THE WEB CONFERENCE ACM

KAIST

Self-Guided Robust Graph Structure Refinement

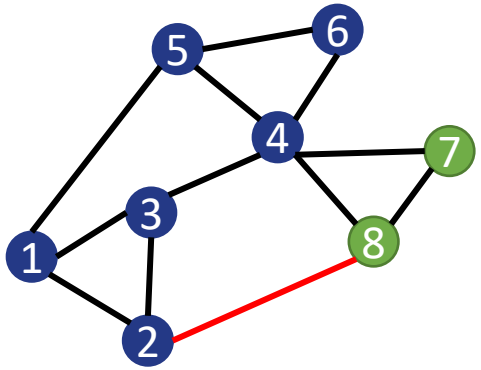
Yeonjun In, Kanghoon Yoon, Kibum Kim, Kijung Shin, Chanyoung Park

Korea Advanced Institute of Science and Technology (KAIST)

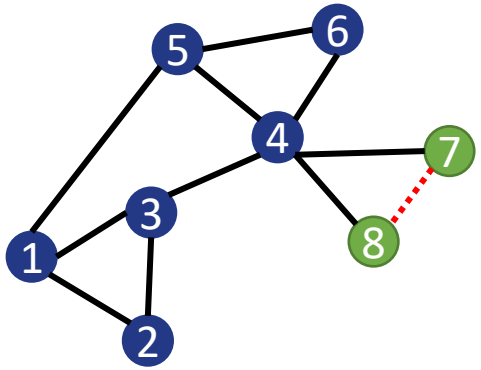
{yeonjun.in, ykhoon08, kb.kim, kijungs, cy.park}@kaist.ac.kr

Adversarial Attacks on Graphs

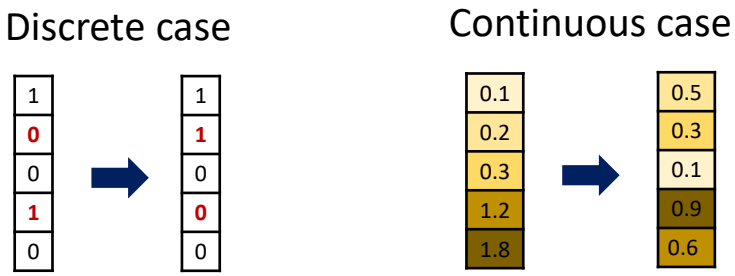
Deep graph learning methods are vulnerable to adversarial attacks !



Adding an edge



Deleting an edge



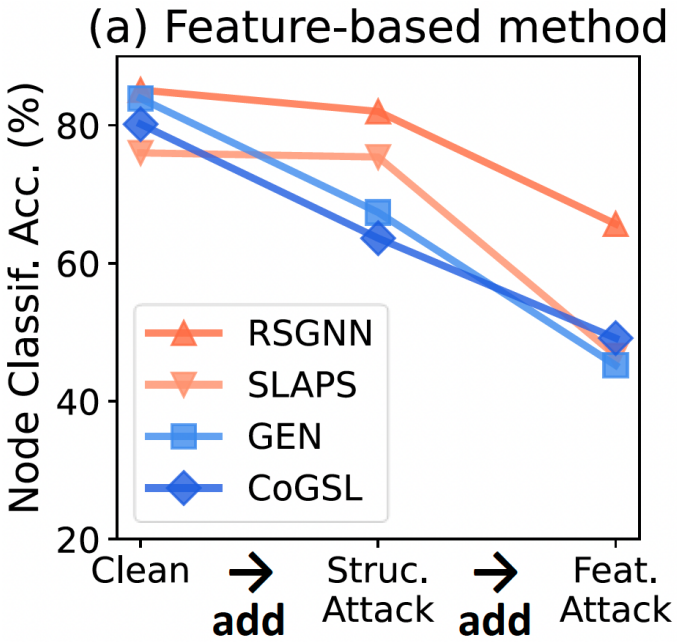
Modifying features

Graph Structure Refinement:

refines a given graph structure by minimizing the impact of adversarial edges.

Graph structure refinement (GSR) methods

(a) Feature-based GSR: *encourage the nodes with similar features to be connected*



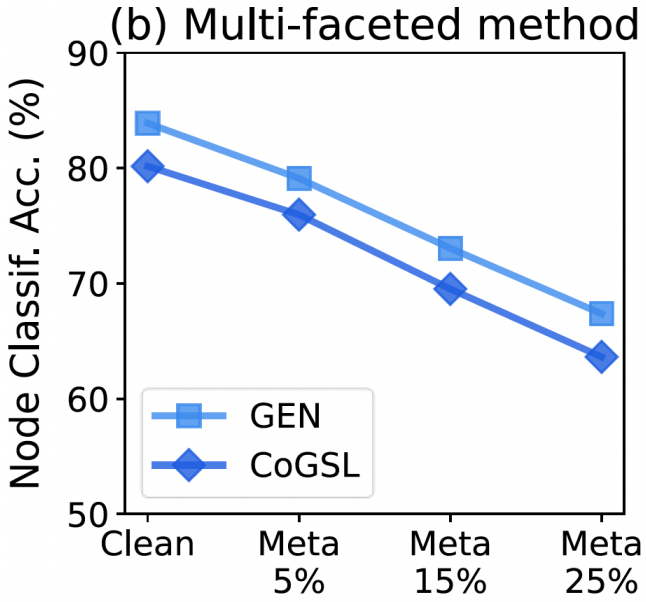
Orange line

→ Performance drops when the node features are attacked.

