Graph Contrastive Learning

Yucheng Wu

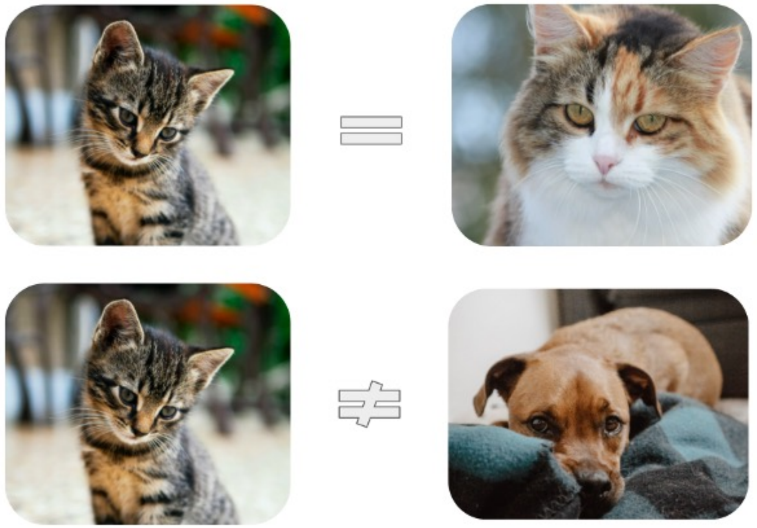
2024/1/26

Contents

Introduction

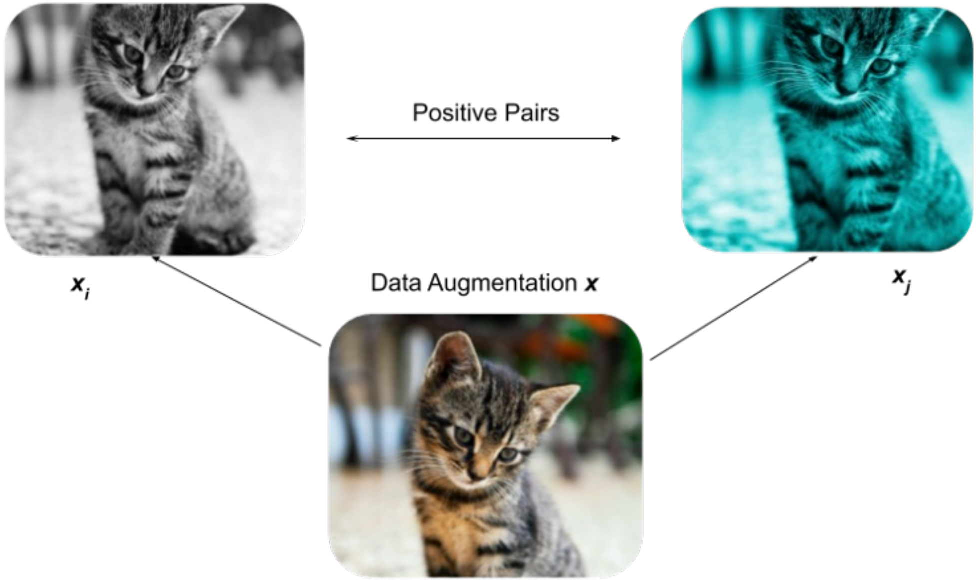
What is contrastive learning?

- contrastive learning
  (origins in computer vision field)
    - a self-supervised learning method to learn general features of datasets without labels
    - based on whether the samples are similar or not



