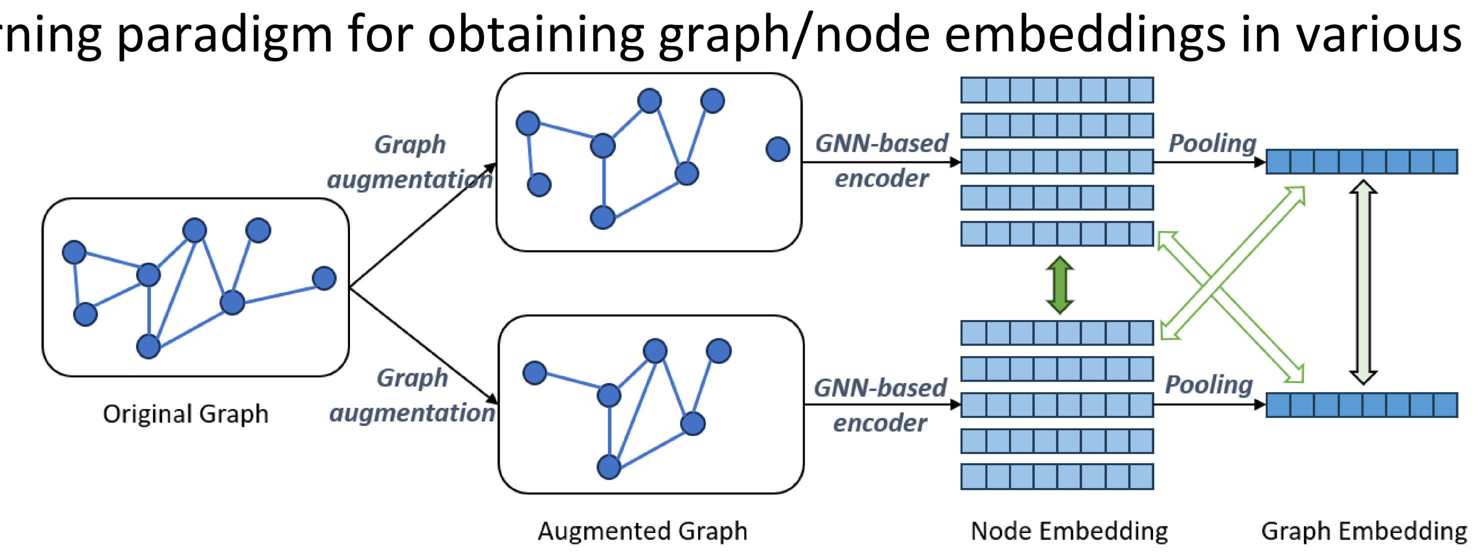
Graph Contrastive Learning with Cohesive Subgraph Awareness

Yucheng Wu[1], Leye Wang[1*], Xiao Han[2*], Han-Jia Ye[3]

