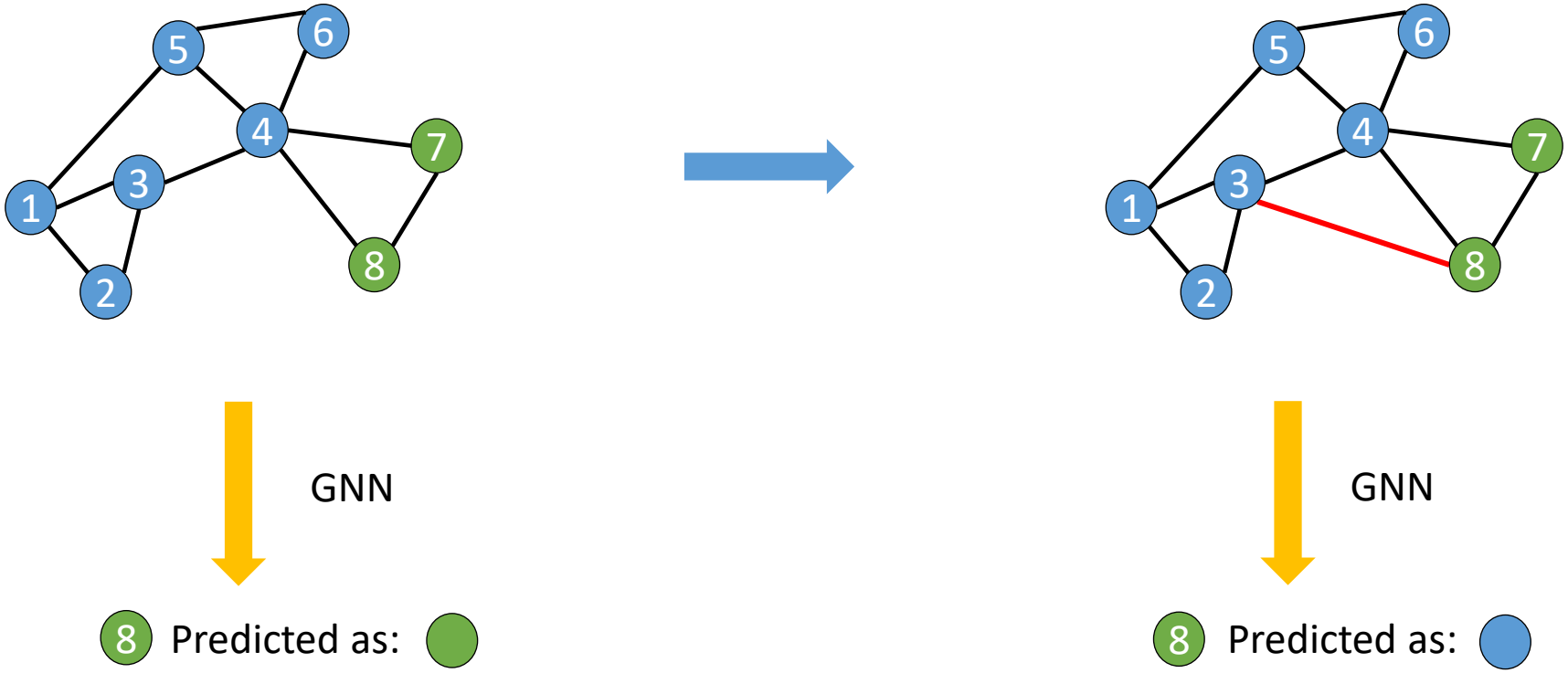
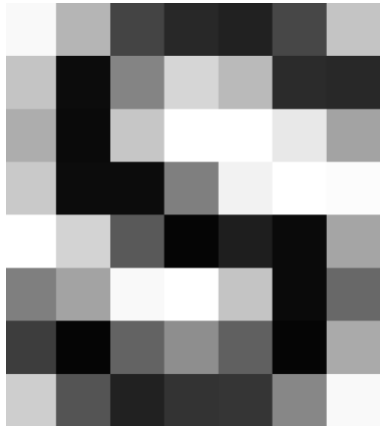
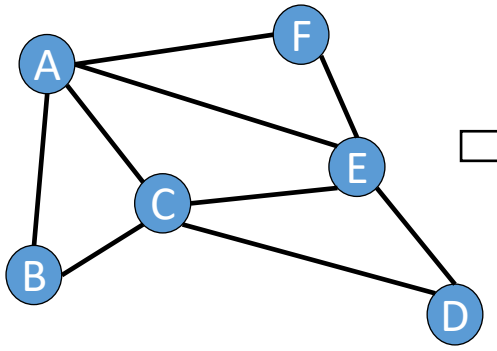


Image vs Graph

- Discreteness
- Perturbation Measure
- Perturbation Type



$$\begin{bmatrix} 0.98, & 0.71, & 0.27, & 0.16, & 0.13, & 0.28, & 0.77 \\ 0.77, & 0.05, & 0.52, & 0.84, & 0.73, & 0.17, & 0.16 \\ 0.68, & 0.04, & 0.78, & 1.00, & 1.00, & 0.91, & 0.64 \\ 0.79, & 0.05, & 0.05, & 0.50, & 0.95, & 1.00, & 0.99 \\ 1.00, & 0.83, & 0.35, & 0.02, & 0.12, & 0.04, & 0.65 \\ 0.50, & 0.64, & 0.98, & 1.00\ , & 0.77, & 0.04, & 0.41 \\ 0.24, & 0.02, & 0.39, & 0.56, & 0.38, & 0.02, & 0.67 \\ 0.81, & 0.33, & 0.13, & 0.20, & 0.21, & 0.53, & 0.98 \end{bmatrix}$$

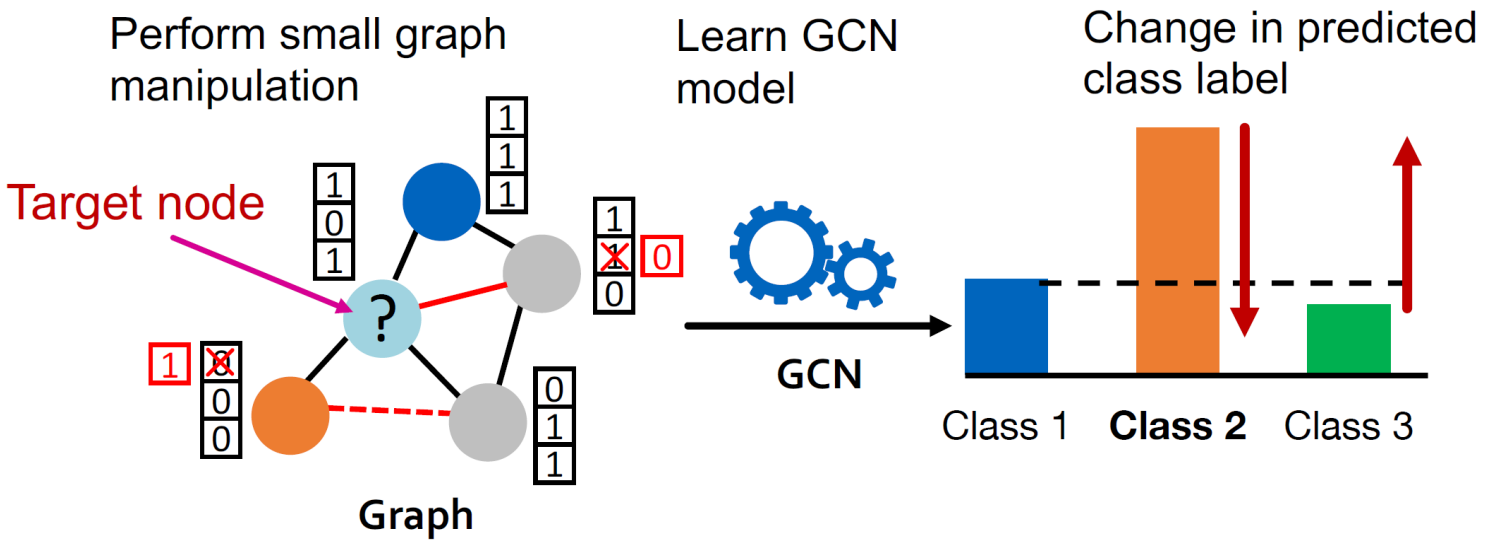


	A	B	C	D	E	F
A	0	1	1	0	1	1
B	1	0	1	0	0	0
C	1	1	0	1	1	0
D	0	0	1	0	1	0
E	1	0	1	1	0	0
F	1	0	0	0	1	0

Formulation

If graph manipulation is too large, it will easily be detected. Successful attacks should change the target prediction with "unnoticeably-small" graph manipulation.

- **Objective for the attacker**:
  - **Maximize** (**change of target node label prediction**)
  - **Subject to** (**graph manipulation is small**)



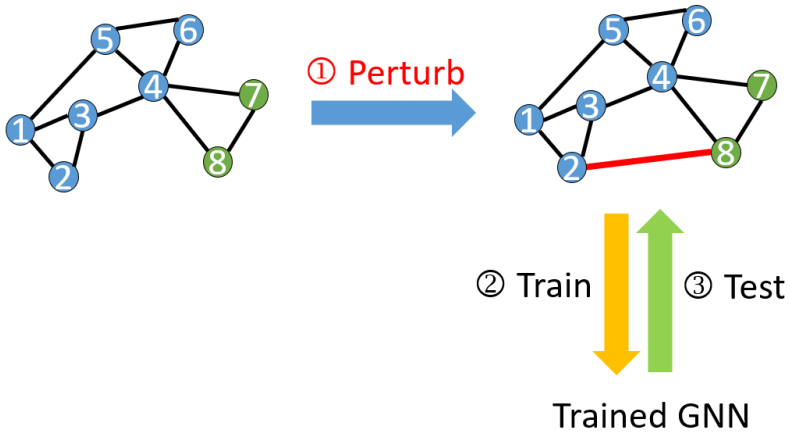
Formulation

- **Original graph**:
  - $\boldsymbol{A}$: adjacency matrix, $\boldsymbol{X}$: feature matrix
- **Manipulated graph (after adding noise):**
  - $\boldsymbol{A}'$: adjacency matrix, $\boldsymbol{X}'$: feature matrix
- **Assumption**: $(\boldsymbol{A}', \boldsymbol{X}') \approx (\boldsymbol{A}, \boldsymbol{X})$
  - Graph manipulation is **unnoticeably small.**
    - Preserving basic graph statistics (e.g,. degree distribution) and feature statistics.
  - Graph manipulation is either **direct** (changing the feature/connection of target nodes) or **indirect**.

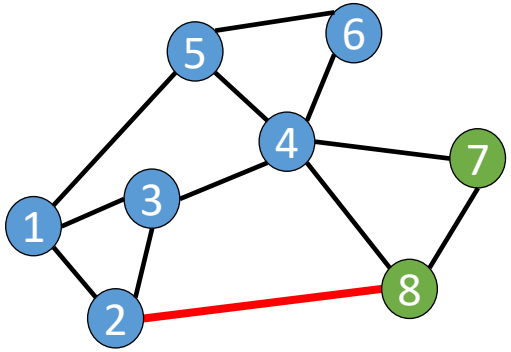
Formulation of Poisoning Attack

- Original adjacency matrix $\boldsymbol{A}$, node features $\boldsymbol{X}$, node labels $\boldsymbol{Y}$.
- $\boldsymbol{\theta}^*$: Model parameter learned over $\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{Y}$.
  - c_v^* : class label of node v predicted by GCN with $\boldsymbol{\theta}^*$
- **An attacker has access to $\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{Y}$, and the learning algorithm.**
- **The attacker modifies $(\boldsymbol{A}, \boldsymbol{X})$ into $(\boldsymbol{A}', \boldsymbol{X}')$.**
- $\boldsymbol{\theta}^{*'}$: Model parameter learned over $\boldsymbol{A}', \boldsymbol{X}', \boldsymbol{Y}$.
  - $c_{v,\boldsymbol{\theta}^{*'}}^{*\prime}$: class label of node v predicted by GCN with $\boldsymbol{\theta}^{*'}$
- The goal of the attacker is to make $c_v^{*\prime} \neq c_v^*$.