→ ***Assumption: node features should NOT be attacked.***

Graph structure refinement (GSR) methods

(b) Multi-faceted GSR: *exploit the high-order structural similarity besides feature similarity*



Performance drops as the degree of structure attack increases.

→ ***<u>Assumption: the amount of structure attack should be moderate.</u>***

Existing GSR methods are limited by narrow assumptions!

Assumption 1. node features should NOT be attacked.

Assumption 2. the amount of structure attack should be moderate.

- *Can NOT be always satisfied in real-world applications*
- *Their practicality and applicability are highly limited*

Solution

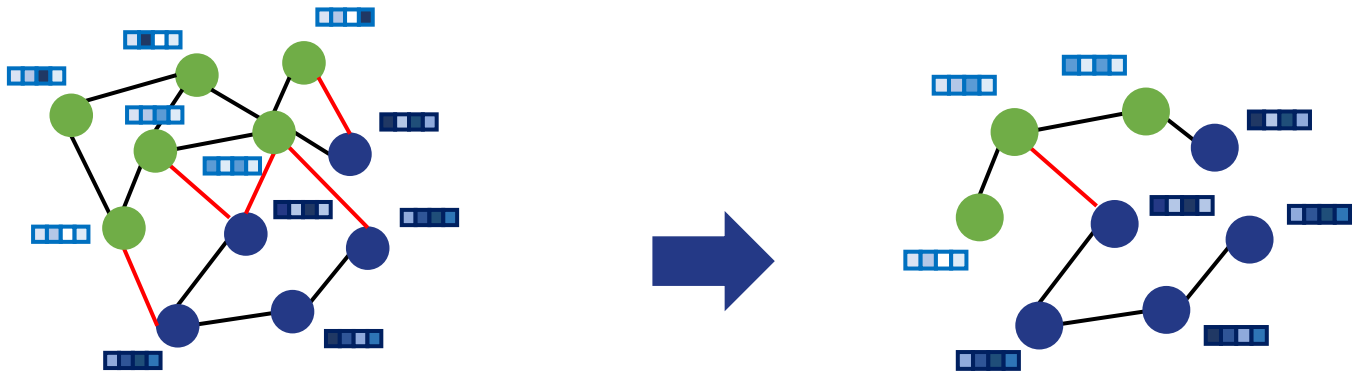
Key idea:

Utilize a clean proxy graph structure in addition to the node feature information.

➡ Free from two narrow assumptions

Where and how to get the clean proxy structure?

Find the clean sub-graph in the given graph itself !



Attacked graph structure **Extracted clean sub-graph**

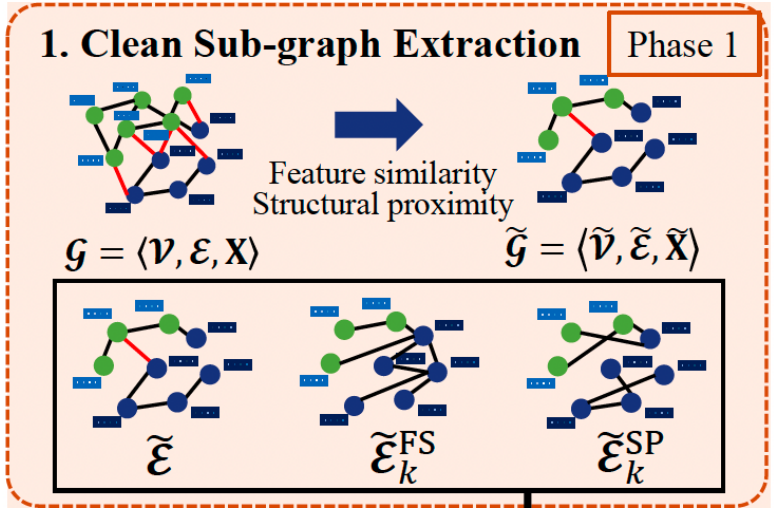
Self-Guided Robust Graph Structure Refinement (SG-GSR)

GSR module: SuperGAT [Kim and Oh]

- Basically, GAT. But, attention weights are estimated by a link predictor.

$$L_{\mathcal{V}} = -\sum_{i \in \mathcal{V}^L} \sum_{c=1}^{C} \mathbf{Y}_{ic} \log \widehat{\mathbf{Y}}_{ic} \qquad L_{\mathcal{E}}^{l} = -\left(\frac{1}{|\mathcal{E}|} \sum_{(i,j) \in \mathcal{E}} \log \phi_{ij}^{l} + \frac{1}{|\mathcal{E}^{-}|} \sum_{(i,j) \in \mathcal{E}^{-}} \log(1 - \phi_{ij}^{l}) \right)$$ ➡ Jointly optimize