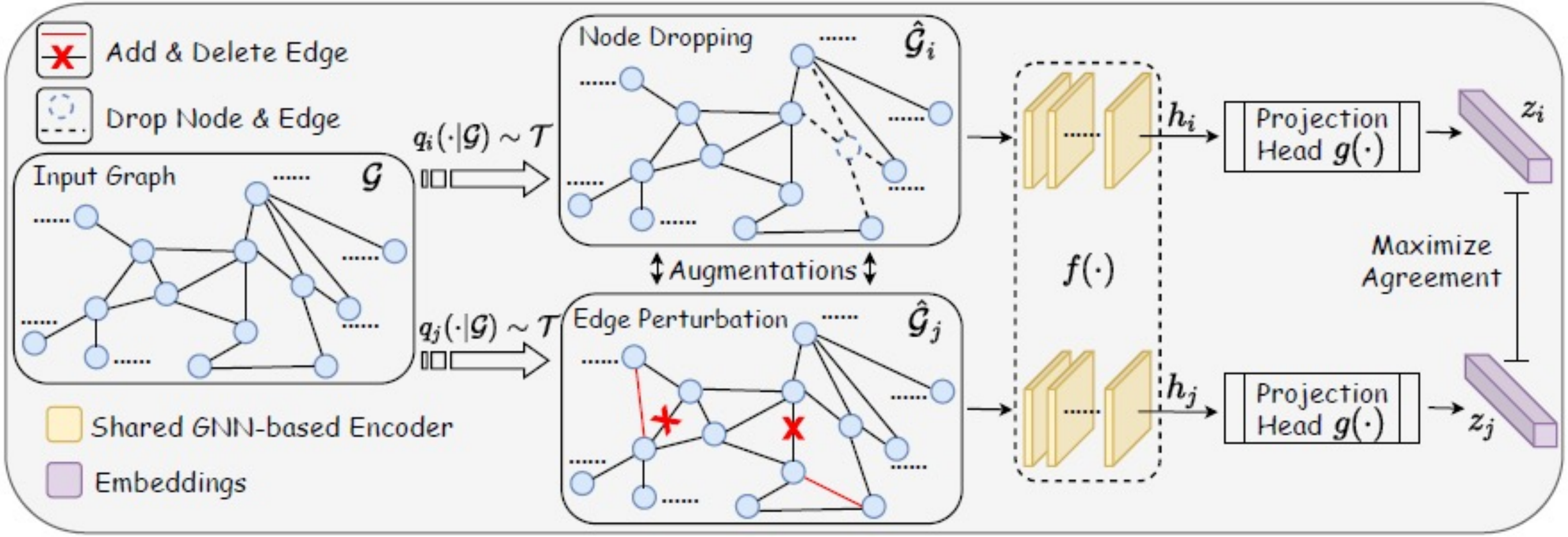
How to get similar samples?

- By data augmentation
    - Crop
    - Resize
    - Recolor
    - …



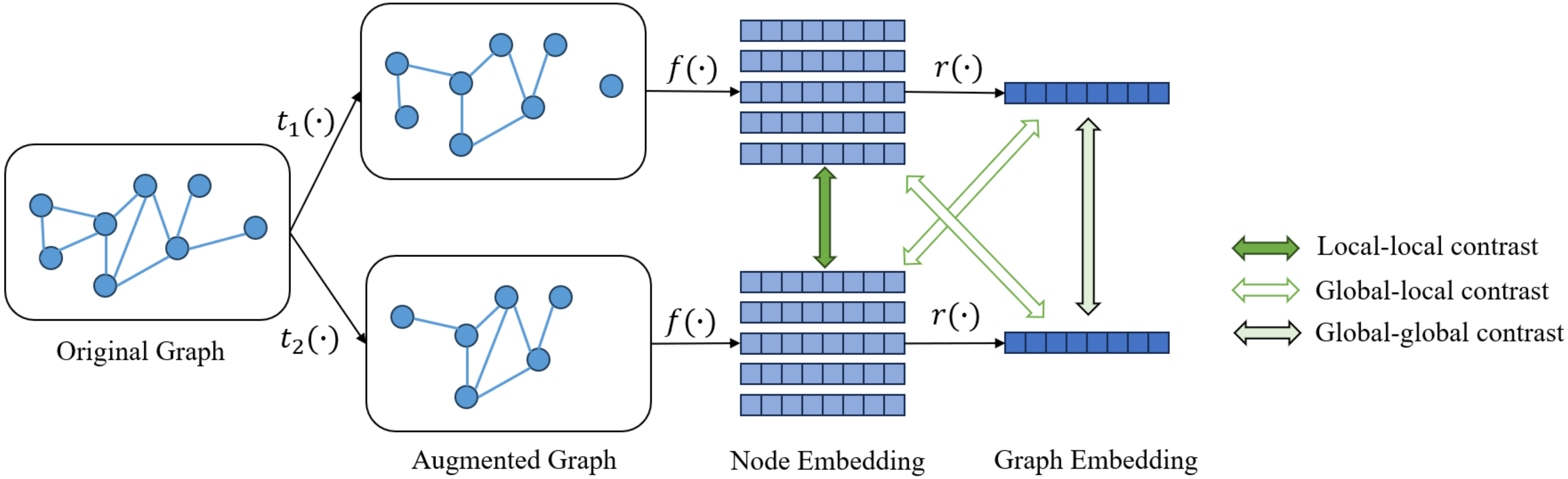
Contrastive Learning → Graph Contrastive Learning