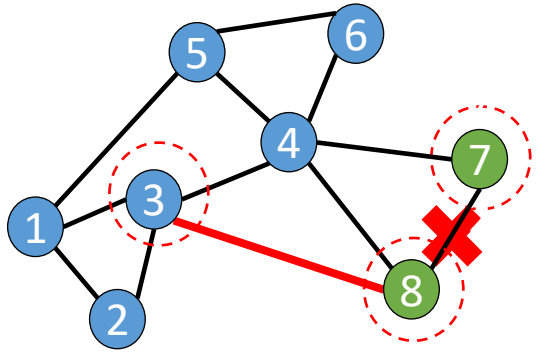
- Poisoning Attack



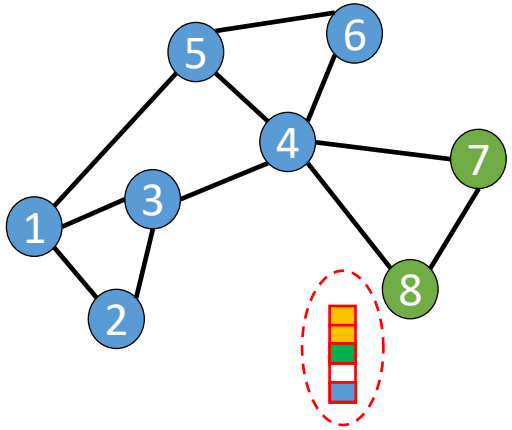
Perturbation Type



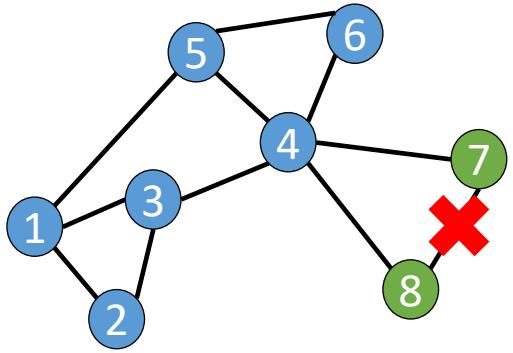
Adding an edge



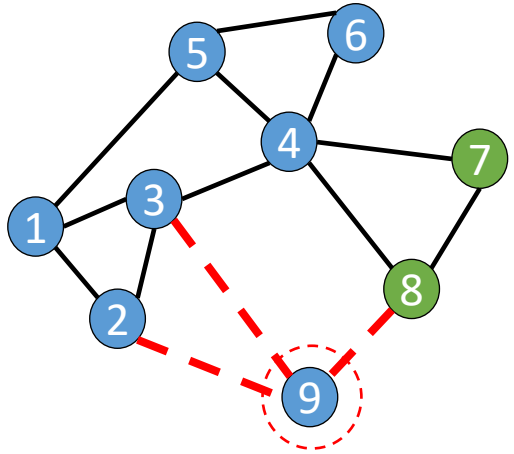
Rewiring



Modifying Features



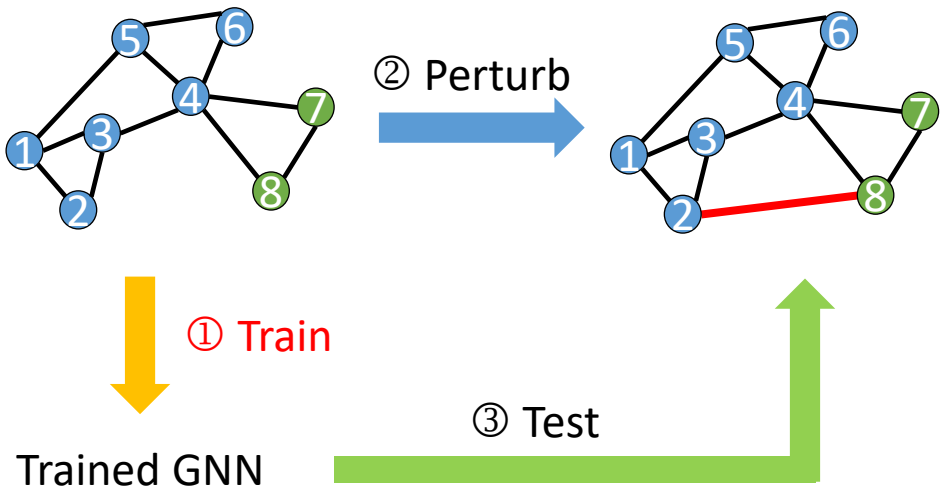
Deleting an edge



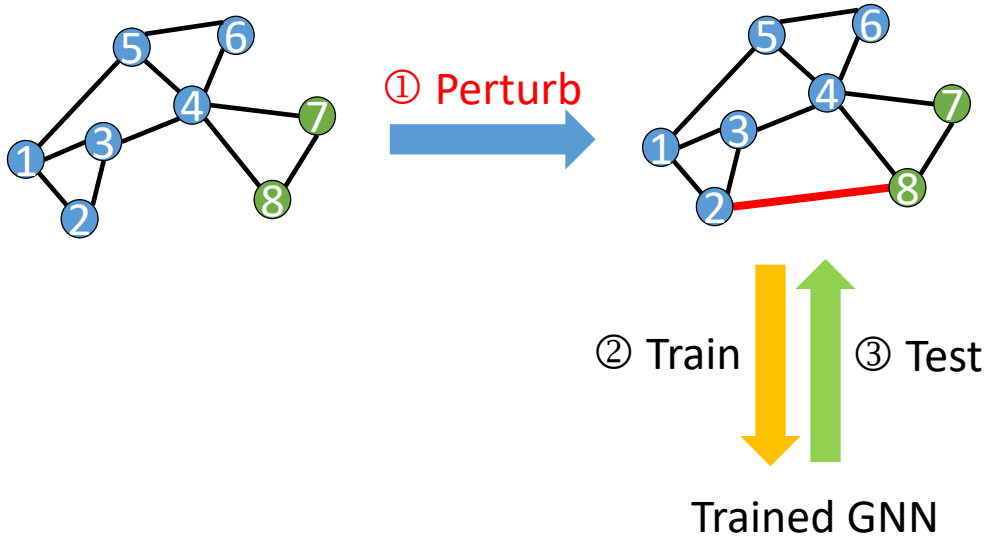
Node Injection

Evasion & Poisoning Attack

- Evasion Attack

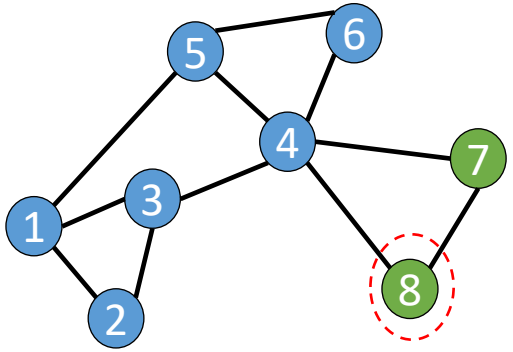


- Poisoning Attack



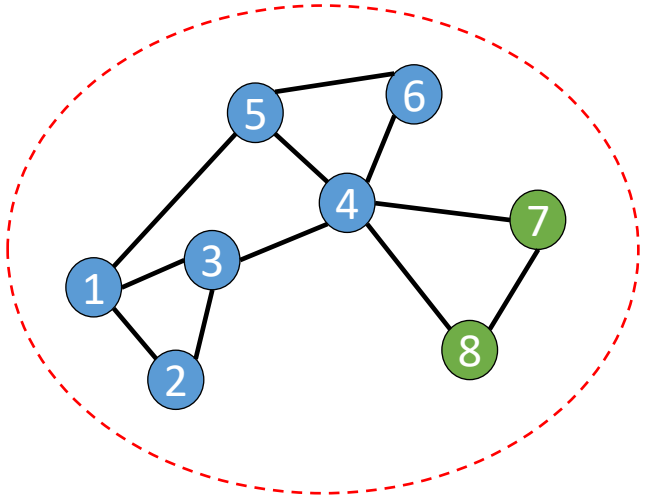
Targeted & Non-Targeted

- Targeted Attackv (Direct)



8 **Target Node**

- Non-Targeted Attack (Indirect)



GradArgmax

- Our Goal:

$$\arg\max_{\hat{A},\hat{X}} \sum_{u \in V_t} \ell\left(f_{\theta^*}\left(\hat{A},\hat{X}\right)_u, y_u\right)$$

$$s.t.\, \theta^* = arg\min_{\theta} \mathcal{L}(f_\theta(A', X'))$$

$$\left|\hat{A} - A\right| + \left|\hat{X} - X\right| < \Delta$$

- Perturbations

  Modifications on A, X

- Evasion: $A' = A,\ X' = X$
- Poisoning: $A' = \hat{A},\ X' = \hat{X}$

- Non-Targeted: $V_t = V_{all}$
- Targeted: V_t is a small subset

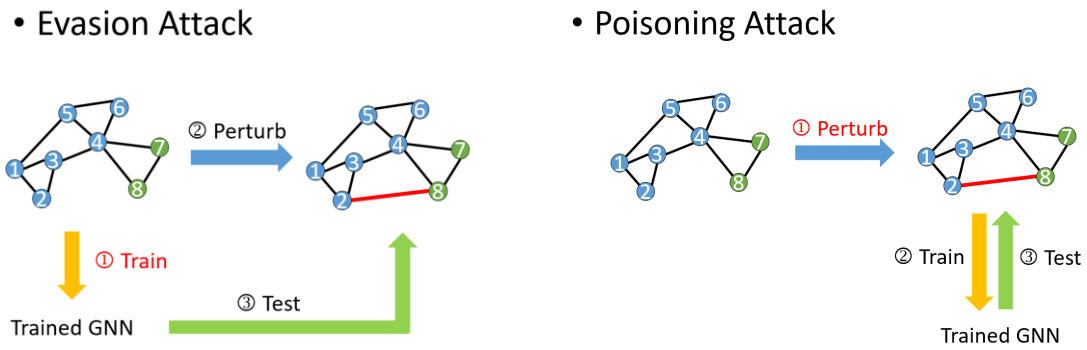
GradArgmax

- Our Goal:

$$\arg\max_{\hat{A},\hat{X}} \sum_{u \in V_t} \ell\left(f_{\theta^*}(\hat{A},\hat{X})_u, y_u\right)$$

$$s.t.\ \theta^* = arg\min_{\theta} \mathcal{L}(f_\theta(A', X'))$$

$$\left|\hat{A} - A\right| + \left|\hat{X} - X\right| < \Delta$$



- Perturbations

  Modifications on A, X

- Evasion: $A' = A,\ X' = X$
- Poisoning: $A' = \hat{A},\ X' = \hat{X}$

- Non-Targeted: $V_t = V_{all}$
- Targeted: V_t is small subset

GradArgmax

- Our Goal:

$$\arg\max_{\hat{A},\hat{X}} \sum_{u \in V_t} \ell\left(f_{\theta^*}\left(\hat{A},\hat{X}\right)_u, y_u\right)$$

$$s.t.\, \theta^* = arg\min_{\theta} \mathcal{L}(f_\theta(A', X'))$$

$$\left|\hat{A} - A\right| + \left|\hat{X} - X\right| < \Delta$$

- Perturbations

  Modifications on A, X

- Evasion: $A' = A,\ X' = X$
- Poisoning: $A' = \hat{A},\ X' = \hat{X}$

- Non-Targeted: $V_t = V_{all}$
- Targeted: V_t is small subset

GradArgmax

- Gradient Ascent:

$$\hat{A} = \hat{A} + \gamma_1 \nabla_A \mathcal{L}(A, X)$$
$$\hat{X} = \hat{X} + \gamma_2 \nabla_X \mathcal{L}(A, X)$$