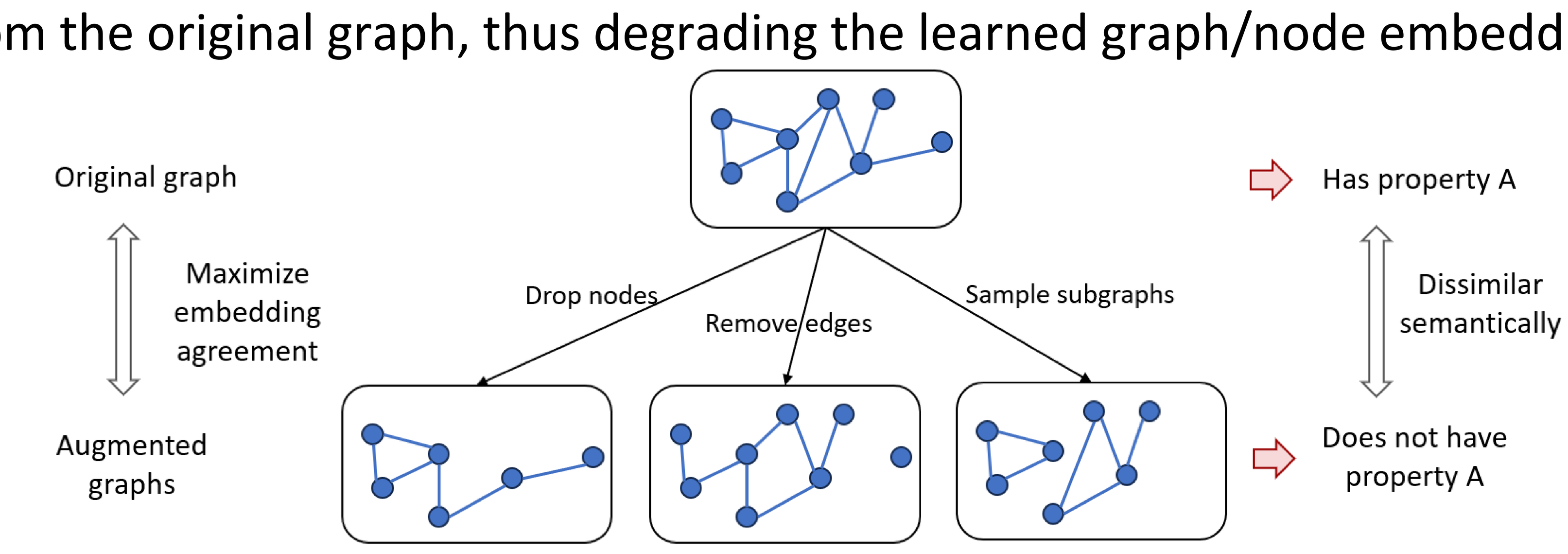
[1]Peking University, [2]Shanghai University of Finance and Economics, [3]Nanjing University, *Corresponding authors

Introduction

1. Graph Contrastive Learning (GCL) has emerged as a promising self-supervised learning paradigm for obtaining graph/node embeddings in various applications.

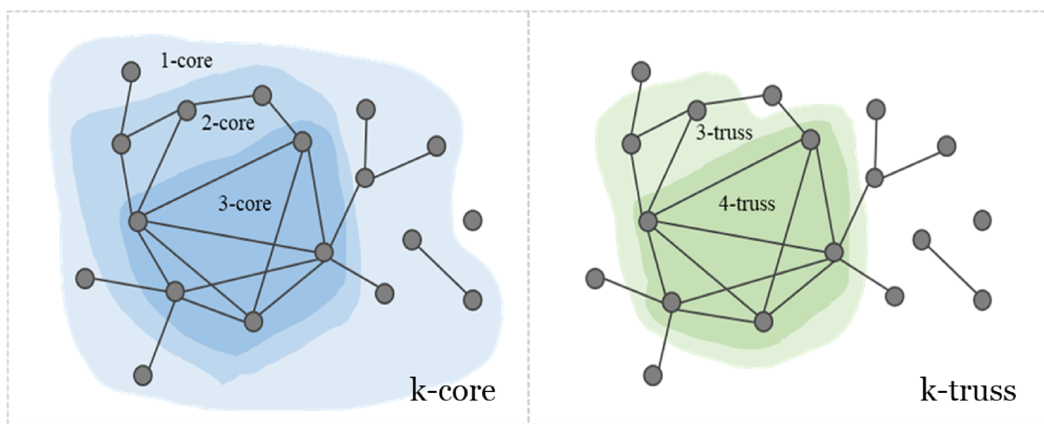


2. Shortcomings of existing augmentation strategies: randomly deleting important edges/nodes may cause the augmented views to vary far away from the original graph, thus degrading the learned graph/node embedding.



3. Basic idea: introduce cohesive subgraphs to guide topology augmentations

- *Cohesive subgraph* is a widely prevalent and significant substructure with crucial applications in various fields.