- An example: GraphCL (Graph contrastive learning with augmentations, NIPS 2020)



Data augmentation	Type	Underlying Prior
Node dropping	Nodes, edges	Vertex missing does not alter semantics.
Edge perturbation	Edges	Semantic robustness against connectivity variations.
Attribute masking	Nodes	Semantic robustness against losing partial attributes per node.
Subgraph	Nodes, edges	Local structure can hint the full semantics.

A General Paradigm of GCL

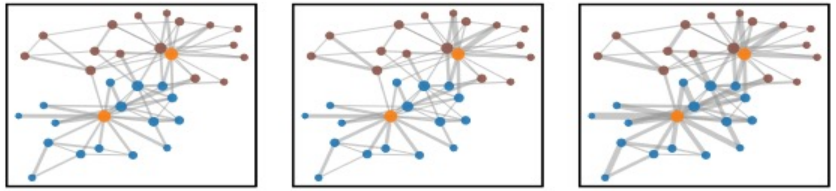


- At first, two augmented graphs are generated via **graph augmentation** functions.
- Then, the two graphs are fed into a shared GNN to learn node embedding, which is then optimized with a contrastive objective that pulls together congruent embedding pairs of the two augmented graphs while pushing others away.

$$\mathcal{L} = -\log \frac{\exp(sim(z_n^1, z_n^2)/\tau)}{\sum_{n'=1, n' \neq n}^{N} \exp(sim(z_n^1, z_{n'}^2)/\tau)}$$

Graph Augmentation Methods of GCL

- Adaptive Augmentation
  - Knowledge-based
    - Spatial



(a) Degree (b) Eigenvector (c) PageRank

$$p_{ij} = \min\left(\frac{s_{\max} - s_{ij}}{s_{\max} - \mu_s} \cdot p_e, p_\tau\right)$$

    - Spectral

$$|\phi_1(\lambda_m) - \phi_2(\lambda_m)| > |\phi_1(\lambda_n) - \phi_2(\lambda_n)|$$

low-frequency parts in a graph are more important than high-frequency parts

  - Learning-based
    - Gradient: adapts the augmentation process based on edge gradient information
    - Adversarial: $\min_{T \in \mathcal{T}} \max_{f} I(f(G); f(t(G))), \text{ where } t(G) \sim T(G)$
    - Rationale: employs a rationale generator to unveil salient features that discriminate between graph instances

Adversarial Graph Augmentation to Improve Graph Contrastive Learning (NIPS 2021)