- Greedy Method

  In each step, choose the perturbation with maximum gradient

- Several approximations are proposed to make the optimization tractable [Zügner et al. KDD2018]

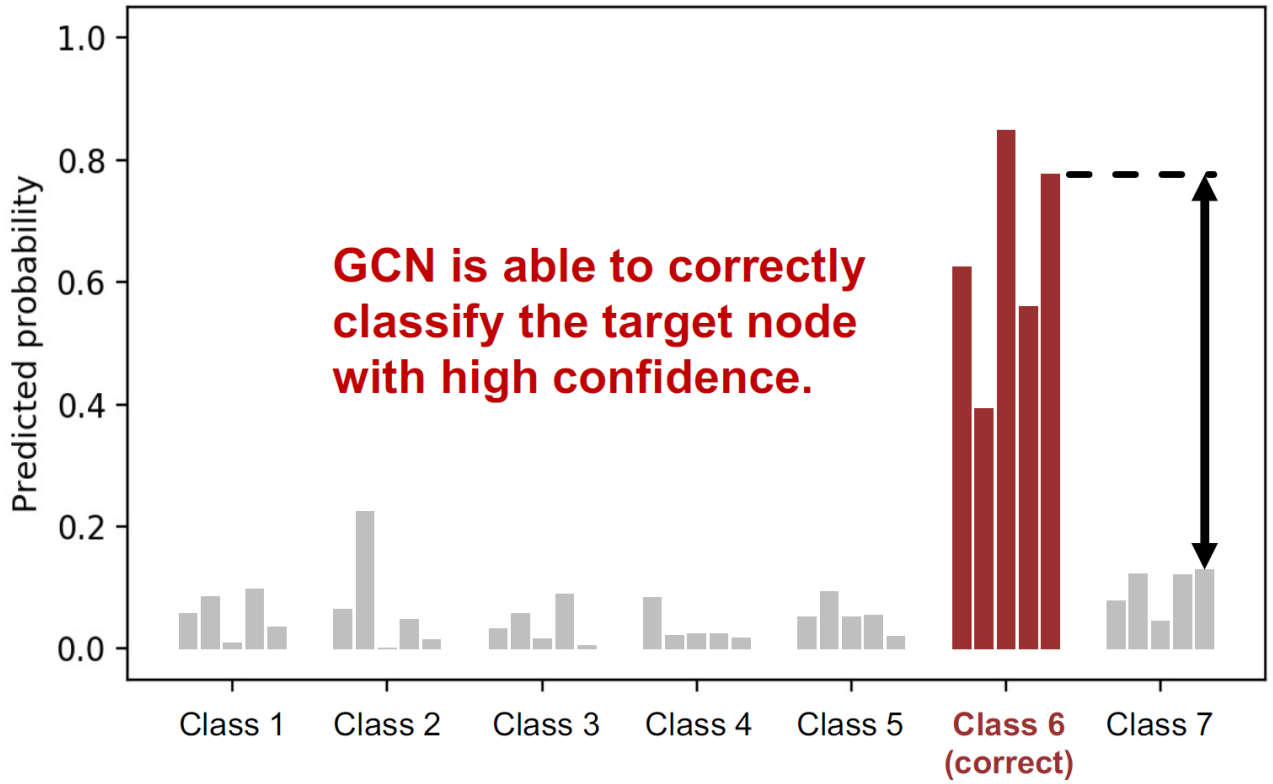
Adversarial Attack on Graph Structured Data. ICML 2018.
Adversarial Attacks on Neural Networks for Graph Data. KDD 2018

Experiments: Setting

- **Setting:** Semi-supervised node classification with GCN
- **Graph:** Paper citation network (2,800 nodes, 8,000 edges).
- **Attack type:** Edge modification (addition or deletion of edges)
- **Attack budget on node v:** d_v + 2 modifications (d_v: degree of node v).
  - **Intuition**: It is harder to attack a node with a larger degree.
- Model is trained and attacked 5 times using different random seeds.

Experiments: Adversarial Attack

Predicted probabilities of a target node v over 5 re-trainings (each bar represents a single trial)
(without graph manipulation, i.e., clean graph)

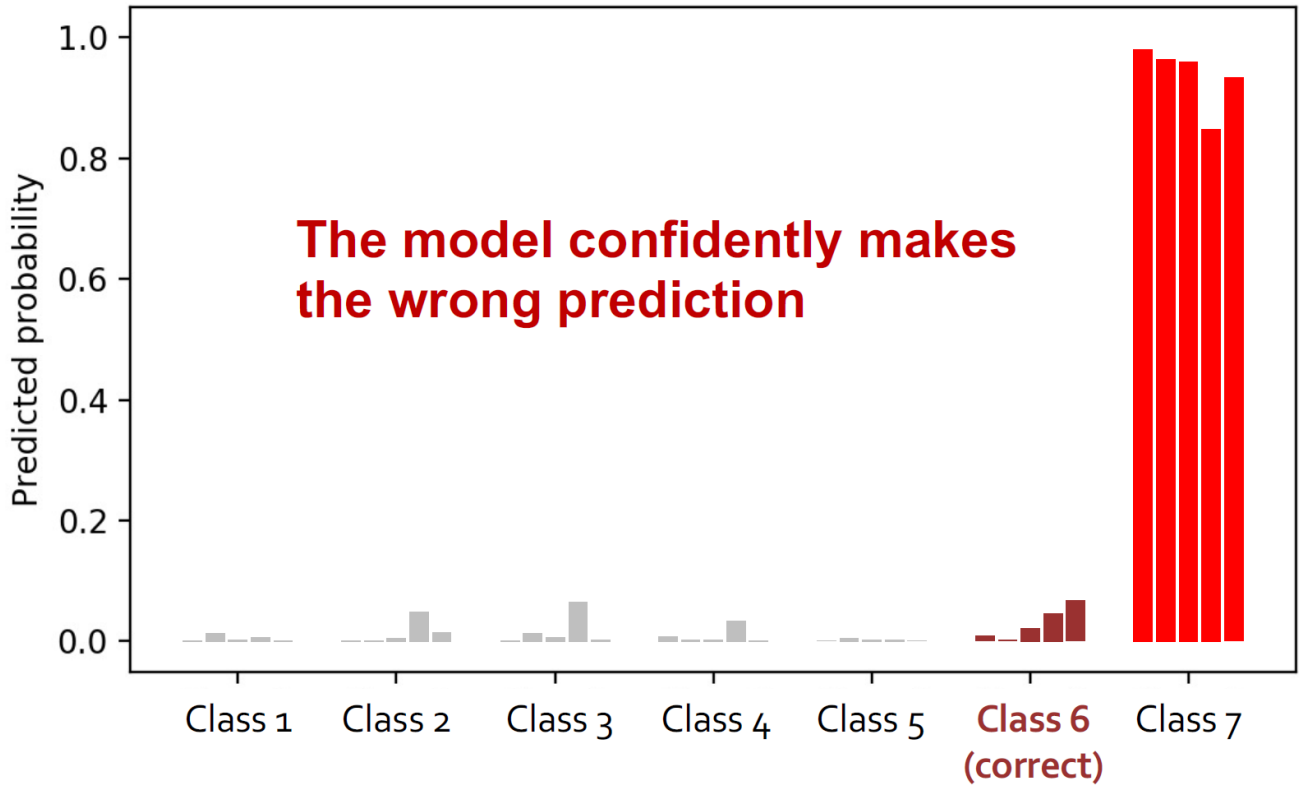


Classification margin
> 0: Correct classification
< 0: Incorrect classification

7-class classification

Experiments: Adversarial Attack

- GCN's prediction after modifying 5 edges attached to the target node (direct adversarial attack).



Predicted probabilities over 5 re-trainings **(with adversarial attacks)**

Nettack

Shortcomings of GradArgmax

- Need to access the model parameters

  Gradient Information

- Perturbation constraint is not enough

$$\left|\hat{A} - A\right| + \left|\hat{X} - X\right| \leq \Delta$$

Adversarial Attacks on Neural Networks for Graph Data. KDD 2018.

