Learning Representations for Information-rich Graphs

PhD Defense

Zexi Huang

Committee: Ambuj Singh (Chair), Yu-Xiang Wang, Xifeng Yan

Department of Computer Science, University of California, Santa Barbara

April 5, 2023

Representation learning

"Extracting key information from raw data to enable effective data science applications via deep neural networks."

Representation learning

"Extracting key information from raw data to enable effective data science applications via deep neural networks."

- For images: Convolutional Neural Networks (CNNs)
- For text: Transformers (e.g., ChatGPT)

Representation learning

"Extracting key information from raw data to enable effective data science applications via deep neural networks."

- For images: Convolutional Neural Networks (CNNs)
- For text: Transformers (e.g., ChatGPT)
- **For graphs**

Representation learning

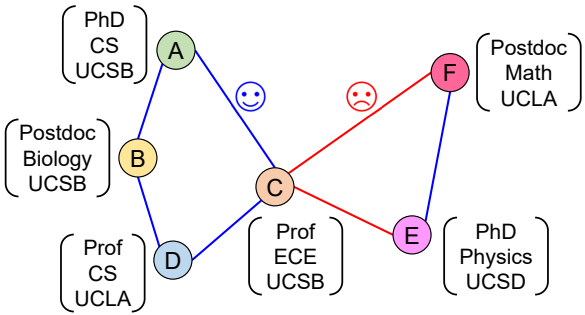
"Extracting key information from raw data to enable effective data science applications via deep neural networks."

- For images: Convolutional Neural Networks (CNNs)
- For text: Transformers (e.g., ChatGPT)
- **For graphs**

Graph applications

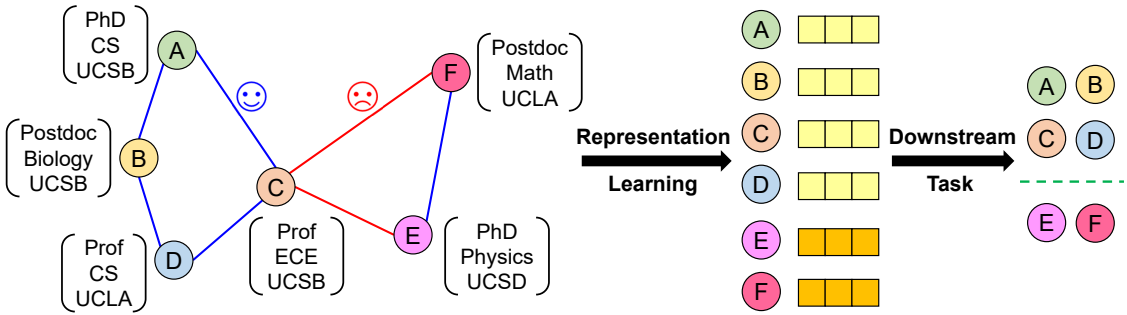
- Social network analysis: community detection
- Product and video recommendation: link prediction
- Financial fraud detection: anomaly detection
- Novel drug discovery: graph classification

Representation learning for information-rich graphs



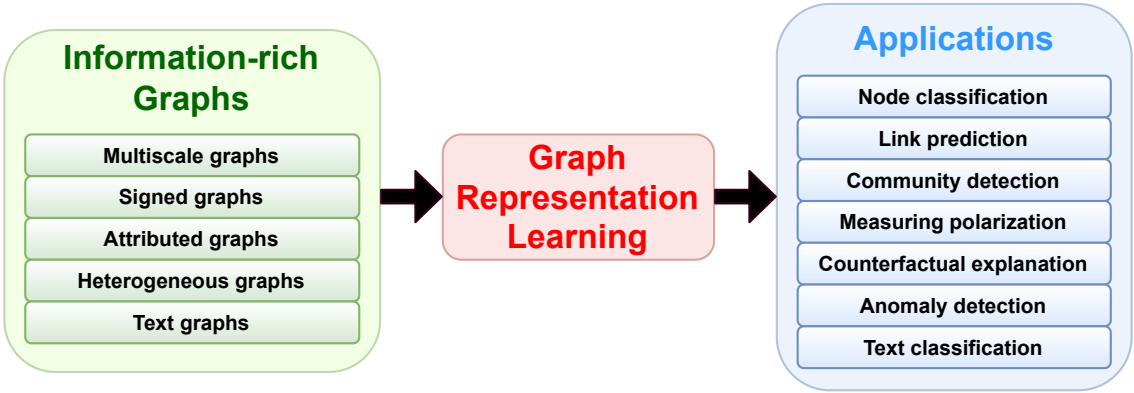
Graph structure: *relationship information*
Node and edge attributes: *descriptive information*

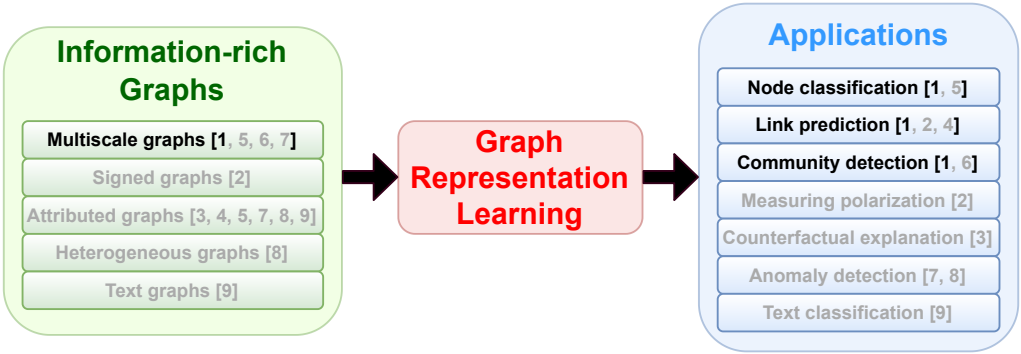
Representation learning for information-rich graphs



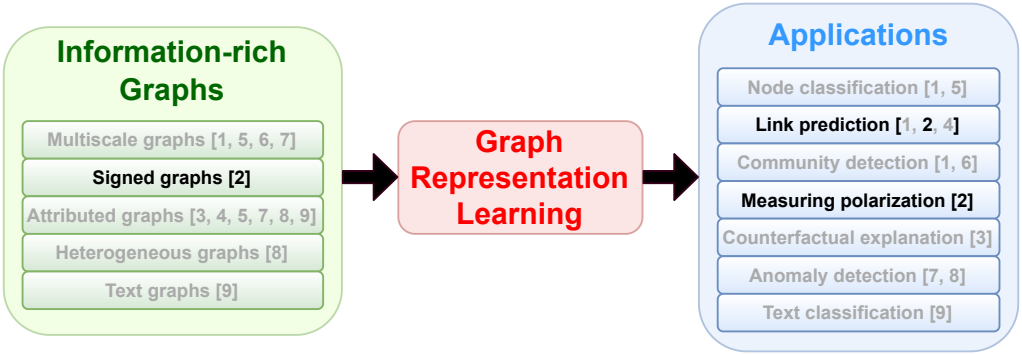
Graph structure: *relationship information*
Node and edge attributes: *descriptive information*

My research

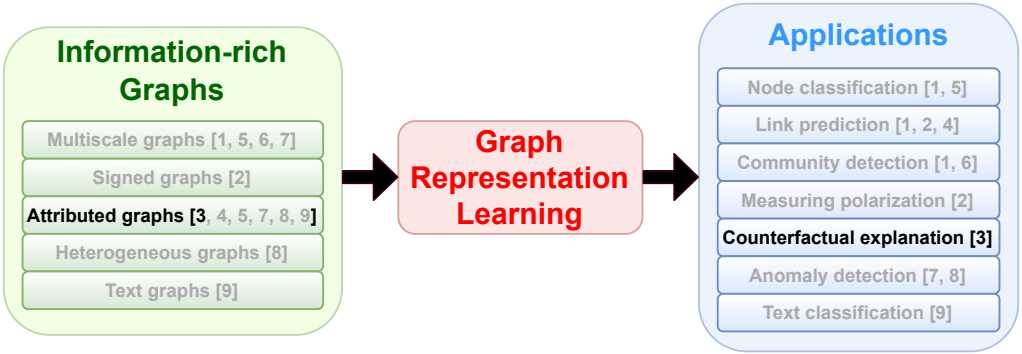

Information-rich Graphs
Multiscale graphs
Signed graphs
Attributed graphs
Heterogeneous graphs
Text graphs
Graph Representation Learning
Applications
Node classification
Link prediction
Community detection
Measuring polarization
Counterfactual explanation
Anomaly detection
Text classification



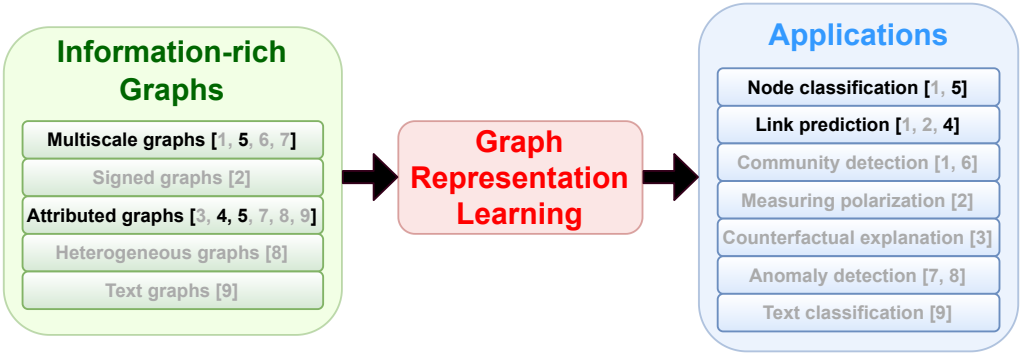
[1] **Huang**, Silva, Singh. A broader picture of random-walk based graph embedding. KDD'21.