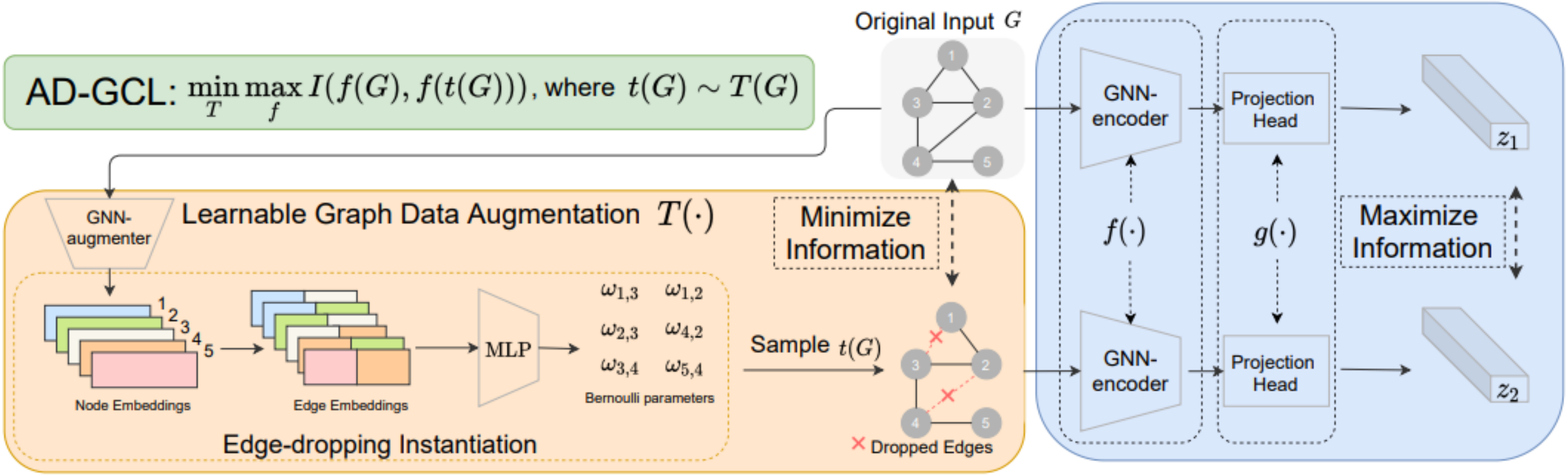
AD-GCL – adversarial GCL with a learnable augmenter

- Using adversarial training to **remove the redundant information**, according to Information Bottleneck
- Formulate as a **min-max** optimization problem:

$$\text{AD-GCL:} \quad \min_{T \in \mathcal{T}} \max_{f} I(f(G); f(t(G))), \quad \text{where } G \sim \mathbb{P}_{\mathcal{G}}, t(G) \sim T(G),$$

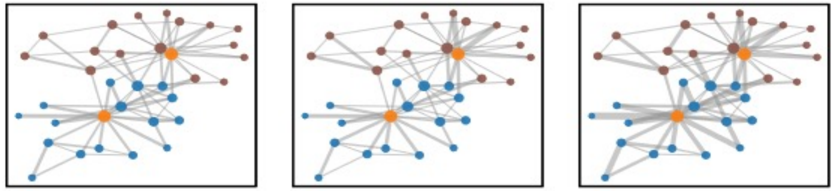
- Instantiate with a GNN-augmenter, learning how to drop edges from the initial graph:

$$\min_{\Phi} \max_{\Theta} I(f_\Theta(G); f_\Theta(t(G))) + \lambda_{\text{reg}} \mathbb{E}_G \Big[\sum_{e \in E} \omega_e / |E| \Big], \text{ where } G \sim \mathbb{P}_{\mathcal{G}}, t(G) \sim T_\Phi(G).$$



Graph Augmentation Methods of GCL

- Adaptive Augmentation
  - Knowledge-based
    - Spatial
      - introduce the **cohesion property** into topology augmentation



(a) Degree (b) Eigenvector (c) PageRank

$$p_{ij} = \min\left(\frac{s_{\max} - s_{ij}}{s_{\max} - \mu_s} \cdot p_e, p_\tau\right)$$

    - Spectral

$$|\phi_1(\lambda_m) - \phi_2(\lambda_m)| > |\phi_1(\lambda_n) - \phi_2(\lambda_n)|$$