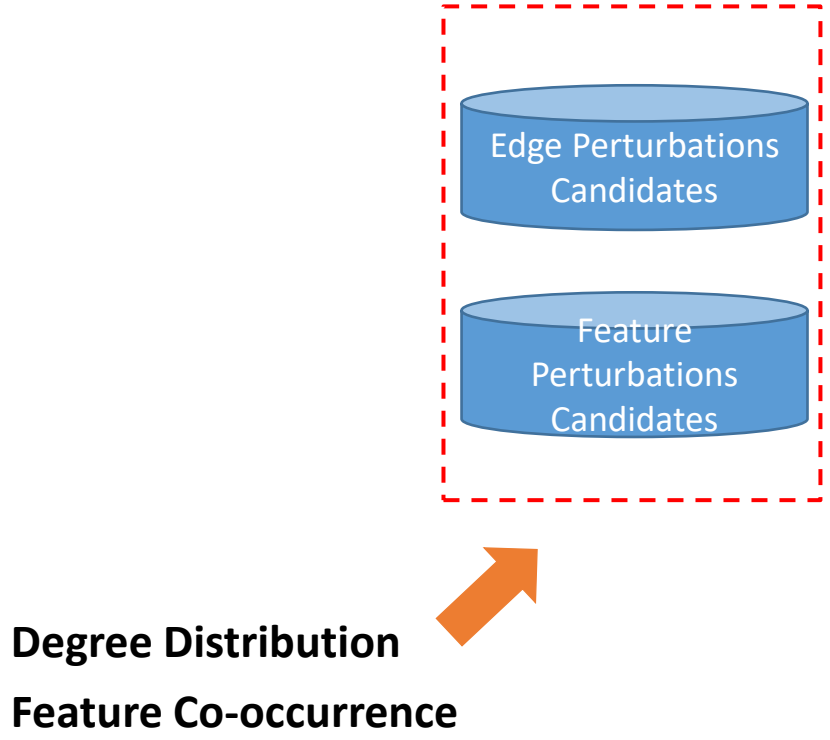
Nettack

- Idea 1: Train a surrogate model

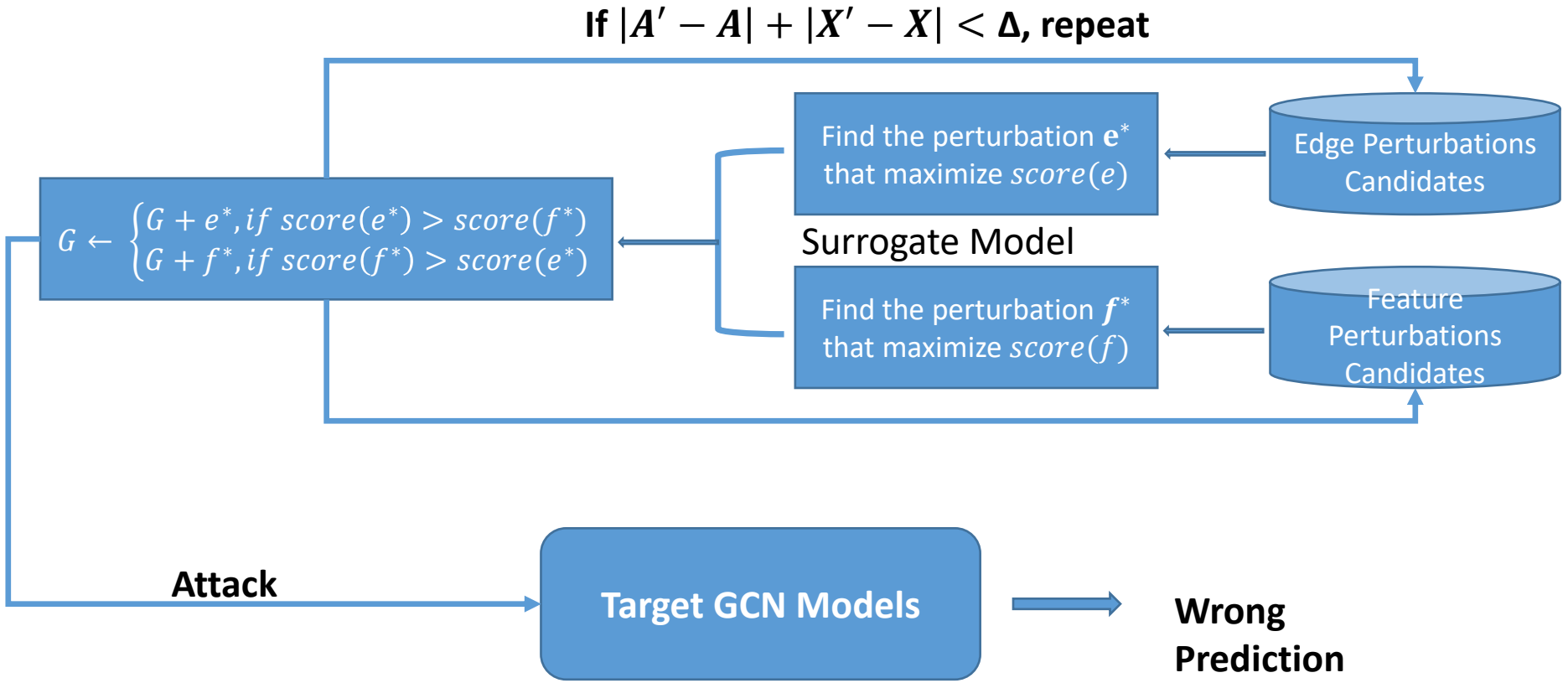
A two-layer linearized GCN trained on original graph

- Idea 2: Perturbation Measure
  - $|A - A'| + |X - X'| < \Delta$
  - Preserving Degree Distribution
  - Preserving Feature Co-occurrence

Nettack


Edge Perturbations Candidates
Feature Perturbations Candidates
Degree Distribution
Feature Co-occurrence

Nettack


If $|A' - A| + |X' - X| < \Delta$, repeat
$G \leftarrow \begin{cases} G + e^*, if\ score(e^*) > score(f^*) \\ G + f^*, if\ score(f^*) > score(e^*) \end{cases}$
Find the perturbation $\mathbf{e}^*$ that maximize $score(e)$
Edge Perturbations Candidates
Surrogate Model
Find the perturbation $\boldsymbol{f}^*$ that maximize $score(f)$
Feature Perturbations Candidates
Attack
Target GCN Models
Wrong Prediction

Defending Against Attacks

- Adversarial Training
- Graph Purifying

Adversarial Training

- Motivation

  Augment the training set with adversarial data

- Main Idea

$$\min_{\theta} \max_{\substack{\delta_A \in \mathcal{P}_A \\ \delta_X \in \mathcal{P}_X}} f_\theta(A + \delta_A, X + \delta_X)$$

Latent Adversarial Training of Graph Convolution Networks. ICML 2019 workshop.

Adversarial Training

- Obstacles
- A is discrete
- X is often discrete

- Hidden Adversarial Training

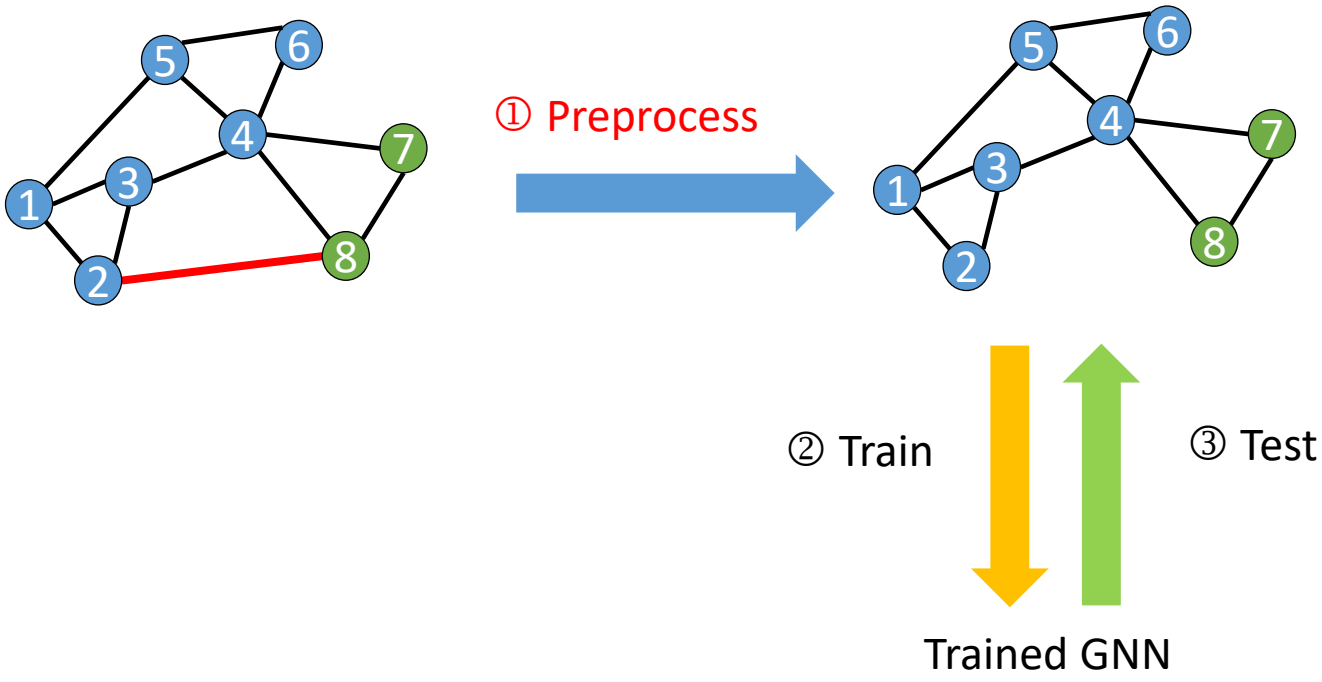
Apply it on the hidden layer !

$$\min_{\theta} \max_{\delta \in \mathcal{P}} f_{\theta}(H^{(1)} + \delta)$$

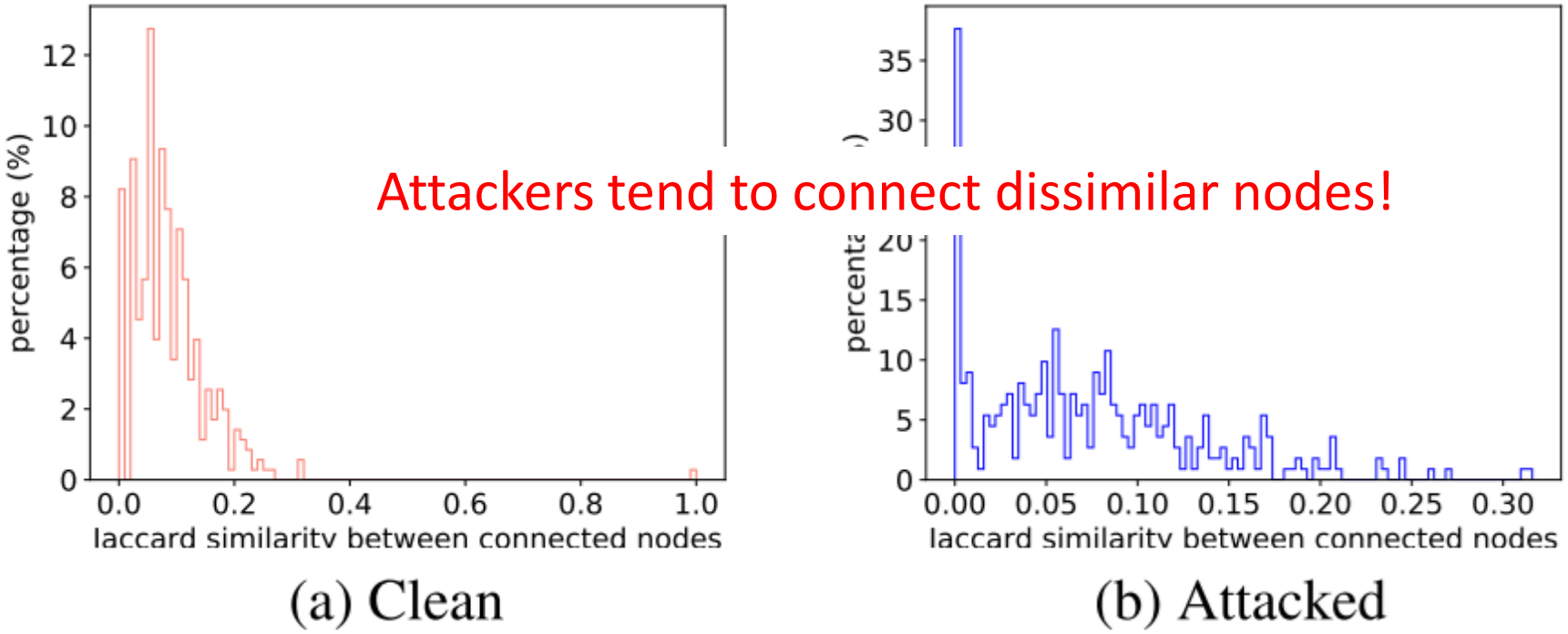
Graph Purifying - Preprocessing

Main Idea

- Purify the poisoned graph
- Train GNN on the purified graph



Graph Purifying - Preprocessing

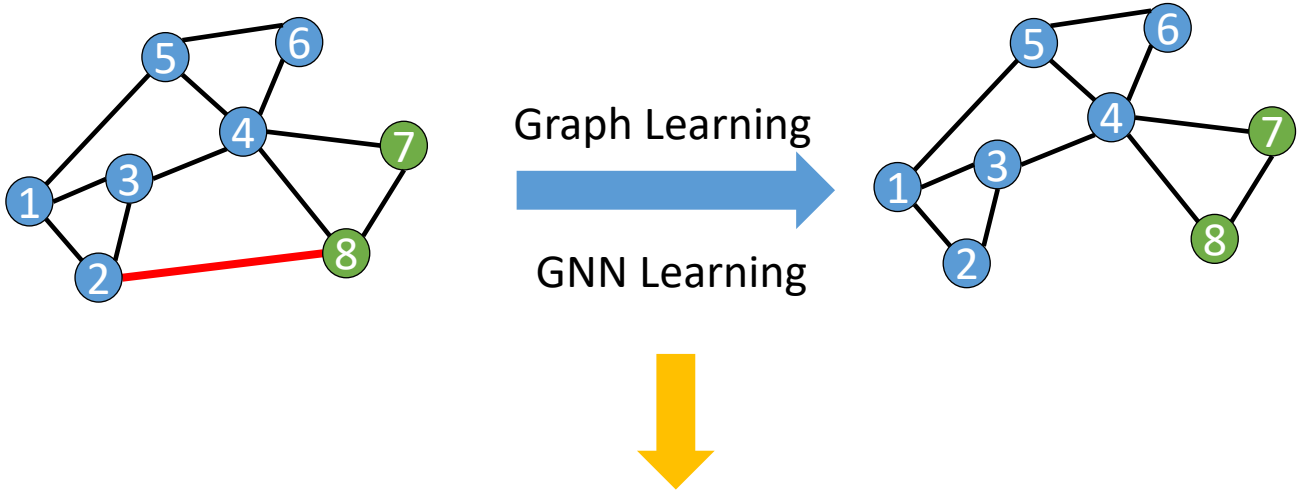
- Observations
  - Attackers favor adding edges than removing edges
- 

Attackers tend to connect dissimilar nodes!