Clean sub-graph Extraction



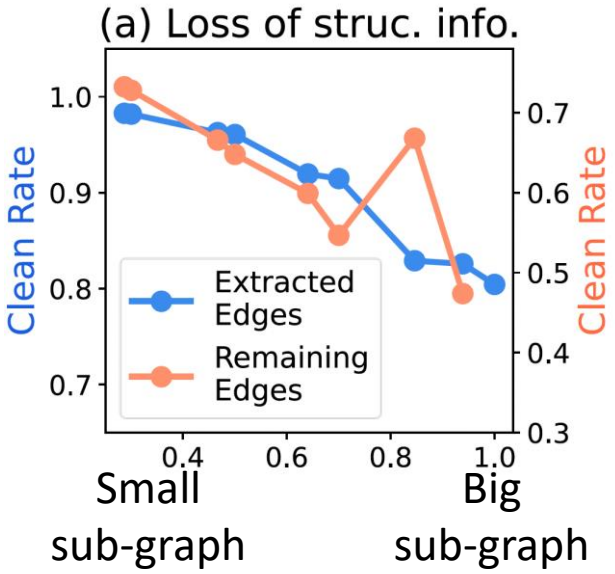
1. Extract edges with BOTH high structural proximity and feature similarity (Done offline)

2. Learn GSR module to minimize a training loss:

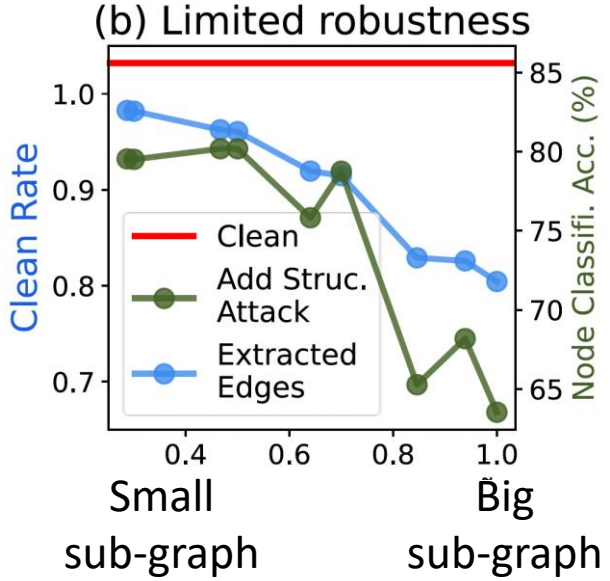
$$\operatorname{argmin} L_{\mathcal{V}} + L_{\tilde{\mathcal{E}}}^{l}$$

Challenges on the extracted sub-graph

- (a) To extract the clean sub-graph, we removed numerous clean edges as well as adv edges.
- (b) The limited structural information hinders the predictive power of GNNs



- Small sub-graph almost has clean edges
- But, the removed edges contain many clean edges



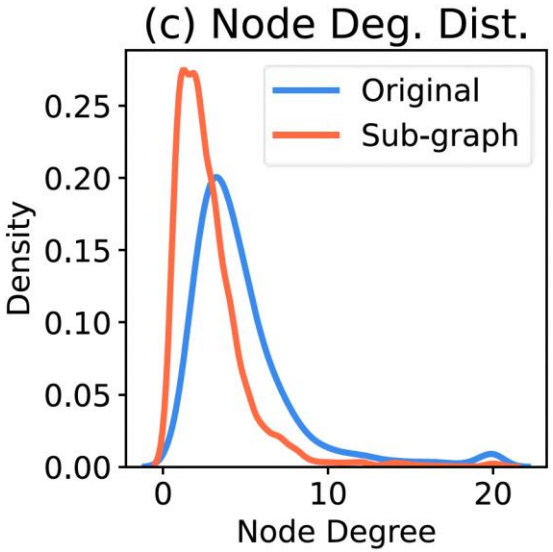
- Performance of GSR trained on clean small sub-graph is limited compared to the upper-bound

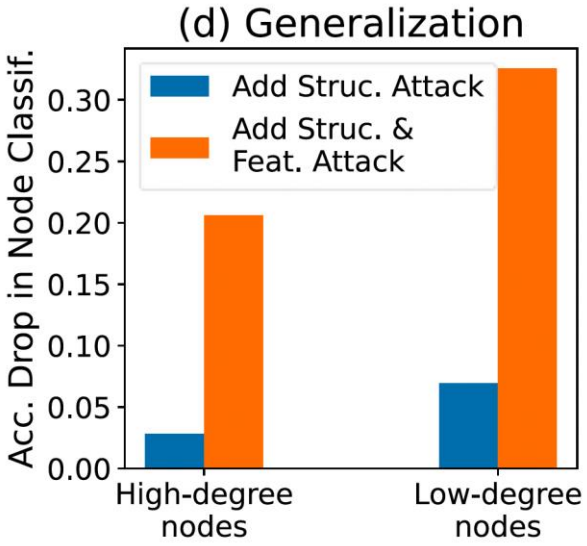
Challenge 1: Loss of Structural Information

Challenges on the extracted sub-graph

- (c) Node deg. distribution of the clean sub-graph **gets more imbalanced**
  - ➢ High degree nodes would dominate the edge set of sub-graph.

- (d) It hinders the generalization ability of GSR module to low-degree nodes.