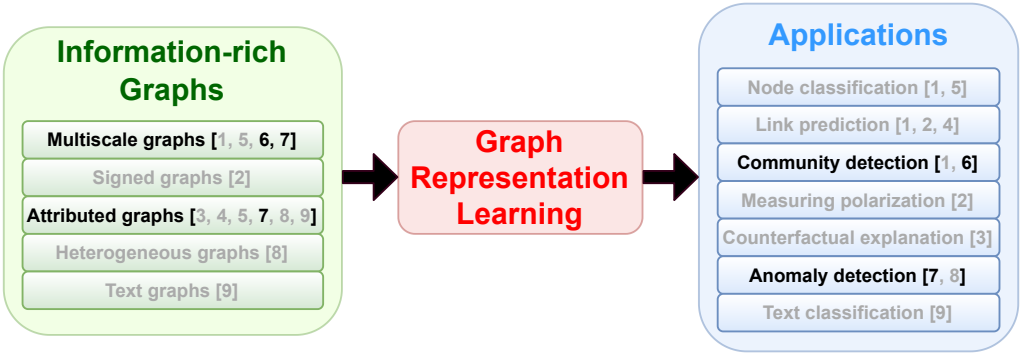
[1] **Huang**, Silva, Singh. A broader picture of random-walk based graph embedding. KDD'21.
[2] **Huang**, Silva, Singh. POLE: Polarized embedding for signed networks. WSDM'22.



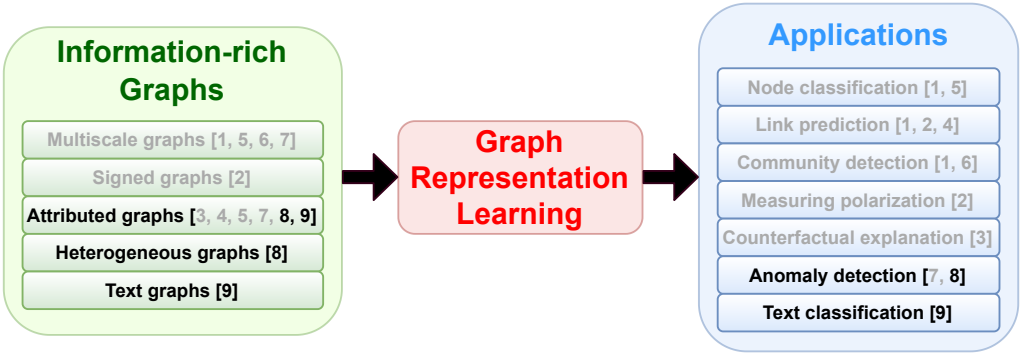
[1] **Huang**, Silva, Singh. A broader picture of random-walk based graph embedding. KDD'21.
[2] **Huang**, Silva, Singh. POLE: Polarized embedding for signed networks. WSDM'22.
[3] **Huang***, Kosan*, et al. GCFExplainer: Global counterfactual explainer for graph neural networks. WSDM'23.



[1] **Huang**, Silva, Singh. A broader picture of random-walk based graph embedding. KDD'21.
[2] **Huang**, Silva, Singh. POLE: Polarized embedding for signed networks. WSDM'22.
[3] **Huang***, Kosan*, et al. GCFExplainer: Global counterfactual explainer for graph neural networks. WSDM'23.
[4] **Huang**, Kosan, Silva, Singh. Link prediction without graph neural networks. Under review.
[5] Ye, **Huang**, Hong, Singh. Graph convolutional networks meet neural diffusions. Under review.



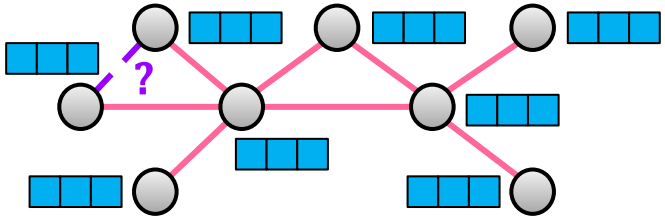
[1] **Huang**, Silva, Singh. A broader picture of random-walk based graph embedding. KDD'21.
[2] **Huang**, Silva, Singh. POLE: Polarized embedding for signed networks. WSDM'22.
[3] **Huang***, Kosan*, et al. GCFExplainer: Global counterfactual explainer for graph neural networks. WSDM'23.
[4] **Huang**, Kosan, Silva, Singh. Link prediction without graph neural networks. Under review.
[5] Ye, **Huang**, Hong, Singh. Graph convolutional networks meet neural diffusions. Under review.
[6] **Huang**, Kondapaneni, Silva, Singh. Multiscale community detection with pointwise mutual information. Ongoing.
[7] Arriola, Kosan, **Huang**, Sharma, Singh. Multiscale anomaly detection with graph autoencoders. Ongoing.



[1] **Huang**, Silva, Singh. A broader picture of random-walk based graph embedding. KDD'21.
[2] **Huang**, Silva, Singh. POLE: Polarized embedding for signed networks. WSDM'22.
[3] **Huang***, Kosan*, et al. GCFExplainer: Global counterfactual explainer for graph neural networks. WSDM'23.
[4] **Huang**, Kosan, Silva, Singh. Link prediction without graph neural networks. Under review.
[5] Ye, **Huang**, Hong, Singh. Graph convolutional networks meet neural diffusions. Under review.
[6] **Huang**, Kondapaneni, Silva, Singh. Multiscale community detection with pointwise mutual information. Ongoing.
[7] Arriola, Kosan, **Huang**, Sharma, Singh. Multiscale anomaly detection with graph autoencoders. Ongoing.
[8] **Huang**. Graph-based fraud detection in Kindle Direct Publishing. Amazon science internship'20,21.
[9] **Huang**. Graph-based text classification for Kindle content intelligence. Amazon science internship'22.

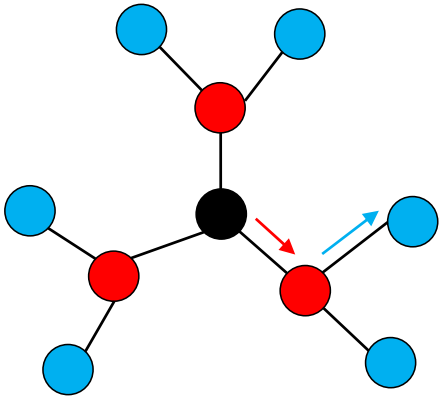
Outline

Link prediction

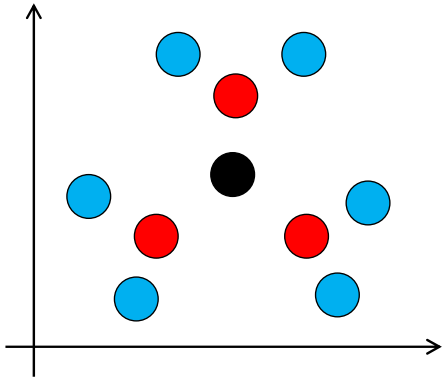


Link prediction: predicting the unobserved interactions (edges) between nodes given the observed graph structure (topology) and other information (e.g., node attributes).

Random-walk based node embedding

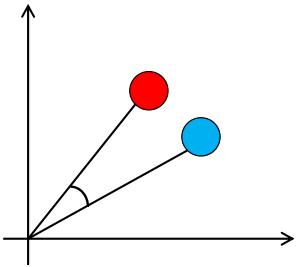


(a) Use random-walks to capture topological proximity (similarity)

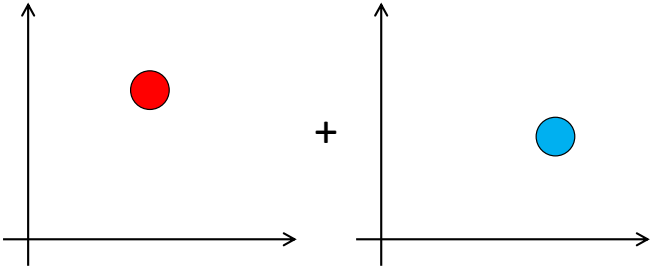


(b) Find low-dimensional latent vectors (embeddings) to preserve similarity

Link prediction based on node embedding

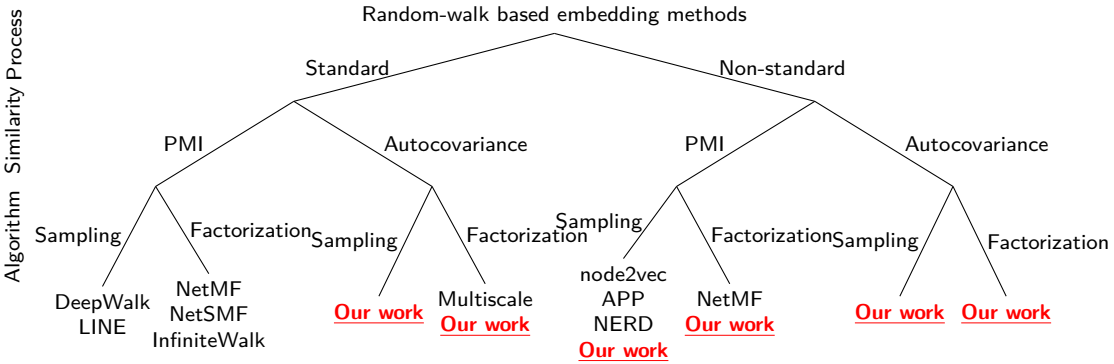


(a) Dot product [1, 2, 3]



(b) Classification based on combined embeddings [4, 5, 6]

A broader picture of random-walk based embedding[1]



- We present a unified framework for random-walk based embedding.
- We find that Autocovariance enables state-of-the-art link prediction.

[1] **Huang**, Silva, Singh. A broader picture of random-walk based graph embedding. KDD'21.

Random-walk based similarity metrics

M: transition matrix π, Π: stationary distribution τ: random-walk scale

Random-walk based similarity metrics

M: transition matrix π, Π: stationary distribution τ: random-walk scale

PMI: $R = \log(\Pi M^{\tau}) - \log(\pi\pi^{\top})$

Random-walk based similarity metrics

M: transition matrix π, Π: stationary distribution τ: random-walk scale

PMI: $R = \log(\Pi M^\tau) - \log(\pi\pi^\top)$ Autocovariance: $R = \Pi M^\tau - \pi\pi^\top$

Random-walk based similarity metrics

M: transition matrix π, Π: stationary distribution τ: random-walk scale

PMI: $R = \log(\Pi M^\tau) - \log(\pi\pi^\top)$ Autocovariance: $R = \Pi M^\tau - \pi\pi^\top$

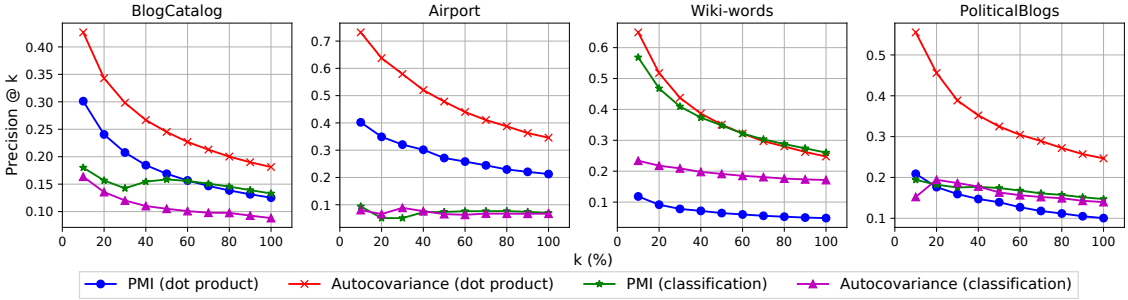


Figure: Autocovariance with dot product ranking consistently outperforms PMI (with either ranking scheme) in link prediction.