Adversarial Examples on Graph Data: Deep Insights into Attack and Defense. IJCAI 2019.

Graph Purifying – Graph Learning: Pro-GNN

- Graph Learning and GNN training



$$\arg\min_{\mathbf{S}\in\mathcal{S},\theta} \mathcal{L} = \mathcal{L}_0 + \lambda\mathcal{L}_s + \gamma\mathcal{L}_{GNN} \tag{9}$$

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

$$s.t. \qquad \mathbf{S} = \mathbf{S}^\top,$$

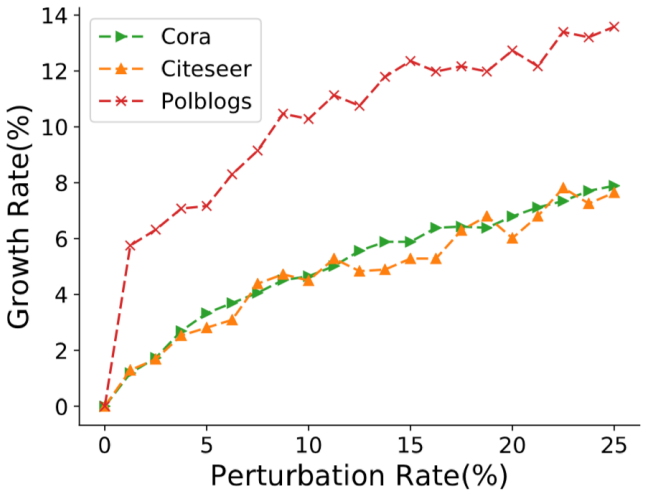
Graph Structure Learning for Robust Graph Neural Networks. KDD 2020.

Pro-GNN: Defend Against Adversarial Attacks

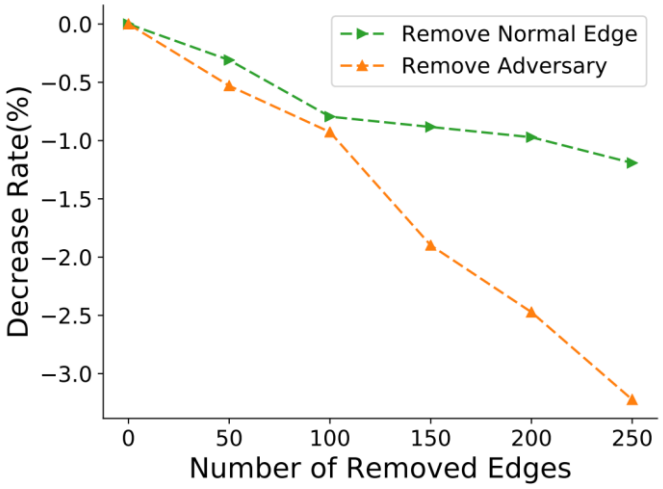
$$\arg\min_{\mathbf{S}\in\mathcal{S}} \mathcal{L}_0 = \|\mathbf{A}-\mathbf{S}\|_F^2 + \alpha\|\mathbf{S}\|_1 + \beta\|\mathbf{S}\|_*,\ s.t., \mathbf{S} = \mathbf{S}^\top$$

Graph Properties

- Low-rank
  - $||\mathbf{S}||_*$ is the nuclear norm of the matrix, which can enforce the matrix to be low-rank
- Sparsity
- Feature smoothness



(b) Rank Growth



(c) Rank Decrease Rate

Figure (b) illustrates that metattack quickly increases the rank of adjacency matrix

(c) chows that removing adversarial edges reduces the rank faster than removing normal edges as demonstrated

Pro-GNN: Defend Against Adversarial Attacks

$$\arg\min_{S \in \mathcal{S}} \mathcal{L}_0 = \|A - S\|_F^2 + \alpha\|S\|_1 + \beta\|S\|_*, \; s.t., S = S^\top$$

r denotes perturbation rate
0% perturbation indicates the original clean graph

Graph Properties

- Low-rank
- Sparsity
  - $||S||_1$ is the L1 norm regularization of the matrix which can enforce it to be sparse
- Feature smoothness

Dataset	r(%)	edge+	edge-	edges	ranks	clustering coefficients
Cora	0	0	0	5069	2192	0.2376
	5	226	27	5268	2263	0.2228
	10	408	98	5380	2278	0.2132
	15	604	156	5518	2300	0.2071
	20	788	245	5633	2305	0.1983
	25	981	287	5763	2321	0.1943
Citeseer	0	0	0	3668	1778	0.1711
	5	181	2	3847	1850	0.1616
	1	341	25	3985	1874	0.1565
	15	485	65	4089	1890	0.1523
	20	614	119	4164	1902	0.1483
	25	743	174	4236	1888	0.1467
Polblogs	0	0	0	16714	1060	0.3203
	5	732	103	17343	1133	0.2719
	10	1347	324	17737	1170	0.2825
	15	1915	592	18038	1193	0.2851
	20	2304	1038	17980	1193	0.2877
	25	2500	1678	17536	1197	0.2723

- Attackers favor adding edges over deleting edges
- Attacks are likely to increase the rank of the adjacency matrix.
- Attacks are likely to reduce the connectivity of a graph. The clustering coefficients of a perturbed graph decrease with the increase of the perturbation rate.

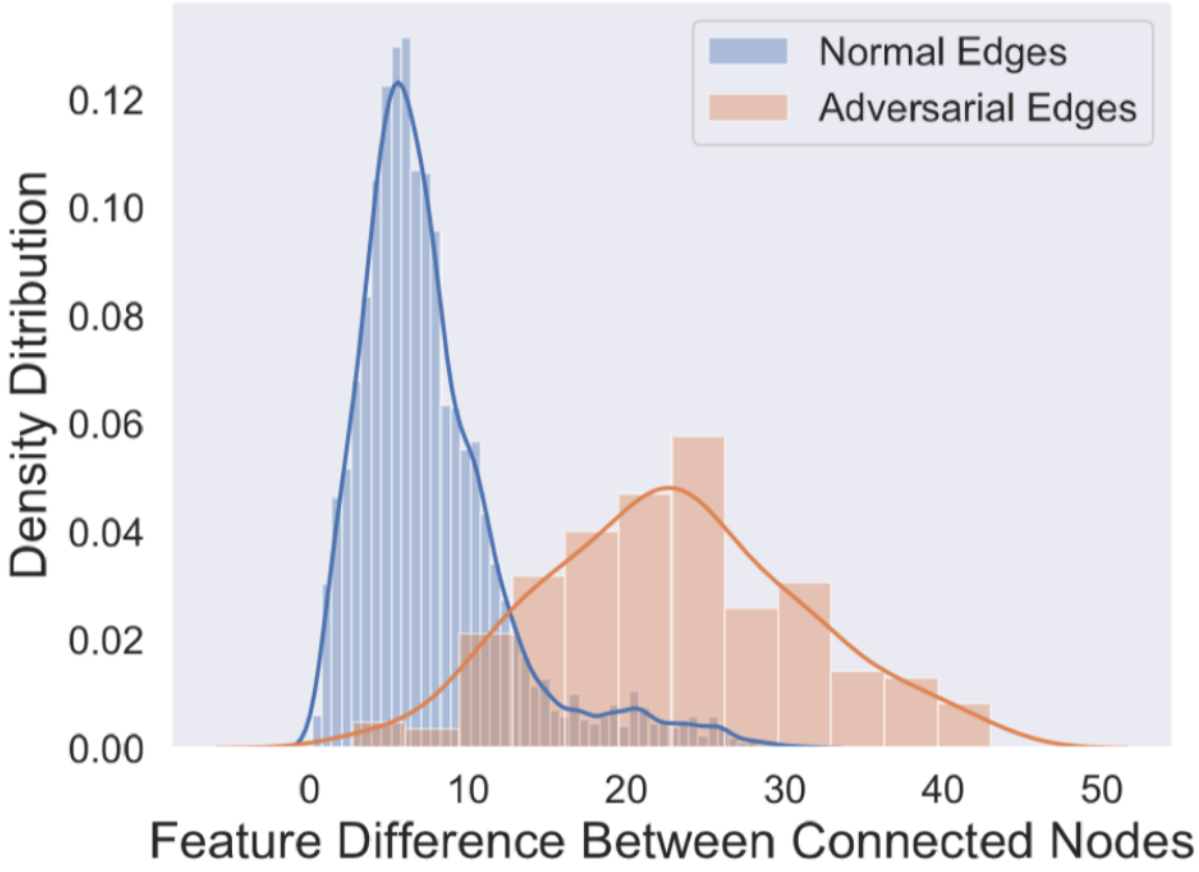
Table Credit: Adversarial Attacks and Defenses on Graphs: A Review and Empirical Study

Pro-GNN: Defend Against Adversarial Attacks

Graph Properties

- Low-rank
- Sparsity
- Feature smoothness

$$\mathcal{L}_s = tr(\mathbf{X}^T \hat{\mathbf{L}} \mathbf{X}) = \frac{1}{2} \sum_{i,j=1}^{N} \mathbf{S}_{ij} \left(\frac{\mathbf{x}_i}{\sqrt{d_i}} - \frac{\mathbf{x}_j}{\sqrt{d_j}} \right)^2$$



(d) Feature Smoothness

Pro-GNN: Framework

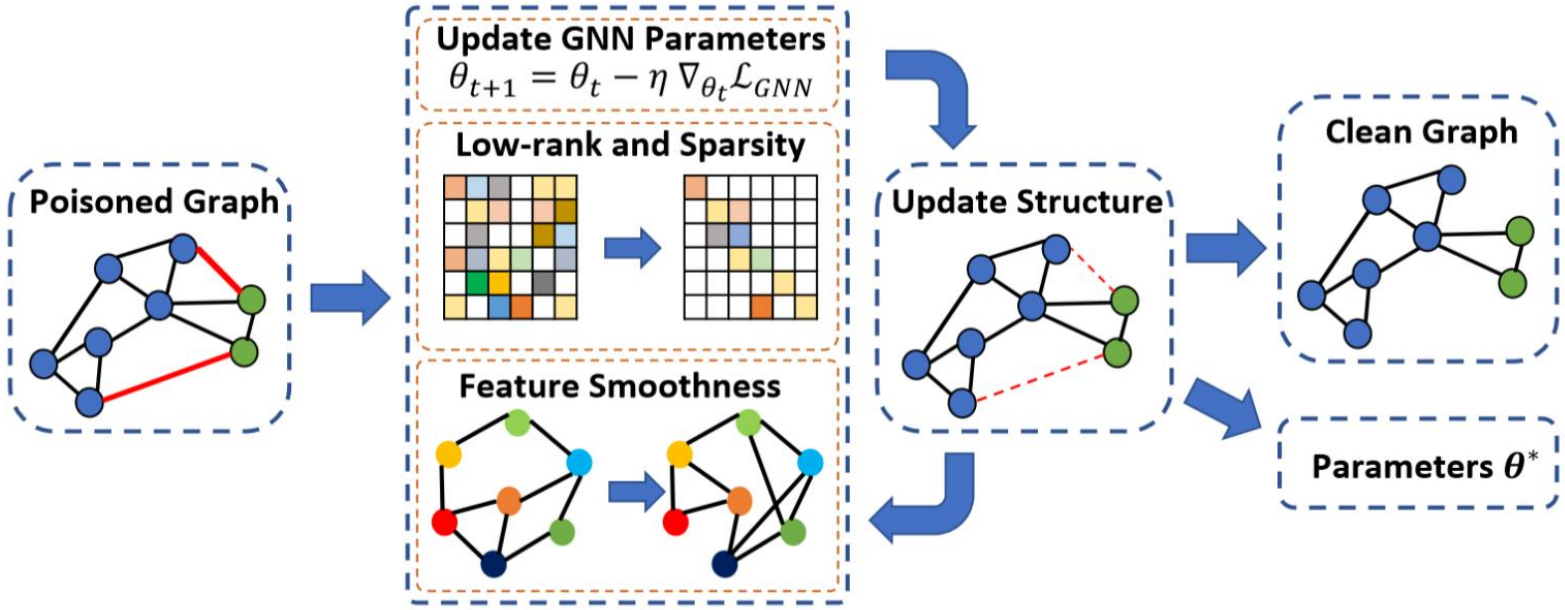


Figure 2: Overall framework of Pro-GNN. Dash lines indicate smaller weights.