- Acc drop is more significant in low degree when attack is added

Challenge 2. Imbalanced Node Degree Distribution

Dealing with the Challenges of Sub-graph Extraction

Graph Augmentation:

- *Supplement the structural information.*
- *Add edges important for predicting node labels to the sub-graph.*

- *Three Importance measures*
  1) **Class homophily:** JSD between the class prediction probability of two nodes.
  2) **Feature smoothness:** node feature similarity between two nodes
  3) **Structure proximity:** node embedding similarity between two nodes (node2vec)

$$\tilde{\mathcal{E}}^{\text{aug}} = \tilde{\mathcal{E}} \cup \tilde{\mathcal{E}}^{\text{JSD}} \cup \tilde{\mathcal{E}}^{\text{FS}} \cup \tilde{\mathcal{E}}^{\text{SP}}$$

Sub-graph | High class hom. edges | High feat. smooth. edges | High struc. prox. edges

Dealing with the Challenges of Sub-graph Extraction

Graph Augmentation:

- *Towards scalability*

$$\tilde{\mathcal{E}}^{\text{aug}} = \tilde{\mathcal{E}} \cup \tilde{\mathcal{E}}^{\text{JSD}} \cup \tilde{\mathcal{E}}^{\text{FS}} \cup \tilde{\mathcal{E}}^{\text{SP}}$$

Sub-graph | High class hom. edges | High feat. smooth. edges | High struc. prox. edges

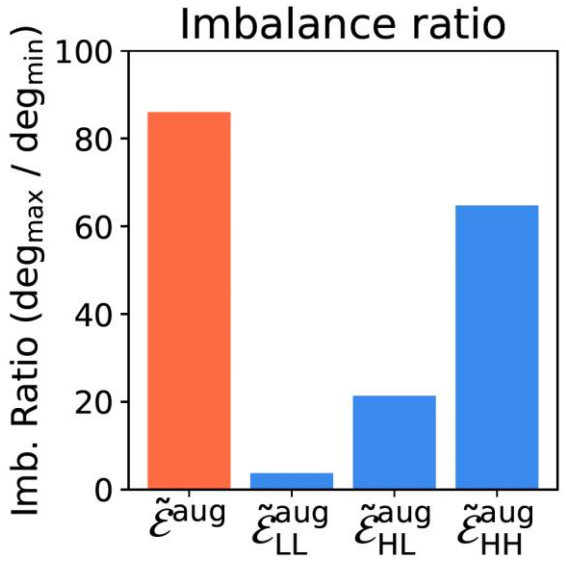
Inefficient solution

It requires computing the JSD between all node pairs in every epoch → Time consuming $O(|\tilde{\mathcal{V}}|^2)$

Efficient solution

1) We construct k-NN graphs of nodes in $\tilde{\mathcal{V}}$ based on the FS and SP, $\tilde{\mathcal{E}}_k^{\text{FS}}$ and $\tilde{\mathcal{E}}_k^{\text{SP}}$, respectively. $\tilde{\mathcal{E}}^{\text{aug}} = \tilde{\mathcal{E}} \cup \tilde{\mathcal{E}}_k^{\text{FS-JSD}} \cup \tilde{\mathcal{E}}_k^{\text{SP-JSD}}$
2) We compute the JSD values of the edges in $\tilde{\mathcal{E}}_k^{\text{FS}}$ and $\tilde{\mathcal{E}}_k^{\text{SP}}$ to obtain $\tilde{\mathcal{E}}_k^{\text{FS}-\text{JSD}}$ and $\tilde{\mathcal{E}}_k^{\text{SP}-\text{JSD}}$.
3) Complexity is alleviated $O(|\tilde{\mathcal{V}}|^2) \rightarrow O(|\tilde{\mathcal{E}}_k^*|)$, $|\tilde{\mathcal{V}}|^2 \gg |\tilde{\mathcal{E}}_k^*|$

Dealing with the Challenges of Sub-graph Extraction



Group-Training:

- *Balance the node degree distribution.*
- *Split the edge sets into multiple sub-groups.*
  - $\tilde{\mathcal{E}}^{\text{aug}}_{\text{LL}}$, $\tilde{\mathcal{E}}^{\text{aug}}_{\text{HL}}$, $\tilde{\mathcal{E}}^{\text{aug}}_{\text{HH}}$
- $\tilde{\mathcal{E}}^{\text{aug}}_{\text{LL}}$, $\tilde{\mathcal{E}}^{\text{aug}}_{\text{HL}}$, and $\tilde{\mathcal{E}}^{\text{aug}}_{\text{HH}}$ get more balanced than $\tilde{\mathcal{E}}^{\text{aug}}$

➡ Design a balanced link prediction loss

- $L_E^l = L^l_{\tilde{\mathcal{E}}^{\text{aug}}_{\text{LL}}} + L^l_{\tilde{\mathcal{E}}^{\text{aug}}_{\text{HL}}} + L^l_{\tilde{\mathcal{E}}^{\text{aug}}_{\text{HH}}}$

➡ Low-degree nodes are far more involved in computing $L^l_{\tilde{\mathcal{E}}^{\text{aug}}_{\text{LL}}}$ and $L^l_{\tilde{\mathcal{E}}^{\text{aug}}_{\text{HL}}}$ than $L^l_{\tilde{\mathcal{E}}^{\text{aug}}}$.