Understanding the difference

$$\text{predicted degree} \propto \text{embedding norm } \|\mathbf{u}\| \propto \begin{cases} \text{actual degree} & \text{for AC} \\ \text{constant} & \text{for PMI} \end{cases}$$

Understanding the difference

$$\text{predicted degree} \propto \text{embedding norm } \|\mathbf{u}\| \propto \begin{cases} \text{actual degree} & \text{for AC} \\ \text{constant} & \text{for PMI} \end{cases}$$

Autocovariance captures heterogeneous degree distribution in graphs!

Understanding the difference

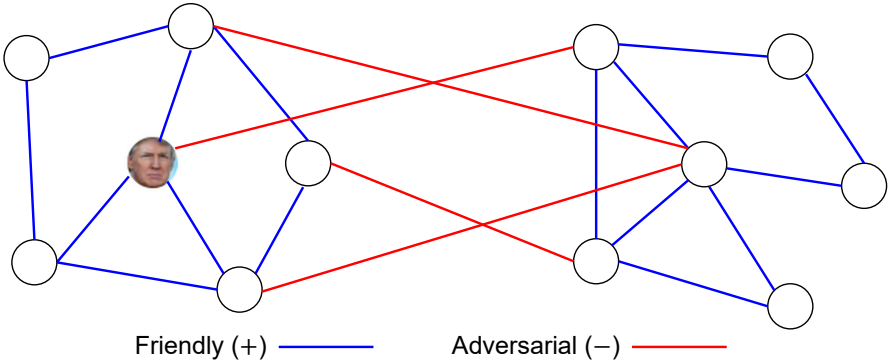
$$\text{predicted degree} \propto \text{embedding norm } \|\mathbf{u}\| \propto \begin{cases} \text{actual degree} & \text{for AC} \\ \text{constant} & \text{for PMI} \end{cases}$$

Autocovariance captures heterogeneous degree distribution in graphs!

(a) PMI (b) Autocovariance

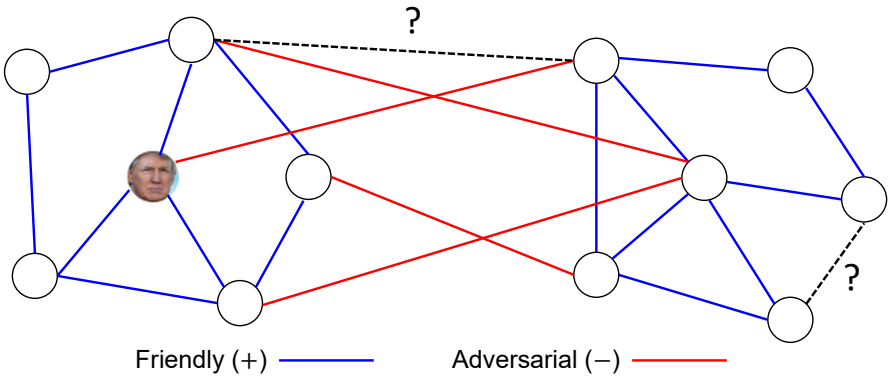
Figure: Autocovariance predicts more edges connecting to the hubs than PMI.

Polarized embedding for effective signed link prediction[2]



[2] **Huang**, Silva, Singh. POLE: Polarized embedding for signed networks. WSDM'22.

Polarized embedding for effective signed link prediction[2]



- We identify the key challenge of embedding polarized signed graphs.
- We develop polarized embedding for SOTA negative link prediction.

[2] **Huang**, Silva, Singh. POLE: Polarized embedding for signed networks. WSDM'22.

Signed link prediction in polarized networks

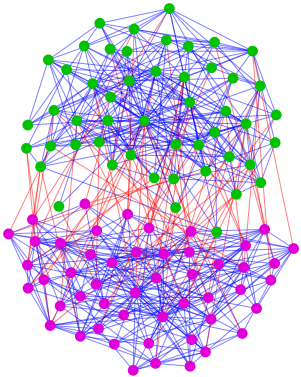


Figure: A synthetic polarized graph based on the LFR benchmark [7].

Signed link prediction in polarized networks

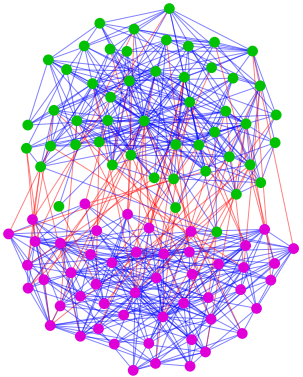


Figure: A synthetic polarized graph based on the LFR benchmark [7].

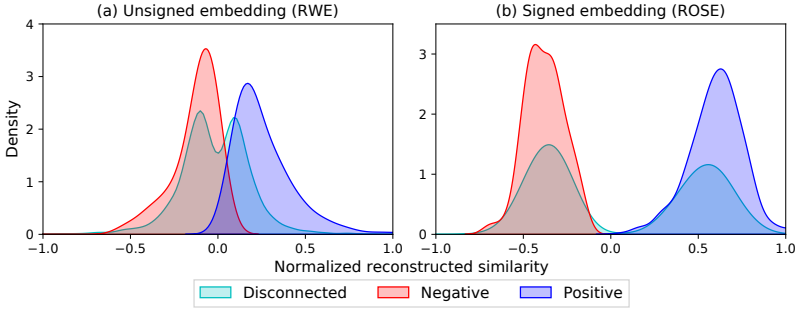


Figure: Distributions of the reconstructed similarity for different types of node pairs in the polarized graph using (a) unsigned [8] and (b) signed [6] embedding.

POLE: polarized embedding

- Signed random-walks to capture both similarities:

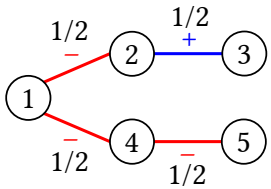
$$M_{uv}(t) = \sum_{\text{all length-}t \text{ paths } l \text{ between } u \text{ and } v} \text{Prob}(l)\text{Sign}(l)$$

POLE: polarized embedding

- Signed random-walks to capture both similarities:

$$M_{uv}(t) = \sum_{\text{all length-}t\text{ paths }l\text{ between }u\text{ and }v} \text{Prob}(l)\text{Sign}(l)$$

  - $\text{Prob}(l)$ captures the unsigned similarity.
  - $\text{Sign}(l)$ based on the social balance theory captures the signed similarity.



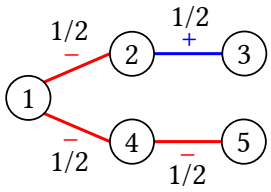
l	$\text{Prob}(l)$	$\text{Sign}(l)$
$\langle 1,2,3\rangle$	$1/2 \times 1/2 = 1/4$	$-1 \times 1 = -1$
$\langle 1,4,5\rangle$	$1/2 \times 1/2 = 1/4$	$-1 \times -1 = 1$

POLE: polarized embedding

- Signed random-walks to capture both similarities:

$$M_{uv}(t) = \sum_{\text{all length-}t\text{ paths }l\text{ between }u\text{ and }v} \text{Prob}(l)\text{Sign}(l)$$

  - $\text{Prob}(l)$ captures the unsigned similarity.
  - $\text{Sign}(l)$ based on the social balance theory captures the signed similarity.



l	$\text{Prob}(l)$	$\text{Sign}(l)$
$\langle 1,2,3\rangle$	$1/2 \times 1/2 = 1/4$	$-1 \times 1 = -1$
$\langle 1,4,5\rangle$	$1/2 \times 1/2 = 1/4$	$-1 \times -1 = 1$

- POLE: extends Autocovariance similarity [9, 8] to signed RW

$$R(t) = M(t)^\top W M(t)$$

Polarized similarity consistency

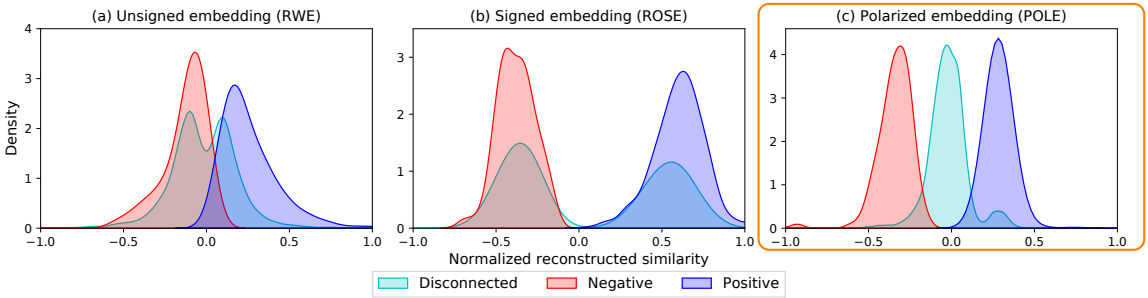


Figure: Polarized embedding (c) preserves *polarized similarity consistency*—negative pairs are separated from other pairs in the similarity spectrum—while unsigned embedding (a) and signed embedding (b) fail to do so.