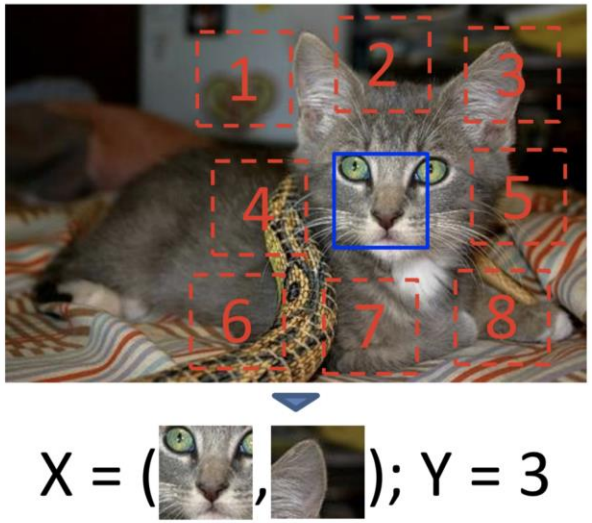
$$\arg\min_{\mathbf{S}\in\mathcal{S},\theta} \mathcal{L} = \mathcal{L}_0 + \lambda\mathcal{L}_s + \gamma\mathcal{L}_{GNN} \tag{9}$$

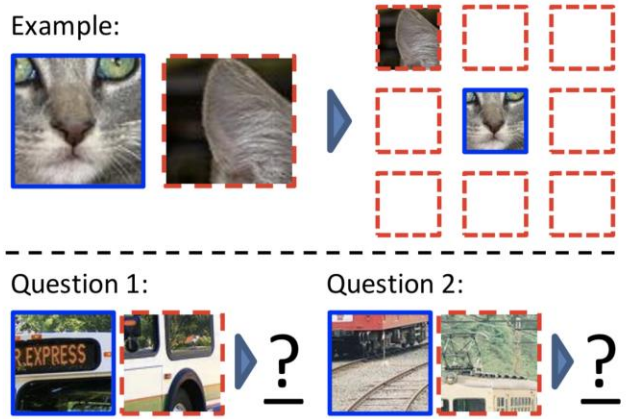
$$= \|\mathbf{A}-\mathbf{S}\|_F^2 + \alpha\|\mathbf{S}\|_1 + \beta\|\mathbf{S}\|_* + \gamma\mathcal{L}_{GNN}(\theta, \mathbf{S}, \mathbf{X}, \mathcal{Y}_L) + \lambda tr(\mathbf{X}^T\hat{\mathbf{L}}\mathbf{X})$$

$$s.t. \qquad \mathbf{S} = \mathbf{S}^\top,$$

Self-Supervised Learning for Graph Neural Networks

Self-Supervised Learning



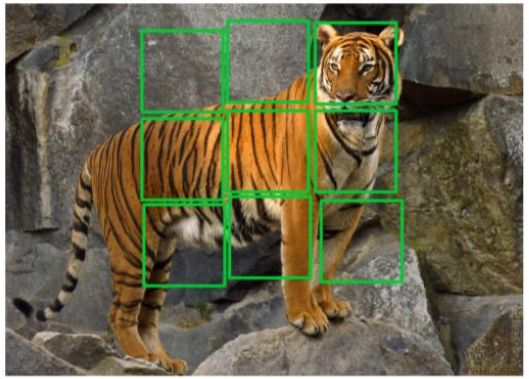


Relative position pretext task

Doersch et al., 2015

Jigsaw puzzle pretext task

Noroozi and Favaro, 2016



(a)