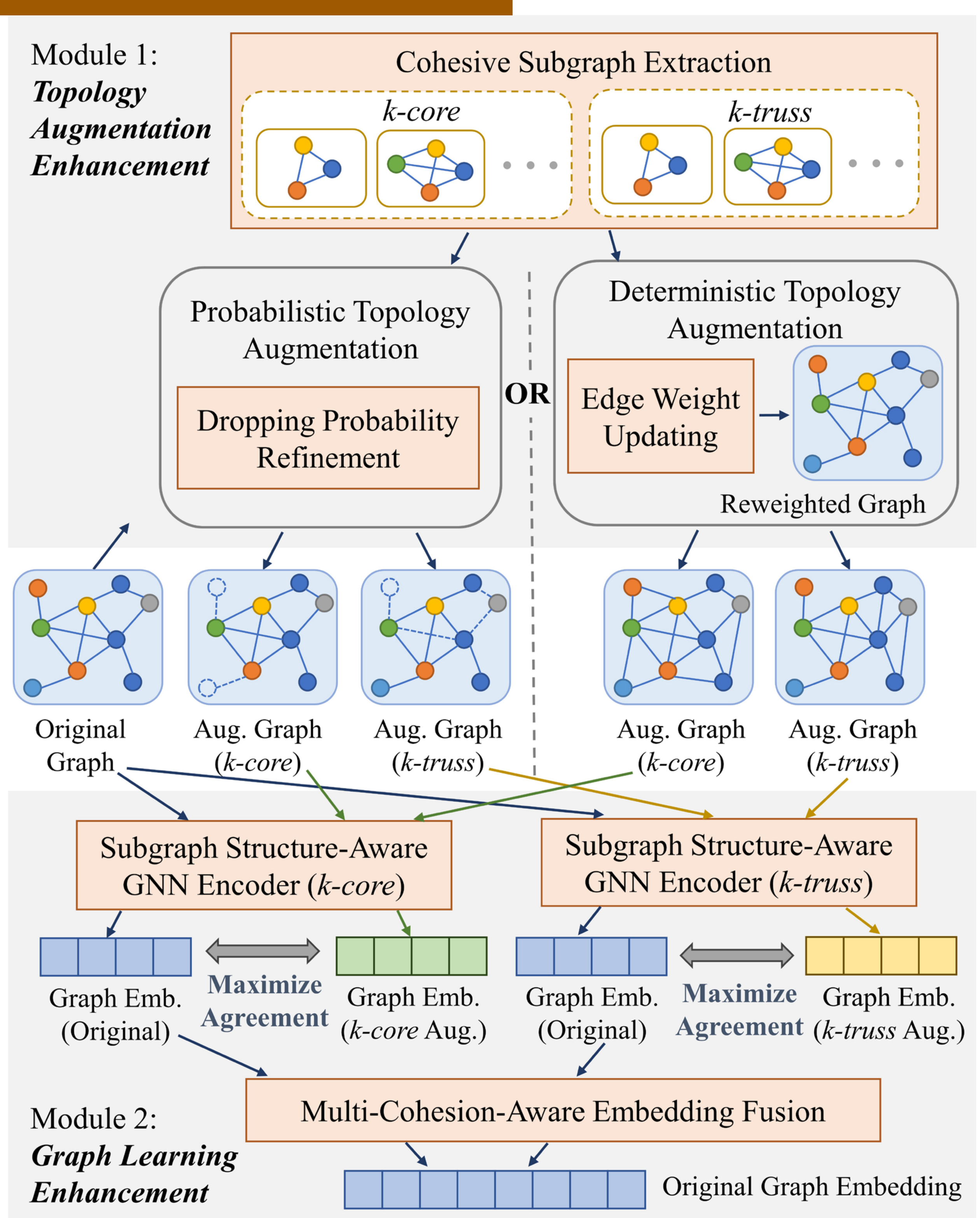


- **k-core:** every node has at least k links to the other nodes.
- **k-truss**: every edge is in at least (k − 2) triangles of the subgraph.