      - low-frequency parts in a graph are more important than high-frequency parts
  - Learning-based
    - Gradient
      - adapts the augmentation process based on edge gradient information
    - Adversarial

$$\min_{T \in \mathcal{T}} \max_{f} I(f(G); f(t(G))), \text{ where } t(G) \sim T(G)$$

    - Rationale
      - employs a rationale generator to unveil salient features that discriminate between graph instances

Graph Contrastive Learning with Cohesive Subgraph Awareness

Yucheng Wu
Key Lab of High Confidence Software Technologies (Peking University), Ministry of Education; School of Computer Science, Peking University
Beijing, China
wuyucheng@stu.pku.edu.cn

Leye Wang
Key Lab of High Confidence Software Technologies (Peking University), Ministry of Education; School of Computer Science, Peking University
Beijing, China
leyewang@pku.edu.cn

Xiao Han
School of Information Management and Engineering, Shanghai University of Finance and Economics
Shanghai, China
xiaohan@mail.shufe.edu.cn

Han-Jia Ye
National Key Laboratory for Novel Software Technology, Nanjing University; School of Artificial Intelligence, Nanjing University
Nanjing, China
yehj@lamda.nju.edu.cn

ABSTRACT

Graph contrastive learning (GCL) has emerged as a state-of-the-art strategy for learning representations of diverse graphs including social and biomedical networks. GCL widely uses stochastic graph topology augmentation, such as uniform node removal, to generate augmented graphs. However, such stochastic augmentations may severely damage the intrinsic properties of a graph and deteriorate the following representation learning process. Specifically, cohesive topological properties (e.g., k-core and k-truss) indicate strong and critical connections among multiple nodes; randomly removing nodes from a cohesive subgraph may remarkably alter the graph properties. In contrast, we argue that incorporating an awareness of cohesive subgraphs during the graph augmentation

CCS CONCEPTS

• **Computing methodologies** → **Unsupervised learning**; • **Information systems** → *Social networks.*

KEYWORDS

Graph contrastive learning, self-supervised learning, social networks, cohesive subgraph

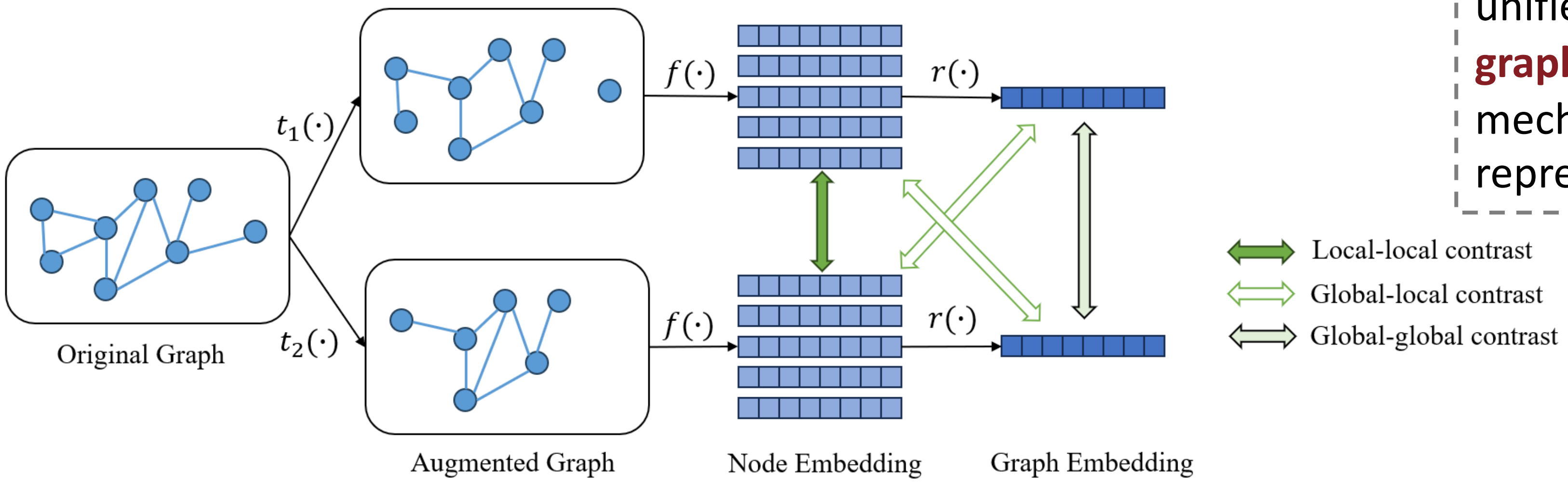
ACM Reference Format:
Yucheng Wu, Leye Wang, Xiao Han, and Han-Jia Ye. 2024. Graph Contrastive Learning with Cohesive Subgraph Awareness. In *Proceedings of the ACM Web Conference 2024 (WWW '24), May 13–17, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/XXXXXX.XXXXXX