(b) Shuffled

(c) Solved

Applying SSL to Graphs

Similarities to Image and Text Domains

- Nodes have features like images or text

  → Pretext tasks using *attribute information*

- Topological structure associated with unlabeled samples

  → Pretext tasks using *structural information*

Fundamental Differences Found in Graph Domain

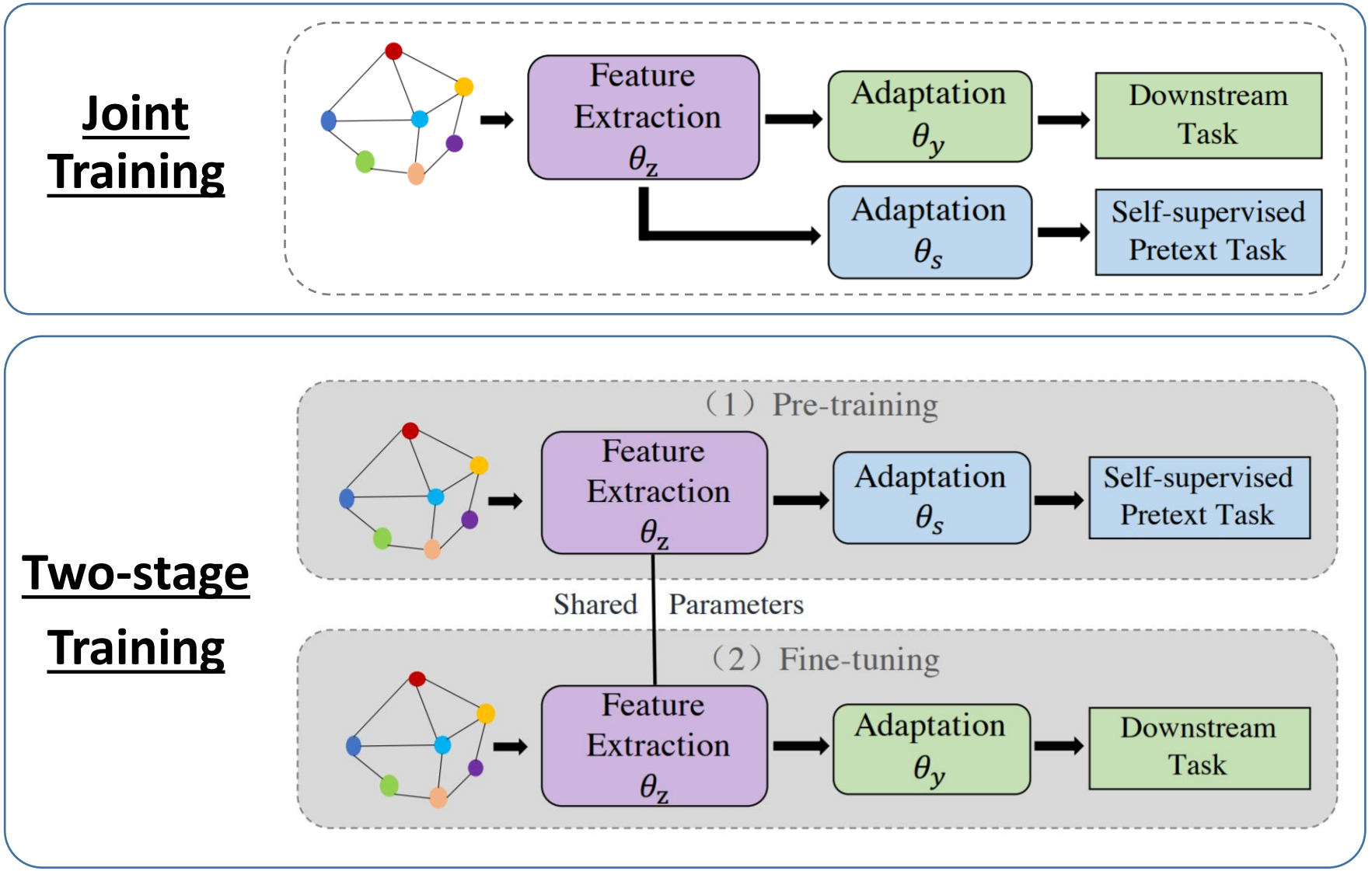
- Nodes are connected and dependent

  → Pretext tasks using *node pairs* or even sets

- Unlabeled nodes have structural relations to labeled nodes

  → Pretext tasks using *label information*

Two Strategies to Merge SSL Tasks with GNN



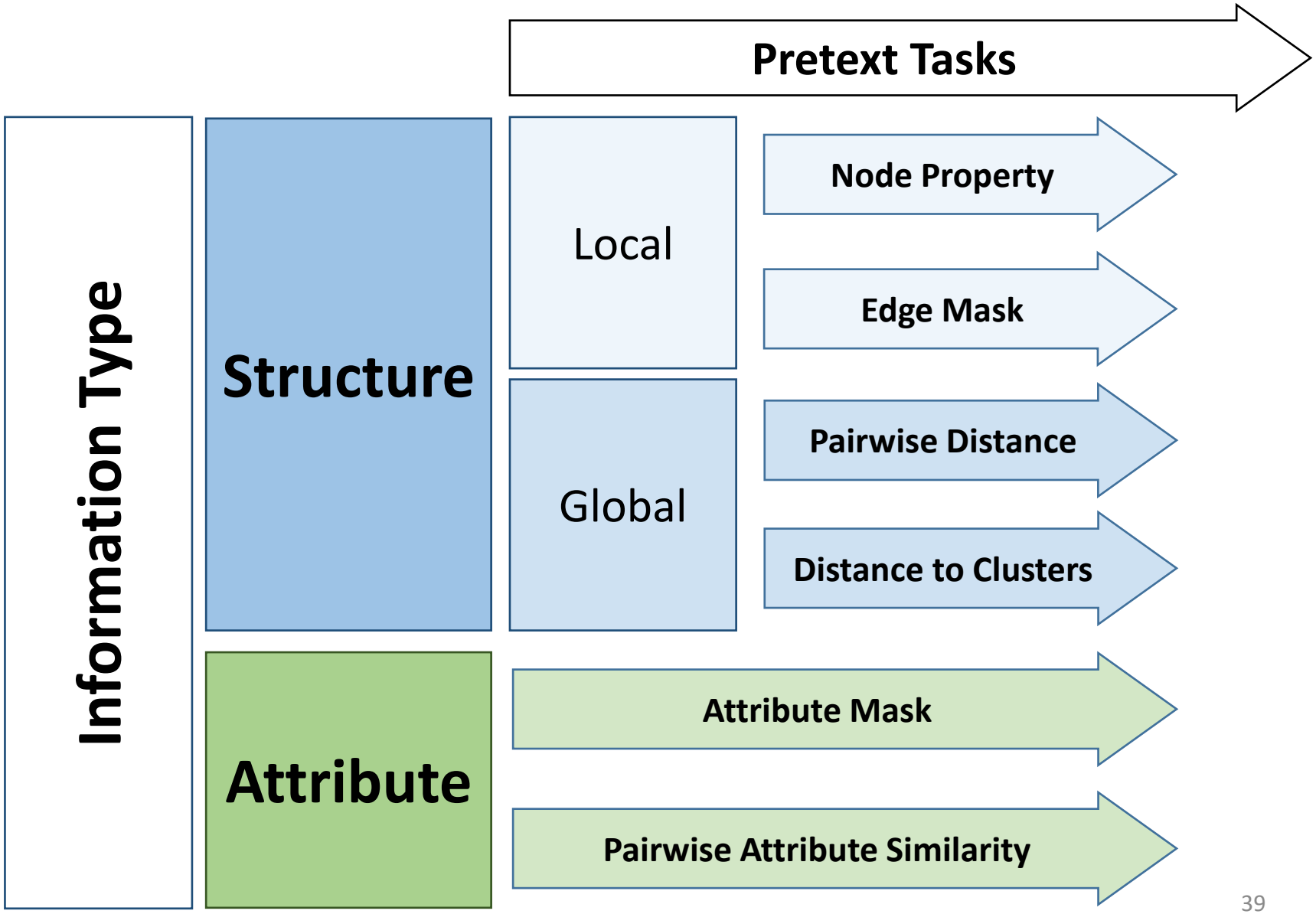
When and Why SSL Works on GNNs

- Presents a set of basic pretext tasks using structure and attribute information

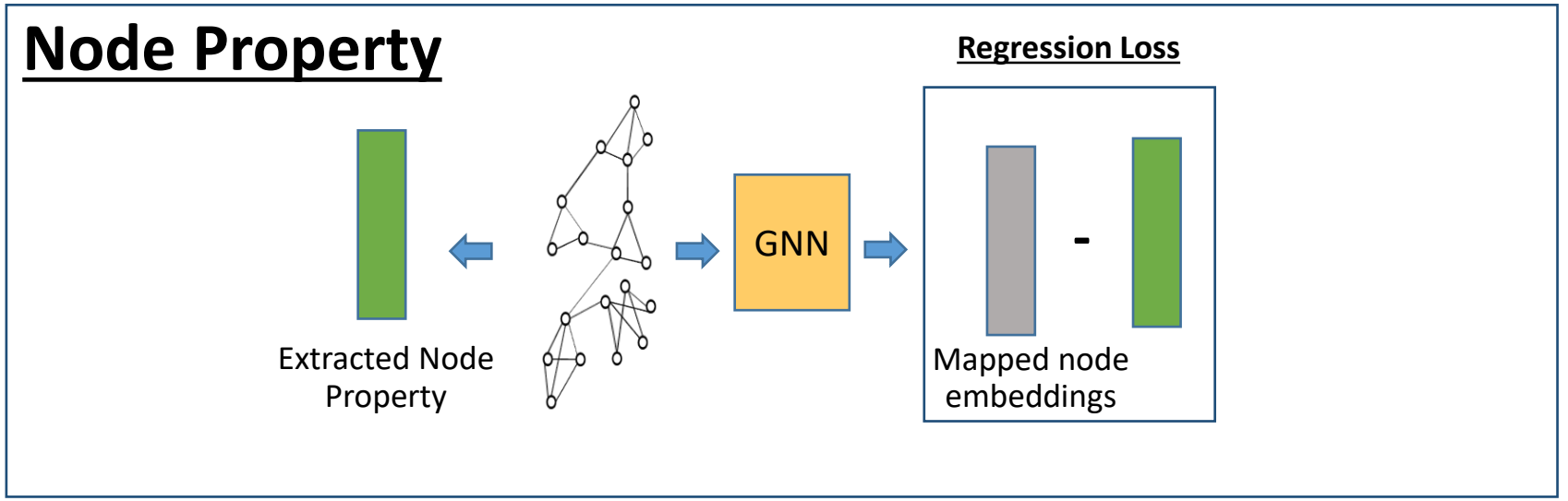
- Insights gained on:
  - Which strategy to harness SSL on GNNs?
  - Why do some pretext tasks work other others do not on GNNs?
  - How to construct advanced pretext tasks beyond basic structure and attributes?

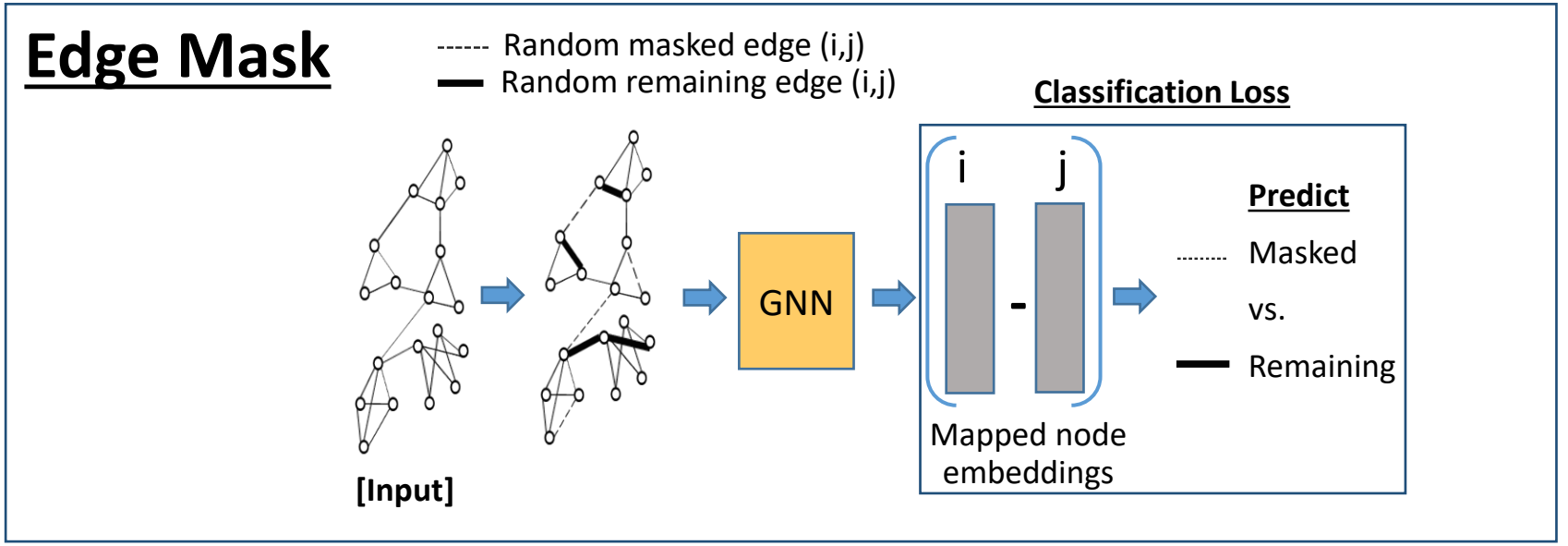
Self-Supervised Learning on Graphs: Deep Insights and New Directions. arXiv. 2020.