4. Research questions

- **Property Enrichment.** Can we enrich the topology augmentation with more essential graph properties to improve GCL?
- **Unified framework.** Can we design a unified framework that incorporates graph properties into various GCL methods?
- **Expressive Networks.** Most existing GCL methods use GNNs as encoders, but GNNs encounter difficulties in capturing subgraph properties. Can we design a more expressive graph encoder that can capture subgraph information effectively?

CTAug Framework



Module 1: Topology Augmentation Enhancement

- **Probabilistic Topology Augmentation**
  - reduce the probability of node/edge dropping operations on cohesive subgraphs
- **Deterministic Topology Augmentation**
  - assign larger weights to the graph edges in cohesive subgraphs so that the graph diffusion process would favor the large-weighted edges

Module 2: Graph Learning Enhancement

- **Subgraph-aware GNN encoder**
  - MPNNs have been proven to be limited in capturing subgraph properties, e.g., counting substructures
  - GSN: $AGG\left((h_v, h_u, s_v, s_u)_{u\in\mathcal{N}(v)}\right)$
  - To improve efficiency and tracking of original graph, we propose O-GSN: $AGG\left((h_v, h_u, s_v^o, s_u^o)_{u\in\mathcal{N}(v)}\right)$
- **Multi-Cohesion Embedding Fusion**
  - concatenate embeddings: $z_i = ||_{c\in\mathbb{C}} z_i^c$

Experiments

Table 2: Accuracy (%) on graph classification (OOM: out-of-memory).

Method	Social Graphs (High Degree)				Social Graphs (Low Degree)			Biomedical Graphs		
	IMDB-B	IMDB-M	COLLAB	AVG.	RDT-B	RDT-T	AVG.	ENZYMES	PROTEINS	AVG.
InfoGraph	71.34±0.24	47.93±0.71	69.12±0.15	62.80	89.39±1.81	76.23±0.00	82.81	26.73±3.75	74.09±0.48	50.41
AD-GCL	71.28±1.10	47.59±0.62	71.22±0.89	63.36	88.84±0.90	76.51±0.00	82.68	27.33±2.28	73.39±0.85	50.36
AutoGCL	71.14±0.71	48.61±0.55	67.27±2.64	62.34	89.31±1.48	77.13±0.00	83.22	29.83±2.24	73.33±0.27	51.58
RGCL	71.14±0.64	48.28±0.60	73.48±0.93	64.30	91.38±0.40	OOM	/	33.33±1.61	73.37±0.35	53.35
SimGRACE	71.44±0.28	48.81±0.92	69.07±0.24	63.11	86.65±1.12	76.64±0.01	81.65	31.37±1.59	73.42±0.37	52.40
GCL-SPAN	70.84±0.37	47.95±0.47	74.33±0.40	64.37	OOM	OOM	/	27.63±1.13	72.06±0.25	49.85
GraphCL	71.48±0.44	48.11±0.60	72.36±1.76	63.98	91.69±0.70	77.44±0.03	84.57	32.83±2.05	74.32±0.76	53.58
CTAug-GraphCL	76.60±1.02	51.12±0.57	81.72±0.26	69.81	**92.28±0.33**	**77.48±0.01**	**84.88**	39.17±1.00	74.10±0.33	56.64
JOAO	71.40±0.38	48.68±0.36	73.40±0.46	64.49	91.66±0.59	77.24±0.00	84.45	34.60±1.06	74.32±0.46	54.46
CTAug-JOAO	**76.80±0.71**	**51.19±0.88**	**81.90±0.53**	**69.96**	92.19±0.24	77.35±0.02	84.77	**39.92±1.36**	74.46±0.13	**57.19**
MVGRL	71.88±0.73	50.19±0.40	80.48±0.29	67.52	OOM	OOM	/	34.20±0.67	74.33±0.62	54.27
CTAug-MVGRL	73.04±0.65	50.79±0.54	81.09±0.37	68.31	OOM	OOM	/	35.46±1.20	**75.00±0.38**	55.23