Signed link prediction results

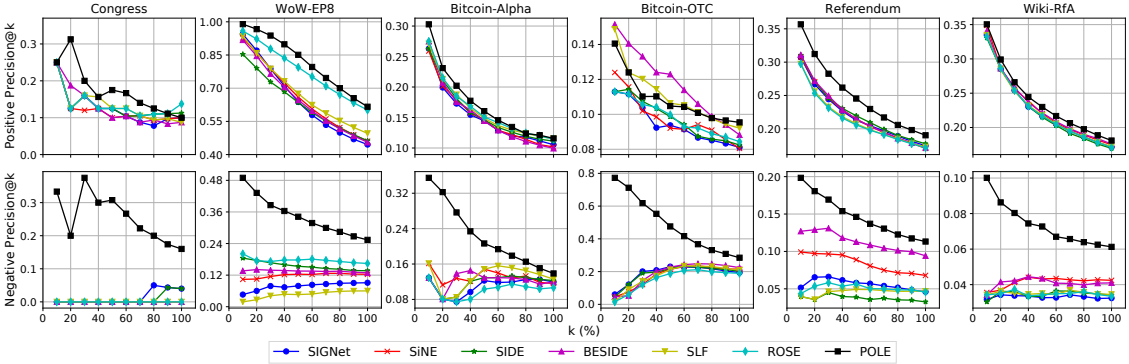
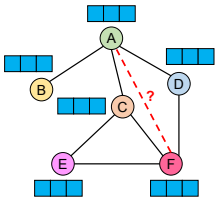


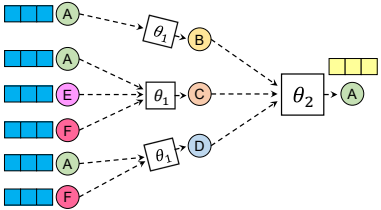
Figure: Signed link prediction with link existence information performance comparison. POLE outperforms all baselines in almost all datasets, especially for the negative links.

Attributed graph embedding and graph neural networks

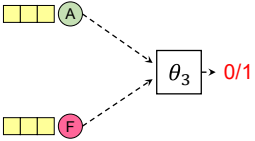
- GNNs [10, 11, 12] are a powerful DL paradigm that learns to generate better node features (embeddings) using structure information.



(a) Input attributed graph



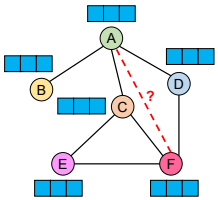
(b) GNN message-passing



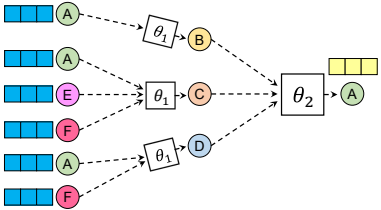
(c) Pooling and classification

Attributed graph embedding and graph neural networks

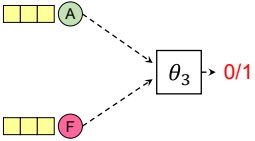
- GNNs [10, 11, 12] are a powerful DL paradigm that learns to generate better node features (embeddings) using structure information.



(a) Input attributed graph



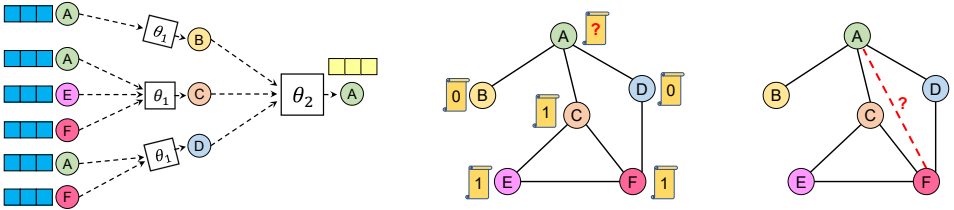
(b) GNN message-passing



(c) Pooling and classification

- Advantages over topological heuristics for link prediction:
  - Potential to discover new heuristics via **supervised learning**.
  - Natural incorporation of **node attribute** information.

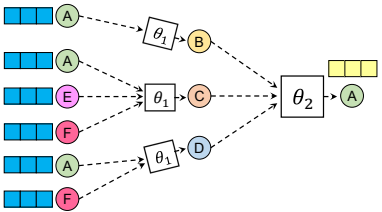
Challenges of GNNs for link prediction



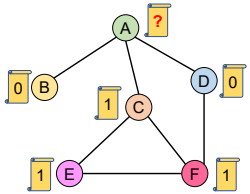
(a) GNN message-passing (b) Node classification (c) Link prediction

Figure: The attribute-centric message-passing mechanism is effective for tasks ***on*** the topology, e.g., node classification. Link prediction, however, is a task ***for*** the topology.

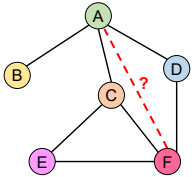
Challenges of GNNs for link prediction



(a) GNN message-passing



(b) Node classification



(c) Link prediction

Figure: The attribute-centric message-passing mechanism is effective for tasks ***on*** the topology, e.g., node classification. Link prediction, however, is a task ***for*** the topology.

Are there better alternatives to message-passing for combining node attributes and graph topology for link prediction?

Challenges of GNNs for link prediction

Table: Common real-world datasets for link prediction benchmark.

	#Nodes	#Edges	Avg. degree	Density	Class ratio
CORA	2,708	5,278	3.90	0.14%	**1:695**
CITESEER	3,327	4,552	2.74	0.08%	**1:1216**
PUBMED	19,717	44,324	4.50	0.02%	**1:4385**
PHOTO	7,650	119,081	31.13	0.41%	**1:246**
COMPUTERS	13,752	245,861	35.76	0.26%	**1:385**

Challenges of GNNs for link prediction

Table: Common real-world datasets for link prediction benchmark.

	#Nodes	#Edges	Avg. degree	Density	Class ratio
CORA	2,708	5,278	3.90	0.14%	**1:695**
CITESEER	3,327	4,552	2.74	0.08%	**1:1216**
PUBMED	19,717	44,324	4.50	0.02%	**1:4385**
PHOTO	7,650	119,081	31.13	0.41%	**1:246**
COMPUTERS	13,752	245,861	35.76	0.26%	**1:385**

Have GNN-based link prediction methods properly addressed the intrinsic class imbalance?

Supervised link prediction evaluation

- Existing work [13, 14, 15, 16, 17, 18, 19, 20, 21] adopts AUC and AP with *biased testing* (downsampling negative/disconnected pairs), which pictures an **overly optimistic** view of model performance.

Supervised link prediction evaluation

- Existing work [13, 14, 15, 16, 17, 18, 19, 20, 21] adopts AUC and AP with *biased testing* (downsampling negative/disconnected pairs), which pictures an **overly optimistic** view of model performance.
- We argue for evaluation under *unbiased testing*, which has been widely applied in unsupervised link prediction [1, 3, 8] and IR.

Supervised link prediction evaluation

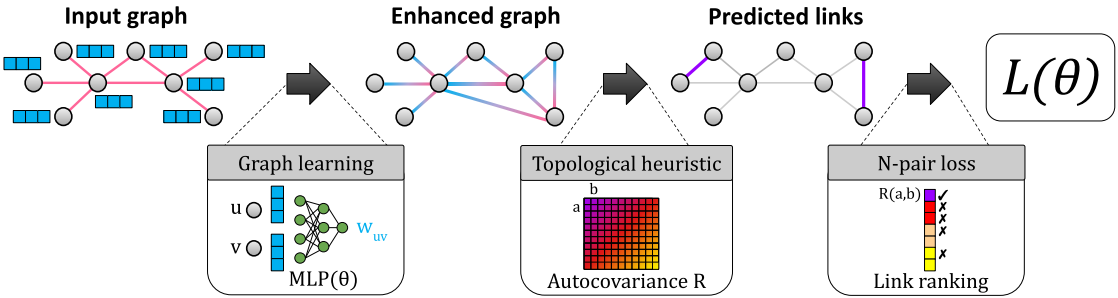
- Existing work [13, 14, 15, 16, 17, 18, 19, 20, 21] adopts AUC and AP with *biased testing* (downsampling negative/disconnected pairs), which pictures an **overly optimistic** view of model performance.
- We argue for evaluation under *unbiased testing*, which has been widely applied in unsupervised link prediction [1, 3, 8] and IR.

Supervised link prediction training

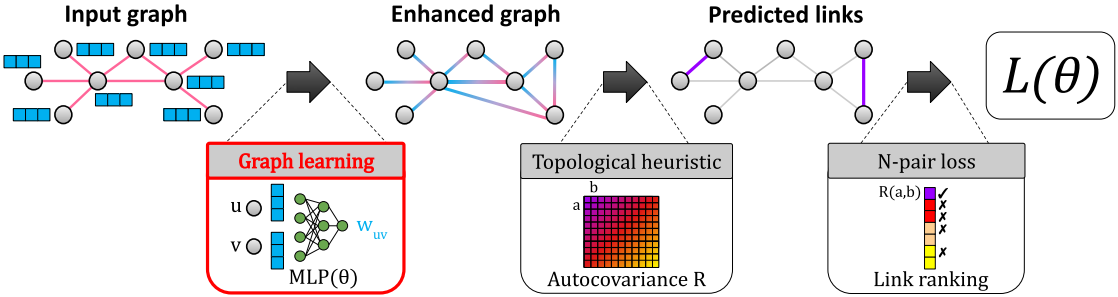
- Existing work uses binary cross entropy loss with *biased training*.
  - It **discards** potentially useful evidence from negative pairs.
  - It induces the model to **overestimate** the probability of positive pairs.

A simpler, faster, and stronger paradigm for link prediction[3]

- Explore alternative frameworks to combine topology and attributes
- Address the problem of class imbalance in training and testing

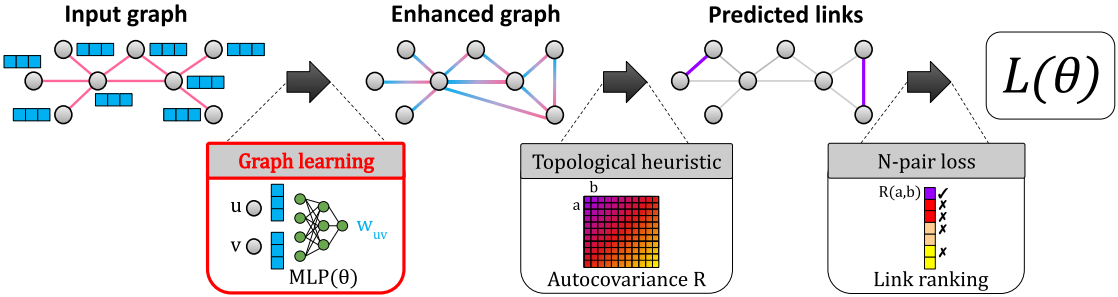


[3] **Huang**, Kosan, Silva, Singh. Link prediction without graph neural networks. Under review.