Basic Pretext Tasks on Graphs

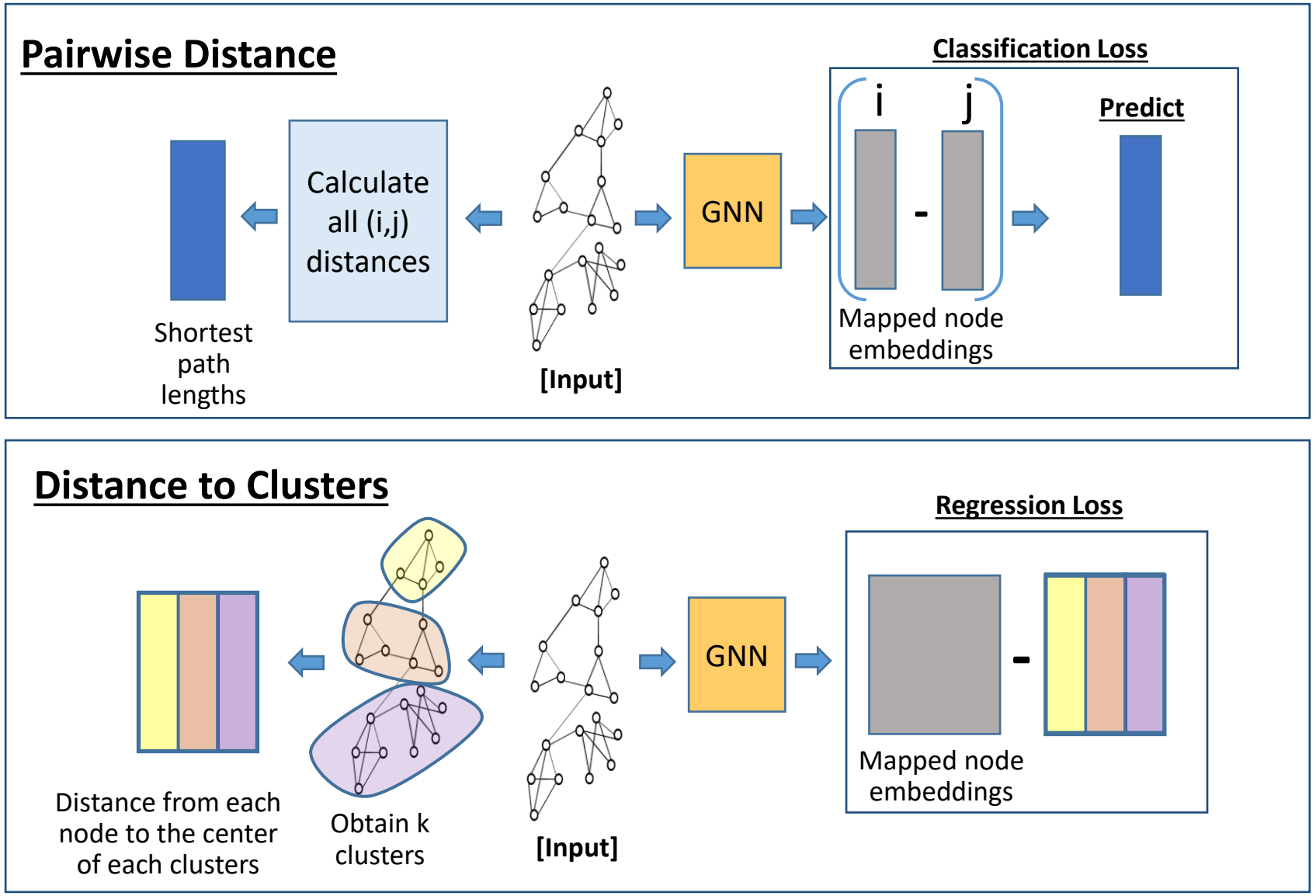


Local Structure Pretext Tasks

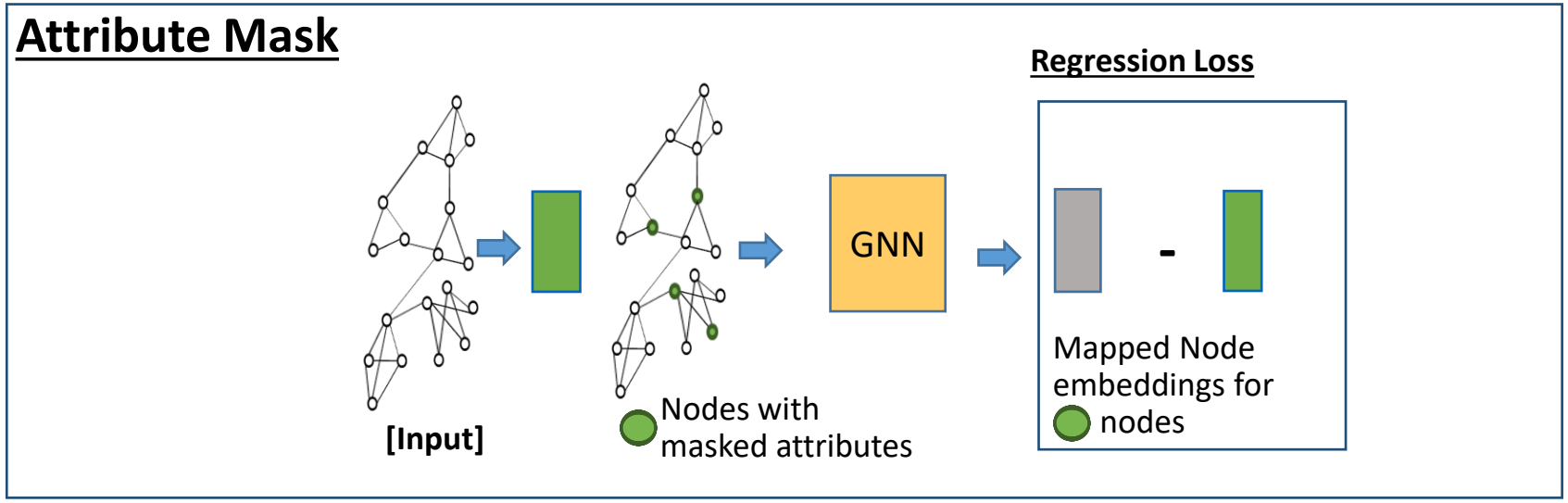


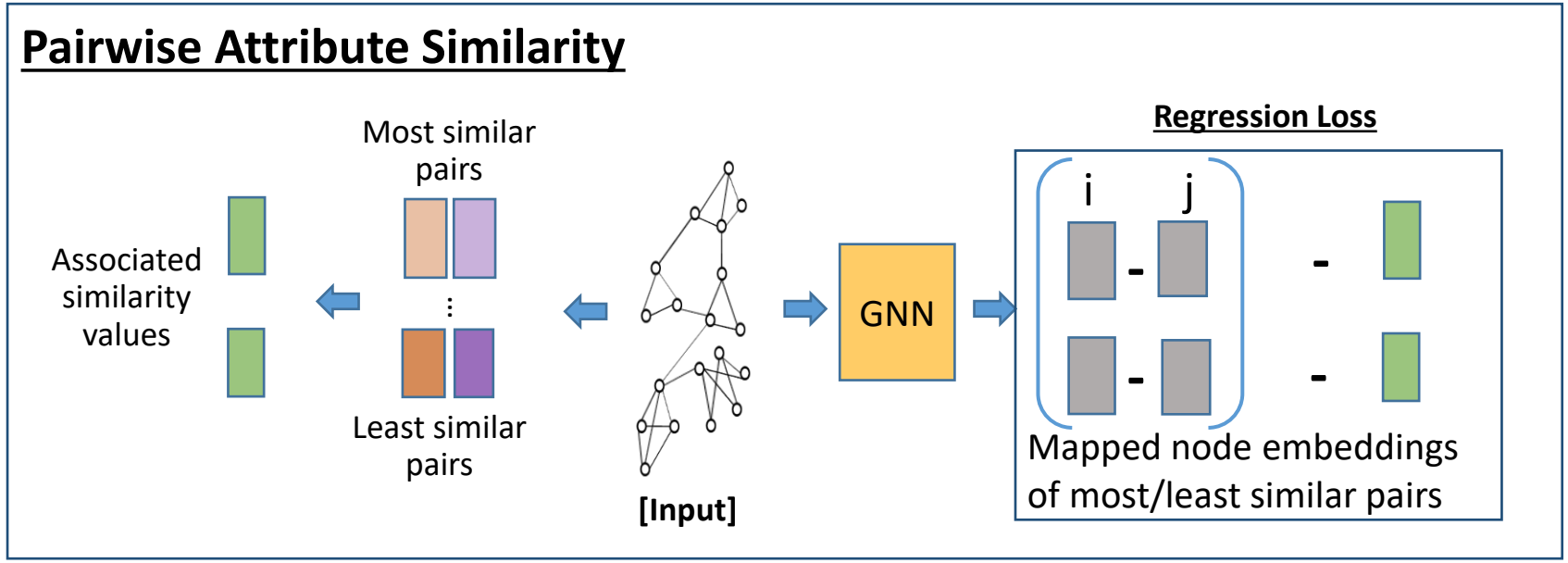


Global Structure Pretext Tasks



Attribute Pretext Tasks





Empirical Study of Basic Pretext Tasks

	Model	Joint Training			Two-stage Training		
		Cora	Citeseer	Pubmed	Cora	Citeseer	Pubmed
	GCN	81.32	71.53	79.28	81.32	71.53	79.28
	GCN-DroppedGraph	81.03	71.29	79.28	81.03	71.29	79.26
	GCN-PCA	81.74	70.38	78.83	81.74	70.38	78.83
Local Structure	NodeProperty	81.94	71.60	79.44	81.59	71.69	79.24
	EdgeMask	81.69	71.51	78.90	81.44	71.57	79.33
Global Structure	PairwiseNodeDistance	83.11	71.90	80.05	82.39	72.02	79.57
	Distance2Cluster	83.55	71.44	79.88	81.80	71.55	79.51
Attribute	AttributeMask	81.47	70.57	78.88	81.31	70.40	78.72
	PairwiseAttrSim	83.05	71.67	79.45	81.57	71.74	79.42

Insights:

- In general joint/multi-task training outperforms pre-training/two-stage training
- Global structure generally outperforms local structure
- Is there a way to further combine and improve these basic methods?

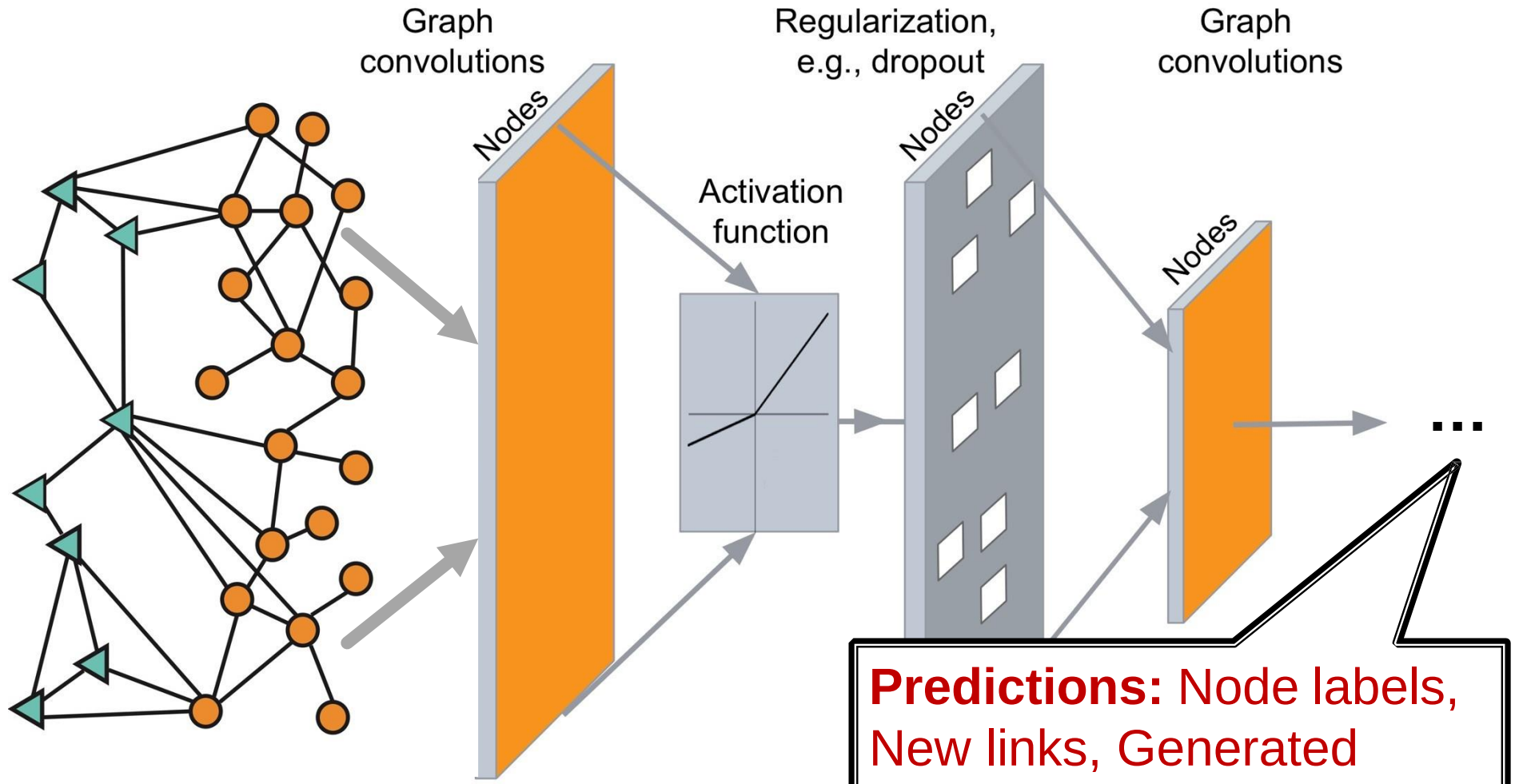
Summary of SSL for GNNs

- SSL for GNNs is still in the early stages but seen rapid growth/interest
- Just as in other domains, not all defined pretext tasks can work
  - Some are more general than others
  - While some can be specifically designed with domain specific knowledge
- Methods have taken a pre-training, self-training, or multi-task training approaches
- Can we further leverage the relation between unlabeled nodes to labeled nodes in advancing pretext tasks?
- Further analysis both theoretically and empirically are desired to better understand when/why/how SSL for GNNs can work

This Course

- **How can we develop neural networks that are much more broadly applicable?**
- Graphs are the new frontier of deep learning
  - How do we take advantage of relational structure for better prediction?

Deep Learing with Graphs



The Bottom Line

- **There is exciting relational structure in many many real-world problems**
  - Molecules/Proteins as strings vs. graphs
  - Travel time duration over the map graph
- **Identifying and harnessing this relational structure leads to better predictions**
  - AlphaFold
  - Biomedicine
  - Recommender systems

What We Have Learned

- Traditional graph-based features and learning algorithms
- Graph convolution
  - Spectral view
  - Spatial view
    - Graph isomorphism view
- Applications
  - Knowledge graphs
  - Recommender systems
- Advanced topics
  - Attack/defence
  - Self-supervised learning

What We Haven’t Introduced

- Graph isomorphism matching and counting
- Knowledge graph reasoning
  - Complex knowledge graph queries
- Community detection
- Graph generation
- Scalability
- …

Following Works

- Project presentations
  - In following five classes
- Project report
  - You can update your model after the presentation
- Final exam
  - We will provide some example papers from my other courses

Thank you everyone!

- The new normal after COVID
  - I am very happy to see everyone in person
- Hope you were interested in what we have introduced
  - The first time offering of this course
- Hope you also enjoy your projects and presentations!