Cohesive Topology Augmentation for Social Graph Contrastive Learning

Introduction

- Common topology augmentation strategies include node dropping, edge removal, subgraph sampling, etc. Existing methods mainly follow a **stochastic manner** to conduct topology augmentation.
- Some methods adopt total randomized augmentation operations, like removing nodes or edges with an **equivalent probability**.
- nodes and edges usually hold diverse levels of importance in a graph → randomly deleting important edges/nodes may cause the augmented views to vary far away from the original graph → degrading the learned graph/node embedding.

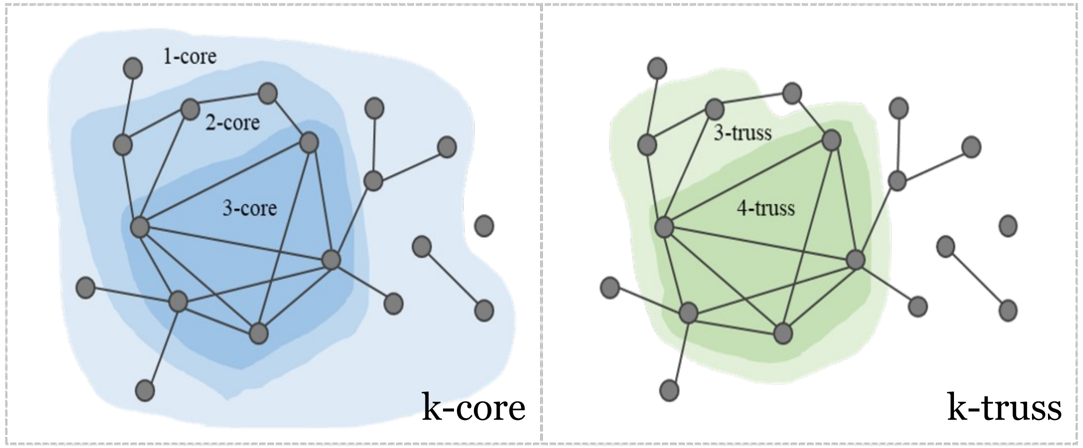
Question: Can we develop a unified framework to incorporate **graph properties** into GCL mechanisms and benefit graph representation learning?