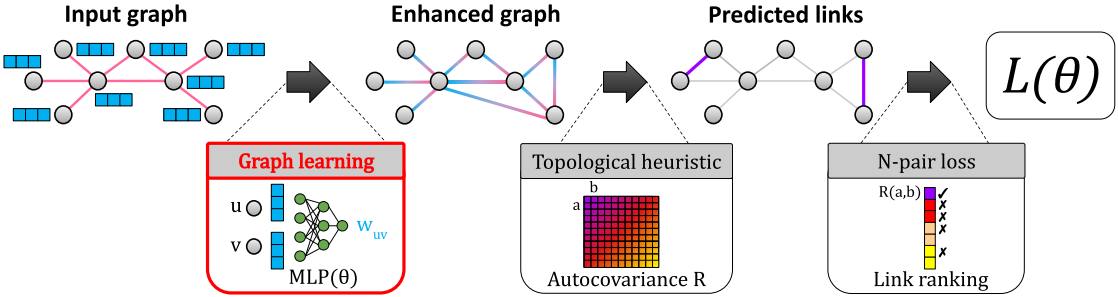
Graph learning

- Graph augmentation: $\widetilde{E} = E + \{(u, v) \mid s(x_u, x_v) > s_\eta\}$



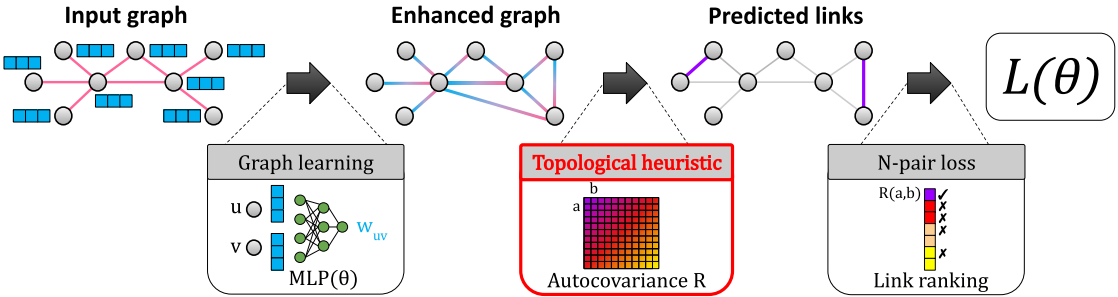
Graph learning

- Graph augmentation: $\widetilde{E} = E + \{(u, v) \mid s(x_u, x_v) > s_\eta\}$
- Trained weighting: $w_{uv} = \mathrm{MLP}([x_u + x_v; |x_u - x_v|]; \theta)$



Graph learning

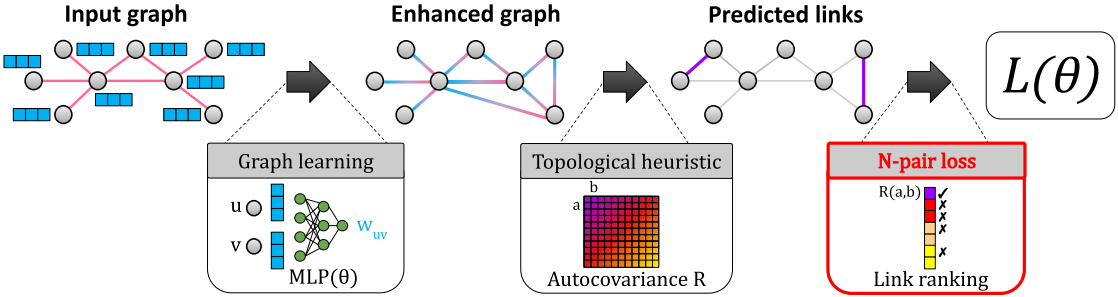
- Graph augmentation: $\widetilde{E} = E + \{(u, v) \mid s(x_u, x_v) > s_\eta\}$
- Trained weighting: $w_{uv} = \mathrm{MLP}([x_u + x_v; |x_u - x_v|]; \theta)$
- Combined weights: $\widetilde{A}_{uv} = \alpha A_{uv} + (1 - \alpha)(\beta w_{uv} + (1 - \beta) s(x_u, x_v))$



Topological heuristic

- Applying Autocovariance [22, 8] to the enhanced graph $\widetilde{A}$:

$$R = \frac{\widetilde{D}}{\mathrm{vol}(\widetilde{G})}(\widetilde{D}^{-1}\widetilde{A})^t - \frac{\tilde{d}\tilde{d}^\top}{\mathrm{vol}^2(\widetilde{G})}$$



N-pair loss [23]

- Contrasting each positive edge (u, v) with a set of negative pairs $N(u, v)$ whose size equals to the class ratio (*unbiased training*):

$$L(\theta) = -\sum_{(u,v)\in E} \log \frac{\exp(R_{uv})}{\exp(R_{uv}) + \sum_{(p,q)\in N(u,v)} \exp(R_{pq})}$$

Table: Link prediction performance comparison (mean ± std AP). Gelato outperforms the best GNN-based method, Neo-GNN, by **145%** and Autocovariance by **53%**.

* Run only once as each run takes ~100 hrs; *** Each run takes >1000 hrs; OOM: Out Of Memory.

		Cora	CiteSeer	PubMed	Photo	Computers
GNN	GAE	0.27 ± 0.02	0.66 ± 0.11	0.26 ± 0.03	0.28 ± 0.02	0.30 ± 0.02
	SEAL	1.89 ± 0.74	0.91 ± 0.66	***	10.49 ± 0.86	6.84*
	HGCN	0.82 ± 0.03	0.74 ± 0.10	0.35 ± 0.01	2.11 ± 0.10	2.30 ± 0.14
	LGCN	1.14 ± 0.04	0.86 ± 0.09	0.44 ± 0.01	3.53 ± 0.05	1.96 ± 0.03
	TLC-GNN	0.29 ± 0.09	0.35 ± 0.18	OOM	1.77 ± 0.11	OOM
	Neo-GNN	2.05 ± 0.61	1.61 ± 0.36	1.21 ± 0.14	10.83 ± 1.53	6.75*
	NBFNet	1.36 ± 0.17	0.77 ± 0.22	***	11.99 ± 1.60	***
	BScNets	0.32 ± 0.08	0.20 ± 0.06	0.22 ± 0.08	2.47 ± 0.18	1.45 ± 0.10
	WalkPool	2.04 ± 0.07	1.39 ± 0.11	1.31*	OOM	OOM

Table: Link prediction performance comparison (mean ± std AP). Gelato outperforms the best GNN-based method, Neo-GNN, by **145%** and Autocovariance by **53%**.

* Run only once as each run takes ~100 hrs; *** Each run takes >1000 hrs; OOM: Out Of Memory.

		Cora	CiteSeer	PubMed	Photo	Computers
GNN	GAE	0.27 ± 0.02	0.66 ± 0.11	0.26 ± 0.03	0.28 ± 0.02	0.30 ± 0.02
	SEAL	1.89 ± 0.74	0.91 ± 0.66	***	10.49 ± 0.86	6.84*
	HGCN	0.82 ± 0.03	0.74 ± 0.10	0.35 ± 0.01	2.11 ± 0.10	2.30 ± 0.14
	LGCN	1.14 ± 0.04	0.86 ± 0.09	0.44 ± 0.01	3.53 ± 0.05	1.96 ± 0.03
	TLC-GNN	0.29 ± 0.09	0.35 ± 0.18	OOM	1.77 ± 0.11	OOM
	Neo-GNN	2.05 ± 0.61	1.61 ± 0.36	1.21 ± 0.14	10.83 ± 1.53	6.75*
	NBFNet	1.36 ± 0.17	0.77 ± 0.22	***	11.99 ± 1.60	***
	BScNets	0.32 ± 0.08	0.20 ± 0.06	0.22 ± 0.08	2.47 ± 0.18	1.45 ± 0.10
	WalkPool	2.04 ± 0.07	1.39 ± 0.11	1.31*	OOM	OOM
Topological Heuristics	CN	1.10 ± 0.00	0.74 ± 0.00	0.36 ± 0.00	7.73 ± 0.00	5.09 ± 0.00
	AA	2.07 ± 0.00	1.24 ± 0.00	0.45 ± 0.00	9.67 ± 0.00	6.52 ± 0.00
	RA	2.02 ± 0.00	1.19 ± 0.00	0.33 ± 0.00	10.77 ± 0.00	7.71 ± 0.00
	AC	2.43 ± 0.00	2.65 ± 0.00	2.50 ± 0.00	16.63 ± 0.00	11.64 ± 0.00

Table: Link prediction performance comparison (mean ± std AP). Gelato outperforms the best GNN-based method, Neo-GNN, by **145%** and Autocovariance by **53%**.

* Run only once as each run takes ~100 hrs; *** Each run takes >1000 hrs; OOM: Out Of Memory.

		Cora	CiteSeer	PubMed	Photo	Computers
GNN	GAE	0.27 ± 0.02	0.66 ± 0.11	0.26 ± 0.03	0.28 ± 0.02	0.30 ± 0.02
	SEAL	1.89 ± 0.74	0.91 ± 0.66	***	10.49 ± 0.86	6.84*
	HGCN	0.82 ± 0.03	0.74 ± 0.10	0.35 ± 0.01	2.11 ± 0.10	2.30 ± 0.14
	LGCN	1.14 ± 0.04	0.86 ± 0.09	0.44 ± 0.01	3.53 ± 0.05	1.96 ± 0.03
	TLC-GNN	0.29 ± 0.09	0.35 ± 0.18	OOM	1.77 ± 0.11	OOM
	Neo-GNN	2.05 ± 0.61	1.61 ± 0.36	1.21 ± 0.14	10.83 ± 1.53	6.75*
	NBFNet	1.36 ± 0.17	0.77 ± 0.22	***	11.99 ± 1.60	***
	BScNets	0.32 ± 0.08	0.20 ± 0.06	0.22 ± 0.08	2.47 ± 0.18	1.45 ± 0.10
	WalkPool	2.04 ± 0.07	1.39 ± 0.11	1.31*	OOM	OOM
Topological Heuristics	CN	1.10 ± 0.00	0.74 ± 0.00	0.36 ± 0.00	7.73 ± 0.00	5.09 ± 0.00
	AA	2.07 ± 0.00	1.24 ± 0.00	0.45 ± 0.00	9.67 ± 0.00	6.52 ± 0.00
	RA	2.02 ± 0.00	1.19 ± 0.00	0.33 ± 0.00	10.77 ± 0.00	7.71 ± 0.00
	AC	2.43 ± 0.00	2.65 ± 0.00	2.50 ± 0.00	16.63 ± 0.00	11.64 ± 0.00
Attributes + Topology	MLP	0.30 ± 0.05	0.44 ± 0.09	0.14 ± 0.06	1.01 ± 0.26	0.41 ± 0.23
	Cos	0.42 ± 0.00	1.89 ± 0.00	0.07 ± 0.00	0.11 ± 0.00	0.07 ± 0.00
	MLP+AC	3.24 ± 0.03	1.95 ± 0.05	2.61 ± 0.06	15.99 ± 0.21	11.25 ± 0.13
	Cos+AC	3.60 ± 0.00	4.46 ± 0.00	0.51 ± 0.00	10.01 ± 0.00	5.20 ± 0.00
	MLP+Cos+AC	3.39 ± 0.06	4.15 ± 0.14	0.55 ± 0.03	10.88 ± 0.09	5.75 ± 0.11

Table: Link prediction performance comparison (mean ± std AP). Gelato outperforms the best GNN-based method, Neo-GNN, by **145%** and Autocovariance by **53%**.