L_E^l is computed with more emphasis on low-degree nodes

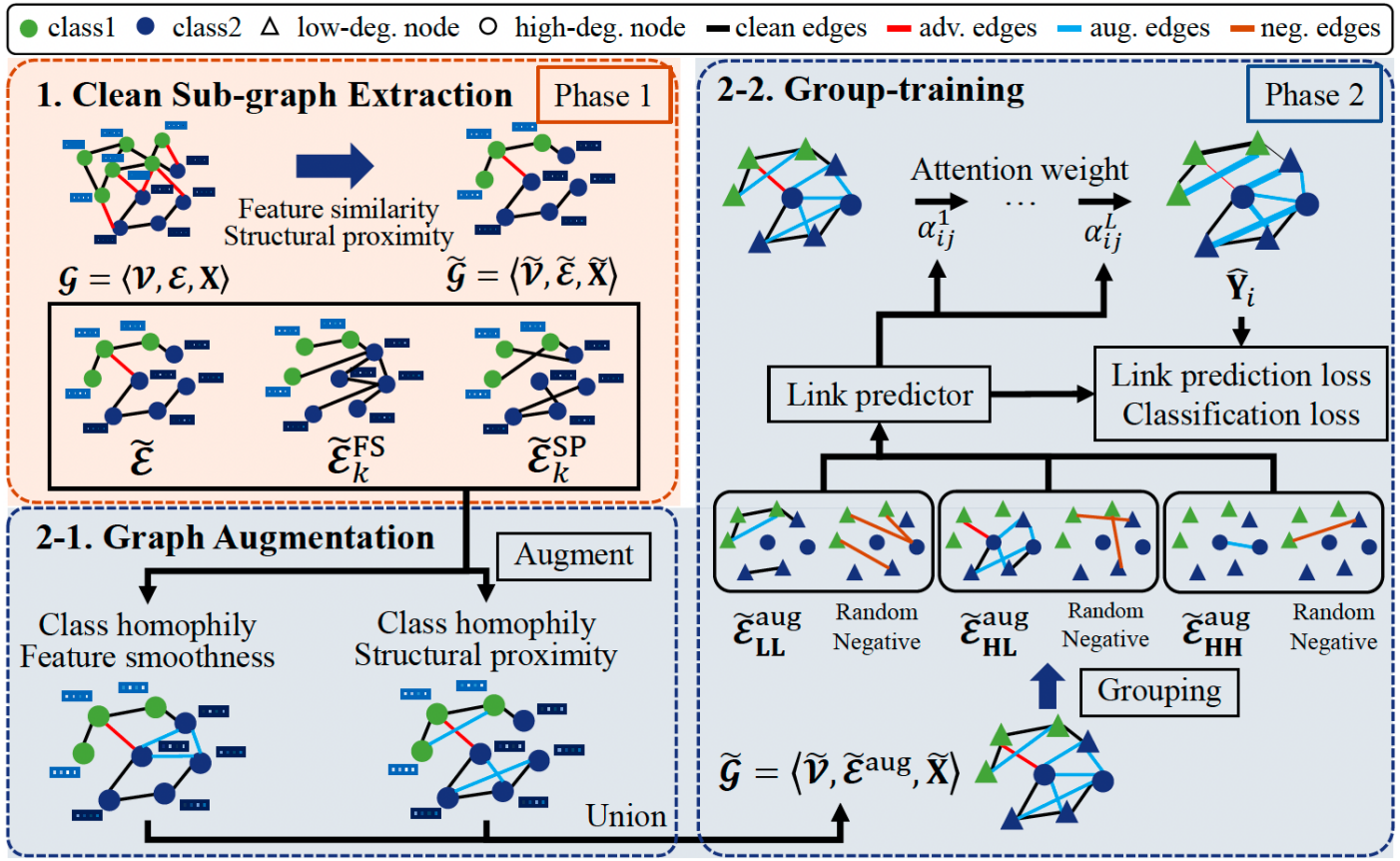
Summary of SG-GSR

Training loss

$$L_{\text{final}} = L_{\tilde{\mathcal{Y}}} + \lambda_E \sum_{l=1}^{L} L_E^l$$

Inference

- The learned link predictor assigns low weights to adv. edges



Experiments

Domain	Dataset	# Nodes	# Edges	# Features	# Classes
Citation graph	Cora	2,485	5,069	1,433	7
	Citeseer	2,110	3,668	3,703	6
	Pubmed	19,717	44,338	500	3
Blog graph	Polblogs	1,222	16,714	/	2
Co-purchase graph	Amazon	13,752	245,861	767	10
Co-review graph	Garden	7,902	19,383	300	5
	Pet	8,664	69,603	300	5

Existing benchmarks

New benchmarks we introduced

For the existing benchmarks, we artificially generate the attacked graph.

- **Structure attack:** metattack, nettack
- **Structure-feature attack:** random gaussian noise 50% in addition to the structure attack

For the new benchmarks, we **simulate real-world fraudsters' attacks** on e-commerce systems.

Experiments

Node classification performance under artificially generated adversarial attack

Table 1: Node classification performance under non-targeted attack (i.e., *metattack*) and feature attack. OOM indicates out of memory on 12GB TITAN Xp. OOT indicates out of time (24h for each run is allowed).