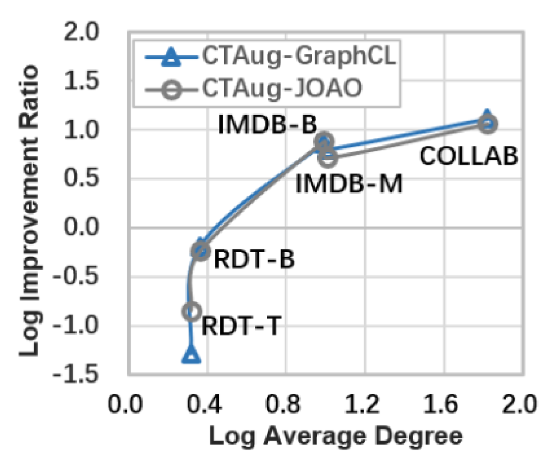


Figure 2: *CTAug*'s improvement on datasets with varying average degrees.

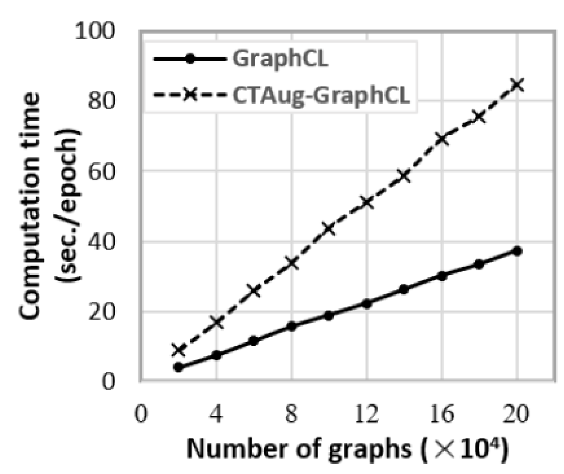


Figure 3: Scalability test on *RDT-T*.

Table 3: Ablation study of *CTAug-GraphCL*.

Method	IMDB-B	IMDB-M	COLLAB	AVG.
CTAug-GraphCL	**76.60±1.02**	51.12±0.57	**81.72±0.26**	**69.81**
Module Ablation				
Only Module 1	71.54±0.27	49.11±0.48	72.64±0.63	64.43
Only Module 2	73.80±1.21	50.27±0.81	80.03±0.42	68.03
Cohesion Property Ablation				
Only k-core	75.92±0.67	**51.39±0.14**	81.36±0.16	69.56
Only k-truss	76.12±1.20	50.99±0.57	80.71±0.30	69.27

Theoretical Analysis

Theorem 4.3. Suppose f is a minimal sufficient encoder. If $I(\mathcal{G}';\mathcal{G};y)$ increases, $I(f(\mathcal{G});y)$ will also increase.

- Cohesive properties are closely tied to graph label y
- Preserve more cohesive properties of the original graph $\mathcal{G}$ during graph augmentation → retain more information related to y for embedding → increase downstream task performance

Theorem 4.4. Let f_1 represent our proposed O-GSN encoder with k-core ($k \geq 2$) or k-truss ($k \geq 3$) subgraphs considered in subgraph structures $\mathcal{H}$, and let f_2 denote GIN (the default encoder). After sufficient training of f_1 and f_2, $I(f_1(\mathcal{G});y) > I(f_2(\mathcal{G});y)$.

- Substitute the default GIN encoder with O-GSN encoder → empower the encoder to preserve more information associated with y → boost the performance of downstream tasks

Conclusion

- We propose CTAug, to ***incorporate cohesion properties into the topology augmentation and graph learning processes of GCL***, which can be applied to various existing GCL mechanisms.
- Our framework provides ***a general approach for generating augmented graphs guided by prior knowledge of substructures*** applicable to any domain.

Paper | Code