* Run only once as each run takes ~100 hrs; *** Each run takes >1000 hrs; OOM: Out Of Memory.

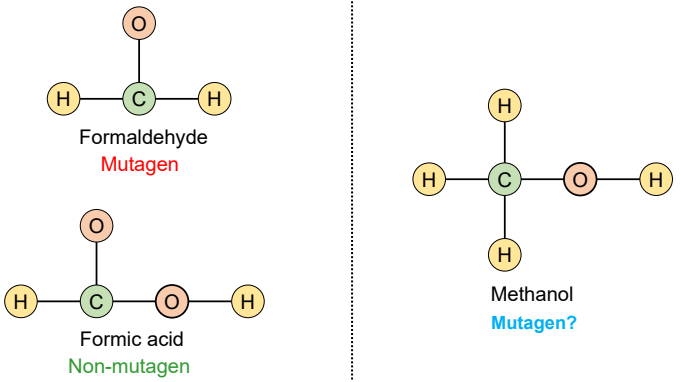
		Cora	CiteSeer	PubMed	Photo	Computers
GNN	GAE	0.27 ± 0.02	0.66 ± 0.11	0.26 ± 0.03	0.28 ± 0.02	0.30 ± 0.02
	SEAL	1.89 ± 0.74	0.91 ± 0.66	***	10.49 ± 0.86	6.84*
	HGCN	0.82 ± 0.03	0.74 ± 0.10	0.35 ± 0.01	2.11 ± 0.10	2.30 ± 0.14
	LGCN	1.14 ± 0.04	0.86 ± 0.09	0.44 ± 0.01	3.53 ± 0.05	1.96 ± 0.03
	TLC-GNN	0.29 ± 0.09	0.35 ± 0.18	OOM	1.77 ± 0.11	OOM
	Neo-GNN	2.05 ± 0.61	1.61 ± 0.36	1.21 ± 0.14	10.83 ± 1.53	6.75*
	NBFNet	1.36 ± 0.17	0.77 ± 0.22	***	11.99 ± 1.60	***
	BScNets	0.32 ± 0.08	0.20 ± 0.06	0.22 ± 0.08	2.47 ± 0.18	1.45 ± 0.10
	WalkPool	2.04 ± 0.07	1.39 ± 0.11	1.31*	OOM	OOM
Topological Heuristics	CN	1.10 ± 0.00	0.74 ± 0.00	0.36 ± 0.00	7.73 ± 0.00	5.09 ± 0.00
	AA	2.07 ± 0.00	1.24 ± 0.00	0.45 ± 0.00	9.67 ± 0.00	6.52 ± 0.00
	RA	2.02 ± 0.00	1.19 ± 0.00	0.33 ± 0.00	10.77 ± 0.00	7.71 ± 0.00
	AC	2.43 ± 0.00	2.65 ± 0.00	2.50 ± 0.00	16.63 ± 0.00	11.64 ± 0.00
Attributes + Topology	MLP	0.30 ± 0.05	0.44 ± 0.09	0.14 ± 0.06	1.01 ± 0.26	0.41 ± 0.23
	Cos	0.42 ± 0.00	1.89 ± 0.00	0.07 ± 0.00	0.11 ± 0.00	0.07 ± 0.00
	MLP+AC	3.24 ± 0.03	1.95 ± 0.05	2.61 ± 0.06	15.99 ± 0.21	11.25 ± 0.13
	Cos+AC	3.60 ± 0.00	4.46 ± 0.00	0.51 ± 0.00	10.01 ± 0.00	5.20 ± 0.00
	MLP+Cos+AC	3.39 ± 0.06	4.15 ± 0.14	0.55 ± 0.03	10.88 ± 0.09	5.75 ± 0.11
Gelato		**3.90 ± 0.03**	**4.55 ± 0.02**	**2.88 ± 0.09**	**25.68 ± 0.53**	**18.77 ± 0.19**

Section summary:

- We investigated the problem of representation learning for effective link prediction in scale-free, signed, and attributed networks.
- We scrutinized popular representation learning paradigms (node embedding and GNNs) and proposed novel solutions leveraging random-walk dynamics, social theories, and graph learning.
- Our methods enable state-of-the-art link prediction performance as verified in various real-world datasets.

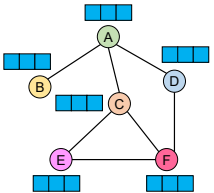
Outline

Graph classification

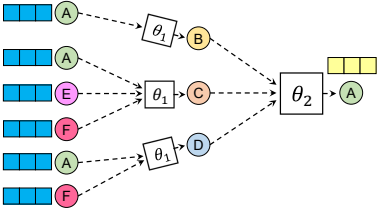


Graph classification: Given a collection of graphs, predict the label of each unknown graph based on known graphs and labels.

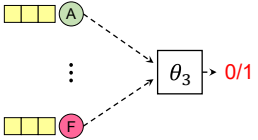
Graph neural networks for graph classification



(a) Input attributed graph



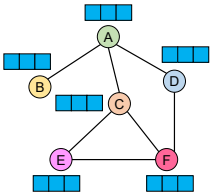
(b) GNN message-passing



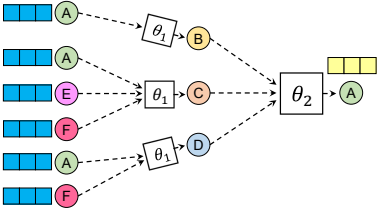
(c) Pooling and classification

- GNNs achieve state-of-the-art performance in graph classification.

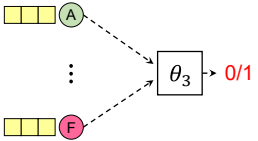
Graph neural networks for graph classification



(a) Input attributed graph



(b) GNN message-passing



(c) Pooling and classification

- GNNs achieve state-of-the-art performance in graph classification.

How can we understand the prediction made by GNNs in high-stakes decision making?

Model understanding and explanation

- Deep predictive model

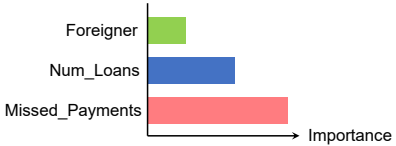
{Foreigner=True, Num_Loans=1, Missed_Payments=2} ⇒ Denied

Model understanding and explanation

- Deep predictive model

{Foreigner=True, Num_Loans=1, Missed_Payments=2} ⇒ Denied

- Feature importance [24, 25]

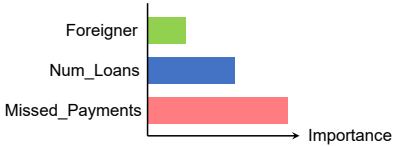


Model understanding and explanation

- Deep predictive model

{Foreigner=True, Num_Loans=1, Missed_Payments=2} ⇒ Denied

- Feature importance [24, 25]



- Local counterfactual [26, 27]

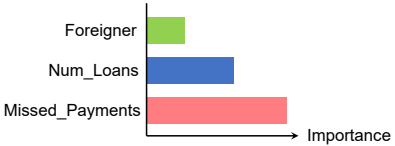
{Foreigner=True, Num_Loans=1, Missed_Payments=**1**} ⇒ Approved

Model understanding and explanation

- Deep predictive model

{Foreigner=True, Num_Loans=1, Missed_Payments=2} ⇒ Denied

- Feature importance [24, 25]



- Local counterfactual [26, 27]

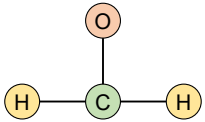
{Foreigner=True, Num_Loans=1, Missed_Payments=**1**} ⇒ Approved

- Global counterfactual [28]

If Foreigner=True *and* Num_Loans≥1 *then* Missed_Payments≤1 *for* approval

Understanding GNNs

- Graph classification GNN

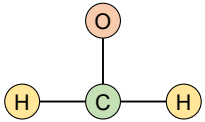


Formaldehyde

⇨ Mutagen

Understanding GNNs

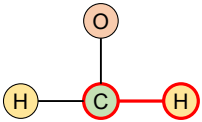
- Graph classification GNN



Formaldehyde

Mutagen

- Subgraph importance [29, 30]

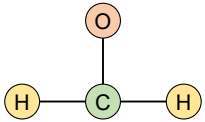


Formaldehyde

Important subgraph

Understanding GNNs

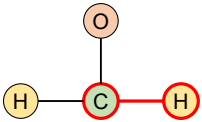
- Graph classification GNN



Formaldehyde

Mutagen

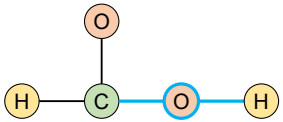
- Subgraph importance [29, 30]



Formaldehyde

Important subgraph

- Local counterfactual [31, 32]

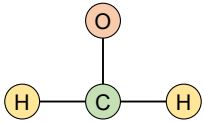


Formic acid

Non-mutagen

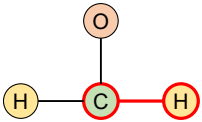
Understanding GNNs

- Graph classification GNN



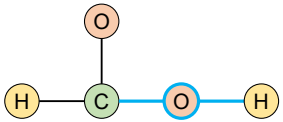
Mutagen

- Subgraph importance [29, 30]



Important subgraph

- Local counterfactual [31, 32]



Non-mutagen

Can we generate global counterfactual explanation for GNNs?