Dataset	Setting	SuperGAT	RGCN	ProGNN	GEN	ELASTIC	AirGNN	SLAPS	RSGNN	CoGSL	STABLE	EvenNet	SE-GSL	SG-GSR
Cora	Clean	85.4±0.3	84.0±0.1	82.9±0.3	83.9±0.8	**85.5±0.4**	83.6±0.3	75.1±0.2	85.1±0.3	80.2±0.3	85.1±0.3	**85.5±0.4**	**85.5±0.3**	**85.5±0.1**
	+ Meta 25%	62.6±1.7	53.4±0.3	70.7±0.2	67.4±1.2	67.5±0.8	63.0±0.7	75.0±0.5	81.8±0.3	63.6±0.3	79.0±0.4	75.0±0.3	70.2±0.4	**83.1±0.5**
	+ Feat attack	56.5±0.6	49.4±0.9	47.7±0.5	45.2±2.4	55.9±1.6	50.4±0.6	49.2±0.1	65.7±2.1	49.6±1.5	52.2±0.1	56.5±0.5	51.8±0.5	**67.6±1.4**
Citeseer	Clean	75.1±0.2	73.0±0.4	72.5±0.5	75.5±0.3	74.7±0.4	73.2±0.3	73.6±0.1	74.4±1.1	**76.2±0.1**	75.5±0.7	74.2±0.2	76.1±0.5	75.4±0.2
	+ Meta 25%	64.5±0.6	58.6±0.9	68.4±0.6	71.9±0.8	66.3±1.0	62.2±0.6	73.1±0.6	73.9±0.7	71.6±0.7	73.4±0.3	71.6±0.3	70.3±0.8	**75.2±0.1**
	+ Feat attack	57.1±0.9	50.3±0.6	52.6±0.2	50.4±1.1	60.8±1.3	58.1±1.1	52.3±0.4	64.0±0.3	57.4±1.5	58.4±0.4	59.7±0.4	59.0±0.9	**66.8±1**
Pubmed	Clean	84.0±0.5	86.9±0.1	OOM	86.5±0.5	**88.1±0.1**	87.0±0.1	83.4±0.3	84.8±0.4	OOM	85.5±0.2	87.5±0.2	OOT	87.6±0.2
	+ Meta 25%	74.4±1.8	82.0±0.3	OOM	80.1±0.3	85.4±0.1	84.2±0.0	83.1±0.1	84.7±0.5	OOM	81.6±0.6	87.2±0.2	OOT	**87.3±0.2**
	+ Feat attack	58.4±0.3	44.9±0.8	OOM	52.6±0.2	55.3±0.6	62.3±0.1	53.3±0.8	65.2±0.3	OOM	54.7±0.7	64.6±3.9	OOT	**65.5±0.5**
Polblogs	Clean	96.0±0.3	95.4±0.1	93.2±0.6	96.1±0.4	95.7±0.3	95.0±0.7	54.1±1.3	93.0±1.8	95.2±0.1	95.6±0.4	95.6±0.4	95.2±0.6	**96.2±0.1**
	+ Meta 25%	79.6±2.0	66.9±2.2	63.2±4.4	79.3±7.7	63.6±1.5	57.3±4.4	52.2±0.1	65.0±1.9	51.9±0.2	75.2±3.4	59.1±6.1	68.3±1.2	**87.8±0.7**
Amazon	Clean	82.5±1.1	82.2±1.3	OOM	90.2±0.2	89.6±0.1	87.6±0.8	79.6±0.8	89.6±1.2	OOM	88.8±0.4	88.8±0.5	OOT	**91.1±0.2**
	+ Meta 25%	76.0±1.6	73.2±0.7	OOM	85.6±0.9	86.7±0.2	85.6±0.4	79.0±0.3	86.9±1.6	OOM	81.7±0.3	85.4±1.5	OOT	**89.2±0.2**
	+ Feat attack	75.2±0.5	71.1±2.3	OOM	85.1±0.6	85.4±0.3	83.3±0.2	71.8±0.6	85.0±1.5	OOM	79.5±0.8	85.3±0.8	OOT	**87.2±0.4**

Table 2: Node classification performance under targeted attack (i.e., *nettack*) and feature attack.

Dataset	Setting	SuperGAT	RGCN	ProGNN	GEN	ELASTIC	AirGNN	SLAPS	RSGNN	CoGSL	STABLE	EvenNet	SE-GSL	SG-GSR
Cora	Clean	83.1±1.0	81.5±1.1	85.5±0.0	82.7±3.5	**86.4±2.1**	79.9±1.1	70.7±2.3	84.3±1.0	76.3±0.6	85.5±1.0	85.1±1.6	85.5 ±1.0	**86.4±1.1**
	+ Net 5	60.6±2.8	55.8±0.6	67.5±0.0	61.5±3.9	67.5±2.1	61.0±2.5	68.7±2.6	73.1±1.5	61.9±0.6	76.3±0.6	66.3±1.2	68.7±0.6	**77.1±1.7**
	+ Feat attack	59.4±2.5	52.6±0.6	57.8±0.0	47.0±2.0	63.1±1.8	54.2±1.0	39.0±3.0	71.9±0.6	46.6±0.6	64.7±1.5	60.6±2.0	59.0±0.5	**72.7±1.1**
Citeseer	Clean	82.5±0.0	81.0±0.0	82.5±0.0	82.5±0.0	82.5±0.0	82.5±1.3	81.5±0.8	84.1±0.0	81.5±0.8	82.5±0.0	82.5±0.0	82.5±0.0	**85.7±2.2**
	+ Net 5	54.5±4.9	50.3±3.7	71.5±0.0	77.3±1.5	79.9±0.9	70.4±2.0	81.0±1.3	78.8±2.0	79.9±0.8	82.5±0.0	79.9±2.1	82.5±0.0	**83.1±0.8**
	+ Feat attack	49.2±1.3	47.1±3.3	68.3±0.0	40.2±4.6	57.7±4.0	52.9±3.0	70.9±2.7	74.6±2.6	46.0±2.6	65.1±3.4	63.5±4.8	77.8±1.5	**83.1±2.7**
Pubmed	Clean	87.6±0.4	89.8±0.0	OOM	89.8±0.0	90.5±0.3	**90.9±0.2**	80.5±1.5	88.4±0.5	OOM	89.3±0.5	**90.9±0.5**	OOT	**90.9±1.9**
	+ Net 5	70.6±0.7	70.1±0.3	OOM	72.0±0.0	85.8±0.3	83.0±0.9	80.5±1.5	87.8±1.3	OOM	83.3±0.3	72.2±0.7	OOT	**88.0±0.3**
	+ Feat attack	70.4±1.2	61.5±1.3	OOM	61.8±0.0	**78.1±0.8**	77.1±0.9	56.6±2.0	76.5±0.7	OOM	73.1±0.7	67.9±0.9	OOT	75.8±2.0
Polblogs	Clean	97.7±0.4	97.4±0.2	97.1±0.3	97.8±0.2	97.8±0.3	97.3±0.2	54.1±1.3	96.4±0.7	96.9±0.2	97.5±0.2	**97.9±0.4**	97.0±0.4	**97.9±0.2**
	+ Net 5	95.9±0.2	93.6±0.5	96.1±0.6	94.8±1.2	96.2±0.3	90.0±0.9	51.4±3.4	93.4±0.7	89.6±0.6	96.1±0.4	94.2±1.1	95.1±0.3	**96.5±0.2**