Cohesive Topology Augmentation for Social Graph Contrastive Learning

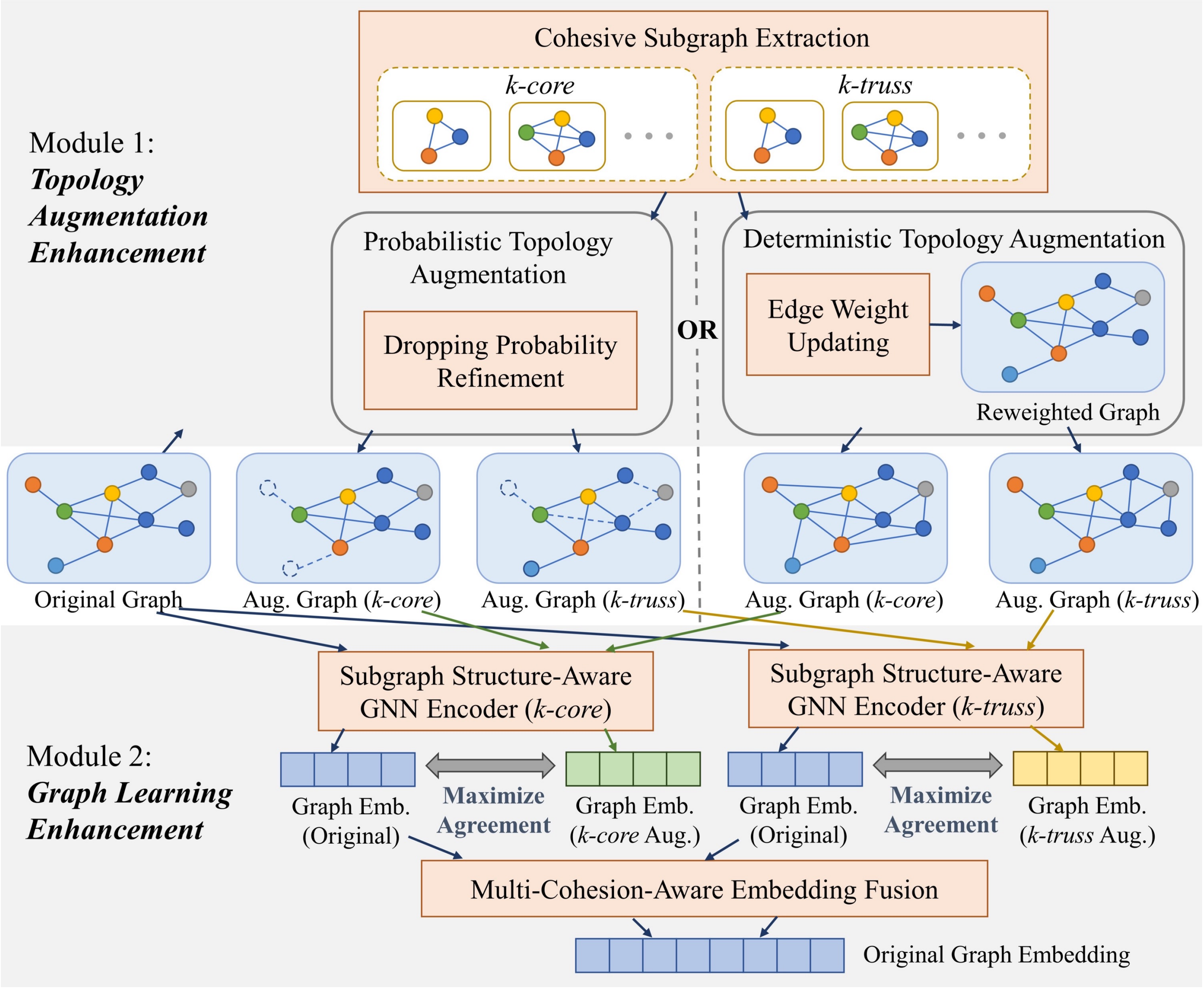
Cohesive Subgraph

- Cohesive subgraph is a widely prevalent and significant substructure with crucial applications in various fields such as Network Modeling and Analysis, Anomaly Detection, Text Analytics, Biology, Ecology, etc.
- For instance, in the field of Biology, some studies detects densely connected regions in large protein-protein interaction networks that may represent molecular complexes.
- **k-core** is a maximal subgraph in which every node has at least k links to the other nodes.
  - provide rich information for various applications, such as *user influence* and *community detection*
  - plays an important role in analyzing coauthor social networks
- **k-truss** is the largest subgraph in which every edge is in at least (k − 2) triangles of the subgraph.
  - triangle can indicate the stability of the social network topology
  - reveal the transitivity in the link formation of networks



CTAug Framework

- **Topology Augmentation Enhancement:** enhances the probabilistic and deterministic augmentation process separately with the consideration of the cohesive subgraphs;
- **Graph Learning Enhancement:** boosts GNN encoder to better capture the original graph's cohesion properties.