First global graph counterfactual explainer for GNNs[4]

[4] **Huang***, Kosan*, et al. Global counterfactual explainer for graph neural networks. WSDM'23.

First global graph counterfactual explainer for GNNs[4]

- Global recourse: For any undesired graph $G \in \mathcal{G}$ ($\mathbf{GNN}(G) = 0$), the explanation r should provide a recourse: $\mathbf{GNN}(r(G)) = 1$.
- Interpretable: r should be (much) easier to understand (than **GNN**).

[4] **Huang***, Kosan*, et al. Global counterfactual explainer for graph neural networks. WSDM'23.

First global graph counterfactual explainer for GNNs[4]

- Global recourse: For any undesired graph $G \in \mathcal{G}$ ($\mathbf{GNN}(G) = 0$), the explanation r should provide a recourse: $\mathbf{GNN}(r(G)) = 1$.
- Interpretable: r should be (much) easier to understand (than **GNN**).

Explanation based on counterfactual summary

- Represent r with a summary set $\mathcal{C}$ of counterfactual graphs.
- The recourse for G is the minimal cost (distance) summary graph:

$$r_{\mathcal{C}}(G) = \arg\min_{C \in \mathcal{C}} \mathrm{dist}(G, C)$$

where $\mathrm{dist}(\cdot, \cdot)$ is any distance metric between graphs (e.g., GED).

[4] **Huang***, Kosan*, et al. Global counterfactual explainer for graph neural networks. WSDM'23.

Quantifying explanation quality

- Cost: minimize the overall recourse cost for all undesired graphs:

$$\textbf{cost}(r_{\mathcal{C}}) = \sum_{G \in \mathcal{G}} \min_{C \in \mathcal{C}} \text{dist}(G, C)$$

Quantifying explanation quality

- Cost: minimize the overall recourse cost for all undesired graphs:

$$\textbf{cost}(r_{\mathcal{C}}) = \sum_{G \in \mathcal{G}} \min_{C \in \mathcal{C}} \text{dist}(G, C)$$

- Coverage: maximize the number of undesired graphs that have an actionable recourse (i.e., within a cost budget B):

$$\textbf{cover}(r_{\mathcal{C}}) = |\{G \in \mathcal{G} \mid \min_{C \in \mathcal{C}} \text{dist}(G, C) \leq B\}|$$

Quantifying explanation quality

- Cost: minimize the overall recourse cost for all undesired graphs:

$$\textbf{cost}(r_{\mathcal{C}}) = \sum_{G \in \mathcal{G}} \min_{C \in \mathcal{C}} \text{dist}(G, C)$$

- Coverage: maximize the number of undesired graphs that have an actionable recourse (i.e., within a cost budget B):

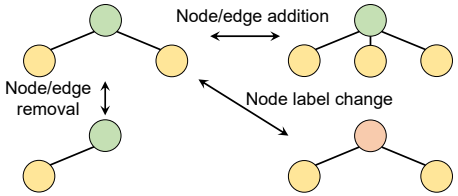
$$\textbf{cover}(r_{\mathcal{C}}) = |\{G \in \mathcal{G} \mid \min_{C \in \mathcal{C}} \text{dist}(G, C) \leq B\}|$$

- Interpretability: minimize the size of the counterfactual summary:

$$\textbf{size}(r_{\mathcal{C}}) = |\mathcal{C}|$$

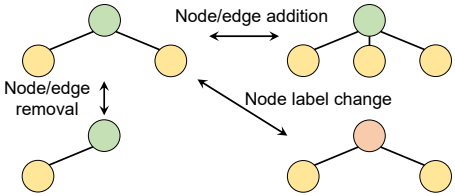
Structuring counterfactual summary search space

- Graph edit map: A (meta)graph of candidate graphs connected by single graph edits.



Structuring counterfactual summary search space

- Graph edit map: A (meta)graph of candidate graphs connected by single graph edits.

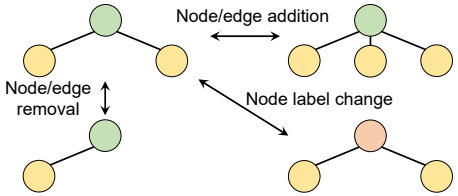


Generating diverse counterfactual summary

- Vertex-reinforced random-walk: Converges to diverse graphs [33].

Structuring counterfactual summary search space

- Graph edit map: A (meta)graph of candidate graphs connected by single graph edits.

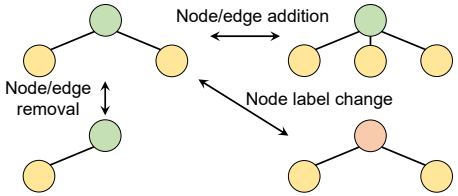


Generating diverse counterfactual summary

- Vertex-reinforced random-walk: Converges to diverse graphs [33].
- Guided transitions: Biases towards good counterfactual summaries.

Structuring counterfactual summary search space

- Graph edit map: A (meta)graph of candidate graphs connected by single graph edits.



Generating diverse counterfactual summary

- Vertex-reinforced random-walk: Converges to diverse graphs [33].
- Guided transitions: Biases towards good counterfactual summaries.
- Random teleportation back: Manages the exponential search space.

Comparison with summary of local counterfactuals

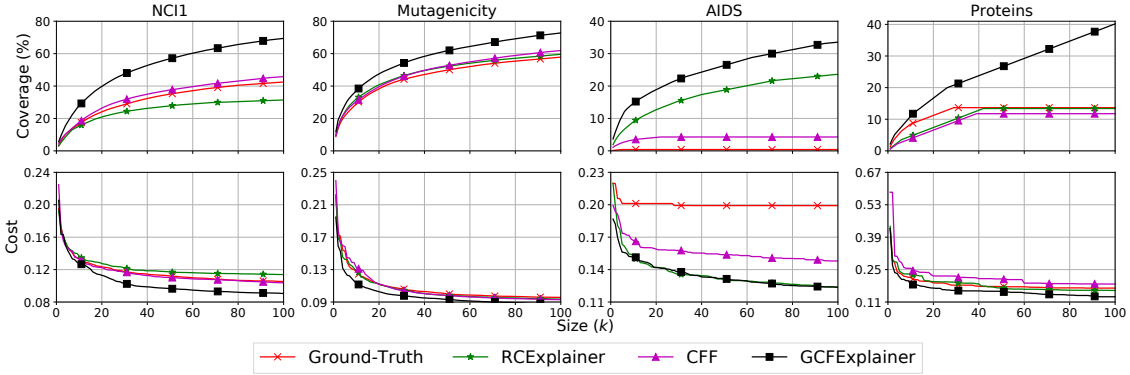


Figure: Global counterfactual quality comparison. GCFExplainer consistently outperforms the baselines with higher coverage and lower cost across different sizes.

Visualizing global counterfactuals

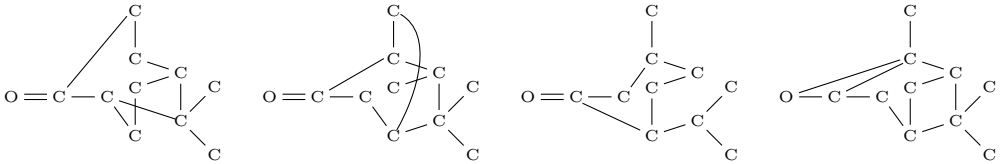


Figure: The global counterfactual explanation graph (third row) presents a high-level recourse rule—changing ketones and ethers into aldehydes—to combat HIV, while the edge removals in local counterfactual examples (second row) are hard to generalize.

Visualizing global counterfactuals

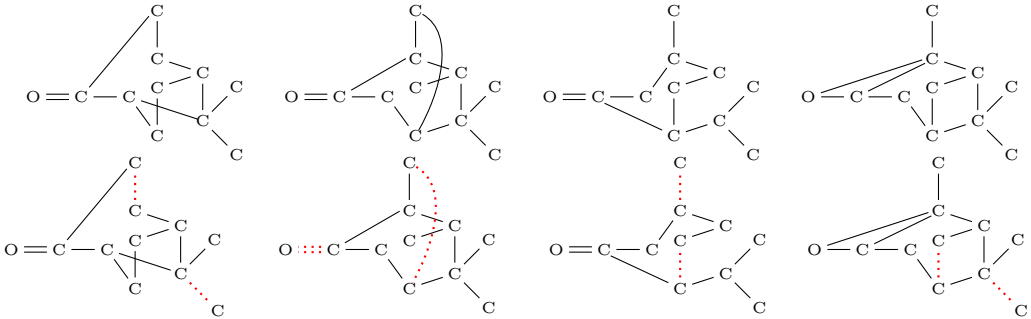


Figure: The global counterfactual explanation graph (third row) presents a high-level recourse rule—changing ketones and ethers into aldehydes—to combat HIV, while the edge removals in local counterfactual examples (second row) are hard to generalize.

Visualizing global counterfactuals

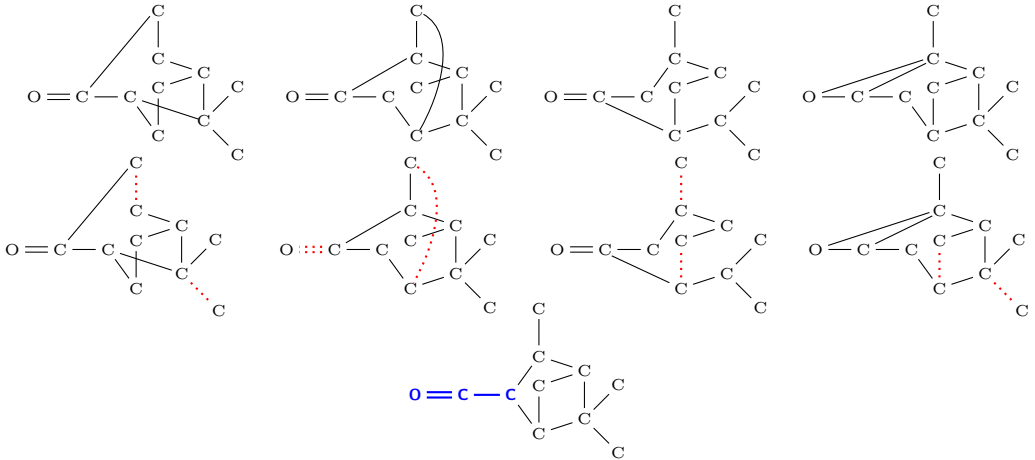


Figure: The global counterfactual explanation graph (third row) presents a high-level recourse rule—changing ketones and ethers into aldehydes—to combat HIV, while the edge removals in local counterfactual examples (second row) are hard to generalize.