SG-GSR consistently outperforms the baselines.

Under structure-attack,

1) SG-GSR outperforms multi-faceted methods
   - Thanks to the clean sub-graph extraction.
2) SG-GSR outperforms feature-based methods
   - Thanks to exploiting rich structural information.

Under structure-feature attack

1) SG-GSR outperforms feature-based methods
   - Leveraging the structural information alleviates the weakness of feature-based method.

Experiments

Node classification performance under E-commerce fraud

Table 3: Node classification under e-commerce fraud.

	Garden		Pet	
Methods	Clean	+ Fraud	Clean	+ Fraud
SuperGAT	86.0±0.4	81.8±0.3	87.3±0.1	80.6±0.3
RGCN	87.1±0.5	81.5±0.3	86.6±0.1	78.5±0.2
ProGNN	OOT	OOT	OOT	OOT
GEN	87.1±0.6	82.2±0.0	88.5±0.6	81.1±0.3
ELASTIC	**88.4±0.1**	82.9±0.1	88.9±0.1	81.3±0.2
AirGNN	87.1±0.2	80.9±0.4	88.5±0.1	79.3±0.3
SLAPS	79.3±0.8	74.6±0.2	81.4±0.2	75.8±0.2
RSGNN	81.8±0.6	76.3±0.0	81.6±0.4	74.2±0.0
CoGSL	OOM	OOM	OOM	OOM
STABLE	84.3±0.3	81.0±0.3	87.9±0.2	80.8±0.2
EvenNet	86.3±0.2	81.3±0.4	88.5±0.2	81.0±0.1
SE-GSL	82.0±0.4	77.3±0.6	87.9±0.4	77.5±0.7
SG-GSR	88.3±0.1	**83.3±0.2**	**89.4±0.1**	**81.9±0.1**

- Product-product graph from Amazon product review data.
  - Node feature: product review texts
  - Graph structure: co-review relationships by the same user
  - Node label: product categories
- Fraudsters randomly write fake product reviews on the numerous products.
  - Numerous malicious co-review edges are added to the graph structure.
  - Noisy random review are injected into the node features.
- **SG-GSR also works well under attacks that are plausible in the real-world systems.**