- **Dropping Probability Refinement**
  - reduce the probability of node/edge dropping operations on the cohesive subgraphs
  - $p'_{dr} = (1-\epsilon) \cdot p_{dr}, \ \epsilon \in (0,1]$
  - more importance in cohesive subgraph → higher probability to be saved
- **Subgraph-aware GNN encoder**
  - MPNNs have been proven to be limited in capturing subgraph properties, e.g., counting substructures
  - GSN: $AGG\left((h_v, h_u, s_v, s_u)_{u \in \mathcal{N}(v)}\right)$, s_v counts how many times node v appears in a set of subgraph structures
  - directly applying GSN into CTAug face will two issues: *low efficiency*; *losing track of the original graph*
  - we propose O-GSN: $AGG\left((h_v, h_u, s_v^o, s_u^o)_{u \in \mathcal{N}(v)}\right)$

Cohesive Topology Augmentation for Social Graph Contrastive Learning

Experiments

Table 1: Dataset statistics for graph classification.

Category	Dataset	#Graph	#Class	Avg. #Nodes	Avg. #Edges	Avg. Degree	Avg. $k_{\max}$ (k-core)	Avg. $k_{\max}$ (k-truss)
Social Graph	IMDB-B	1,000	2	19.77	96.53	9.76 (high)	9.16	10.16
	IMDB-M	1,500	3	13.00	65.94	10.14 (high)	8.15	9.15
	COLLAB	5,000	3	74.49	2457.78	65.97 (high)	40.53	41.52
	RDT-B	2,000	2	429.63	497.75	2.32 (low)	2.33	3.09
	RDT-T	203,088	2	23.93	24.99	2.08 (low)	1.58	2.46
Biomedical Graph	ENZYMES	600	6	32.63	62.14	3.81 (low)	2.98	3.80
	PROTEINS	1,113	2	39.06	72.82	3.73 (low)	3.00	3.83

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
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

Cohesive Topology Augmentation for Social Graph Contrastive Learning

Experiments

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

Table 5: Results on node classification. The baseline results (except *GRACE* and *GCA*) are copied from [60] because we follow the same experimental setup. Meanwhile, we run *GRACE* and *GCA* by ourselves as we need to ensure that the exactly same configurations (neural network hidden units, training algorithm parameters, etc.) are used for *GRACE*/*GCA* and our enhanced *CTAug-GRACE*/*CTAug-GCA* for a fair comparison (OOM: out-of-memory).

Method	Coauthor CS	Coauthor Physics	Amazon Computers	AVG.
DeepWalk+features	87.70±0.04	94.90±0.09	86.28±0.07	89.63
GAE	90.01±0.71	94.92±0.07	85.27±0.19	90.07
VGAE	92.11±0.09	94.52±0.00	86.37±0.21	91.00
DGI	92.15±0.63	94.51±0.52	83.95±0.47	90.20
GMI	OOM	OOM	82.21±0.31	/
MVGRL	92.11±0.12	95.33±0.03	87.52±0.11	91.65
GRACE	92.83±0.10	95.56±0.05	86.96±0.14	91.78
GCA	92.89±0.02	95.55±0.03	87.48±0.11	91.97
CTAug-GRACE	92.96±0.05	**95.68±0.01**	87.59±0.12	92.08
CTAug-GCA	**92.98±0.04**	95.61±0.01	**88.30±0.13**	**92.30**

Cohesive Topology Augmentation for Social Graph Contrastive Learning

Conclusion

- To introduce the awareness of cohesion properties (e.g., *k*-core and *k*-truss) into GCL, this work proposes a unified framework, called *CTAug*, that can be integrated with various existing GCL mechanisms.
- Two modules, including *topology augmentation enhancement* and *graph learning enhancement*, are designed to incorporate cohesion properties into the topology augmentation and graph learning processes of GCL, respectively.
- Extensive experiments have verified the effectiveness and flexibility of the *CTAug* framework.

Significance

- Our method provides a general approach for generating augmented graphs guided by prior knowledge of substructures applicable to any domain.

Thanks