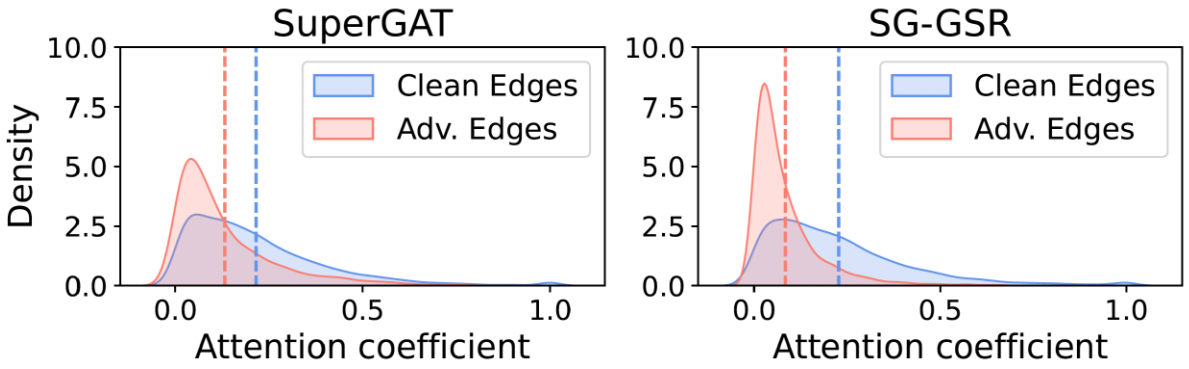
Contribution

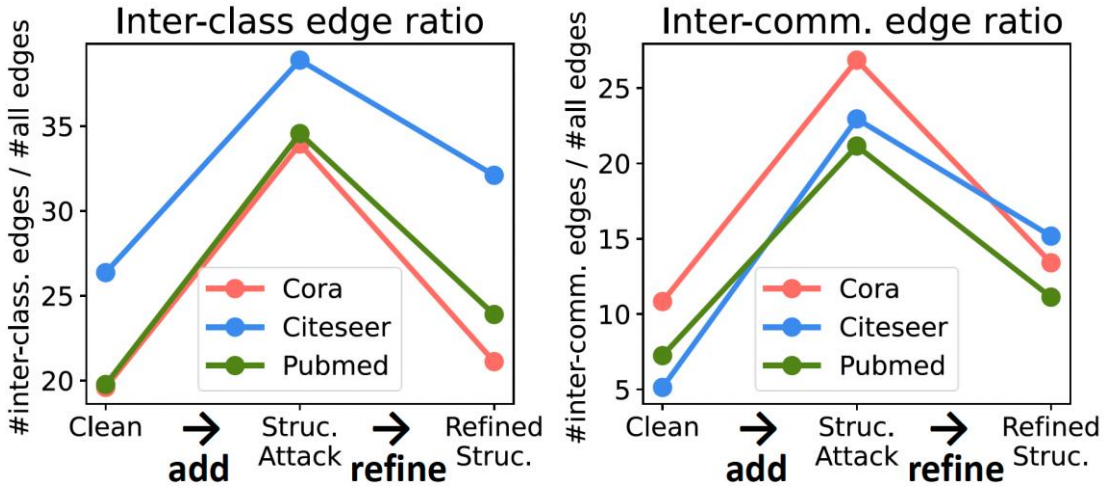
This is the first work proposing new graph datasets that **closely imitate a real-world e-commerce system containing malicious fraudsters.**

We expect these datasets to foster practical research in adversarial attacks on GNNs

Experiments

Analysis of Refined Graph





Attention weight analysis compared to SuperGAT

- Large attention weights for the clean edges
- small attention weights for the adversarial edges

Community structure reconstruction

- Successfully remove the inter-class/community edges by the proposed GSR module.

Experiments

Main Ablation Study

Component			Cora			Citeseer		
SE	GA	GT	Clean	+ Meta 25%	+ Feat. Attack	Clean	+ Meta 25%	+ Feat. Attack
✗	✗	✗	84.3±0.5	62.6±1.7	56.5±0.6	74.2±0.2	64.5±0.6	57.1±0.9
✓	✗	✗	84.4±0.5	80.6±0.7	58.6±1.1	74.6±0.2	74.4±0.4	60.9±0.5
✓	*Rand*	✗	83.6±0.4	78.0±1.4	49.9±1.2	74.9±0.6	73.6±0.4	48.7±1.7
✓	C	✗	84.6±0.3	81.0±0.3	60.1±0.8	74.6±0.2	74.3±0.6	59.0±0.6
✓	C, F	✗	84.9±0.3	81.7±0.1	64.1±0.9	74.5±0.1	74.6±0.6	62.2±0.6
✓	C, F, S	✗	84.6±0.1	82.1±0.3	64.3±0.7	74.8±0.2	74.7±0.4	62.6±0.6
✓	C, F, S	✓	**85.5±0.1**	**83.4±0.5**	**67.6±1.4**	**75.4±0.2**	**75.2±0.1**	**66.8±1.0**

Dataset	Attack	Node	SG-GSR w/o *GT*	SG-GSR	Diff.(%)
Cora	Clean	high-degree	85.6±0.5	**86.7±0.2**	1.1
		low-degree	83.8±0.1	**84.5±0.6**	0.7
	+ Meta 25%	high-degree	84.4±0.8	**86.7±0.8**	2.3
		low-degree	78.2±0.8	**80.7±0.1**	2.5
	+ Feat. Attack	high-degree	73.7±1.3	**76.0±1.5**	2.3
		low-degree	56.9±1.1	**61.3±1.4**	4.4
Citeseer	Clean	high-degree	76.4±0.2	**77.7±0.4**	1.3
		low-degree	73.0±0.4	**73.2±0.4**	0.2
	+ Meta 25%	high-degree	77.9±0.7	**78.6±0.2**	0.7
		low-degree	70.8±0.3	**72.1±0.4**	1.3
	+ Feat. Attack	high-degree	71.9±0.6	**73.9±1.1**	2.0
		low-degree	54.0±0.2	**60.3±1.0**	6.3

- Ablation studies indicate that
  1) SE (sub-graph extraction) is considerably helpful for defending adversarial attacks
  2) GA (graph augmentation) supplements the loss of structural information, thereby achieving the robustness
  3) GT (group training) improves the generalization ability to low-degree nodes
     - ➢ resulting in the overall performance improvement.

Conclusion

- We have discovered that existing GSR methods are limited by narrow assumptions.
- We propose SG-GSR, which refines the attacked graph structure in a self-guided manner.
- We propose GA and GT in order to address the two technical challenges.
- We verify the effectiveness of SG-GSR through extensive experiments
- We newly introduce graph datasets that mimics fraudsters' attacks on e-commerce systems.

THE WEB CONFERENCE ACM

KAIST

Thank you for listening!

Please refer to our paper

“Self-Guided Robust Graph Structure Refinement"

Contact: yeonjun.in@kaist.ac.kr