Section summary:

- We formulate the novel problem of global counterfactual reasoning and explanation of GNNs for graph classification.
- We present GCFExplainer, the first global counterfactual explainer for GNNs that generates representative counterfactual summaries.
- We demonstrate the effectiveness and usefulness of GCFExplainer in providing high-level recourse for GNN-based graph classification, in the context of drug property prediction and discovery.

Outline

Multiscale graphs

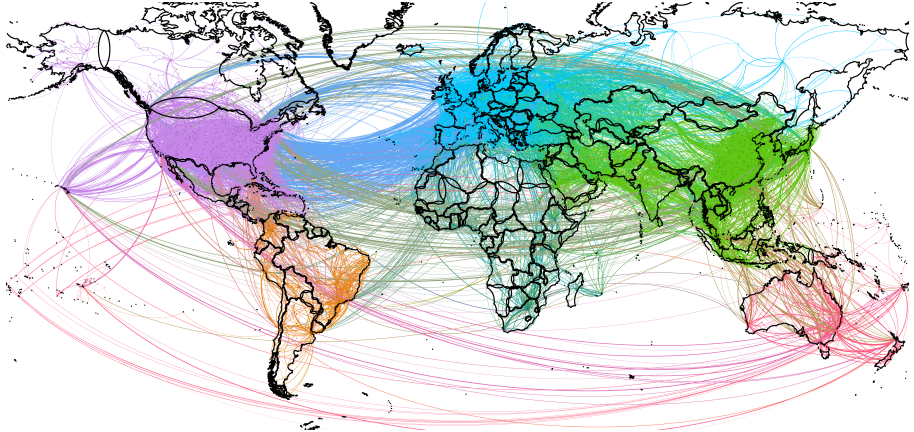


Figure: Global network of airports connected by domestic and international fights. Nodes in multiscale graphs form dense clusters at different structural scales (countries and continents here).

Graph neural diffusion networks[5]

- Multiscale message-passing as graph diffusions:

$$u^{(K)} = \sum_{k=0}^{K-1} \alpha_k M^k u^{(0)}$$

- Existing work [34, 35] adopts fixed diffusion weights α_k.

[5] Ye, **Huang**, Hong, Singh. Graph convolutional networks meet neural diffusions. Under review.

Graph neural diffusion networks[5]

- Multiscale message-passing as graph diffusions:

$$u^{(K)} = \sum_{k=0}^{K-1} \alpha_k M^k u^{(0)}$$

- Existing work [34, 35] adopts fixed diffusion weights α_k.
- We propose to learn the diffusion weights directly from data:

$$u^{(K)} = f([u^{(0)}; Mu^{(0)}; ...; M^{K-1}u^{(0)}]; \theta)$$

- Key result: The learned weights are adaptable to different datasets, leading to better semi-supervised node classification performance.

[5] Ye, **Huang**, Hong, Singh. Graph convolutional networks meet neural diffusions. Under review.

Multiscale community detection[6]

- $R^{AC}(\tau) = \Pi P(\tau) - \pi\pi^\top$ and $R^{PMI}(\tau) = \log(\Pi P(\tau)) - \log(\pi\pi^\top)$

[6] **Huang**, Kondapaneni, Silva, Singh. Multiscale community detection with pointwise mutual information. Ongoing.

Multiscale community detection[6]

- $R^{AC}(\tau) = \Pi P(\tau) - \pi\pi^\top$ and $R^{PMI}(\tau) = \log(\Pi P(\tau)) - \log(\pi\pi^\top)$
- Clustered similarity as a quality function: $r(C;\tau) = \sum_{i,j\in C} R_{ij}(\tau)$

[6] **Huang**, Kondapaneni, Silva, Singh. Multiscale community detection with pointwise mutual information. Ongoing.

Multiscale community detection[6]

- $R^{AC}(\tau) = \Pi P(\tau) - \pi\pi^\top$ and $R^{PMI}(\tau) = \log(\Pi P(\tau)) - \log(\pi\pi^\top)$
- Clustered similarity as a quality function: $r(C;\tau) = \sum_{i,j\in C} R_{ij}(\tau)$
- Markov Stability [22] finds the best communities $\{C_k\}$ by maximizing

$$r^{AC}(C_1, \cdots, C_c; \tau) = \sum_{k=1}^{c} \sum_{i,j\in C_k} R_{ij}^{AC}(\tau)$$

which assumes a fixed scale for each community specified by the user.

[6] **Huang**, Kondapaneni, Silva, Singh. Multiscale community detection with pointwise mutual information. Ongoing.

Multiscale community detection[6]

- $R^{AC}(\tau) = \Pi P(\tau) - \pi\pi^\top$ and $R^{PMI}(\tau) = \log(\Pi P(\tau)) - \log(\pi\pi^\top)$
- Clustered similarity as a quality function: $r(C;\tau) = \sum_{i,j\in C} R_{ij}(\tau)$
- Markov Stability [22] finds the best communities $\{C_k\}$ by maximizing

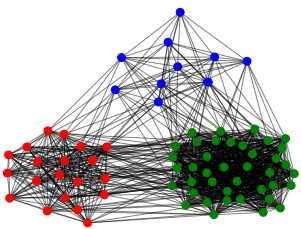
$$r^{AC}(C_1, \cdots, C_c; \tau) = \sum_{k=1}^{c} \sum_{i,j\in C_k} R_{ij}^{AC}(\tau)$$

  which assumes a fixed scale for each community specified by the user.
- We find $\{C_k\}$ along with their natural scales $\{\tau_k\}$ based on PMI:

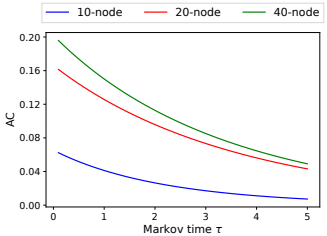
$$r^{PMI}(C_1, \cdots, C_c; \tau_1, \cdots, \tau_c) = \sum_{k=1}^{c} \sum_{i,j\in C_k} R_{ij}^{PMI}(\tau_k)$$

[6] **Huang**, Kondapaneni, Silva, Singh. Multiscale community detection with pointwise mutual information. Ongoing.

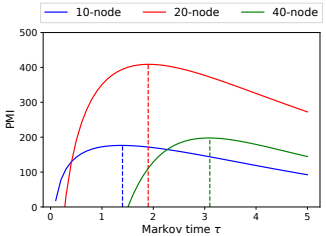
Empirical observation



(a) SBM-$(10, 20, 40)$



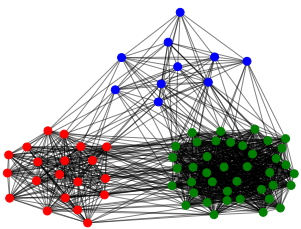
(b) AC



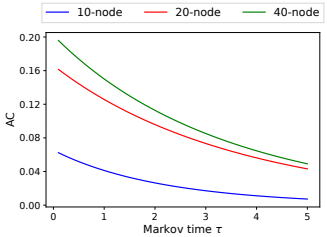
(c) PMI

Figure: Quality functions based on PMI for communities of different sizes reach unique peaks at different τ, revealing their natural scales, while those based on AC cannot.

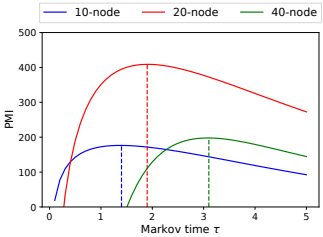
Empirical observation



(a) SBM-$(10, 20, 40)$



(b) AC



(c) PMI

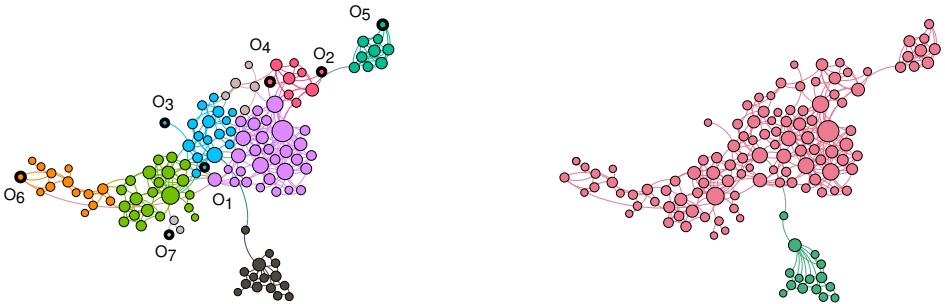
Figure: Quality functions based on PMI for communities of different sizes reach unique peaks at different τ, revealing their natural scales, while those based on AC cannot.

Theoretical analysis

- We have shown that $r^{AC}(C;\tau)$ monotonously decreases with τ.
- For $r^{PMI}(C;\tau)$, monotonicity has only been proved for some cases.

Multiscale anomaly detection[7]

- Existing work focuses on **node-level** anomaly within a particular context [36, 37, 38, 39] or multiscale contexts [40, 41].



(a) Node-level anomalies

(b) Subgraph-level anomalies

Figure: Real-world graphs have anomalies of different scales. Figures from [40].

[7] Arriola, Kosan, **Huang**, Singh. Multiscale anomaly detection with graph autoencoders. Ongoing.

Spectral energy distribution patterns for anomalies

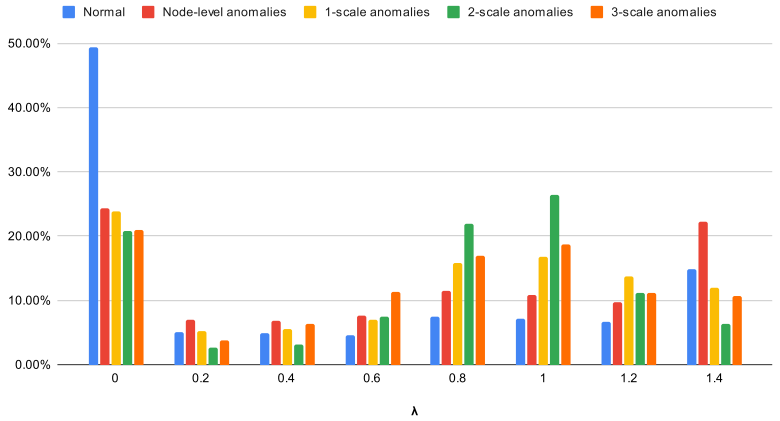
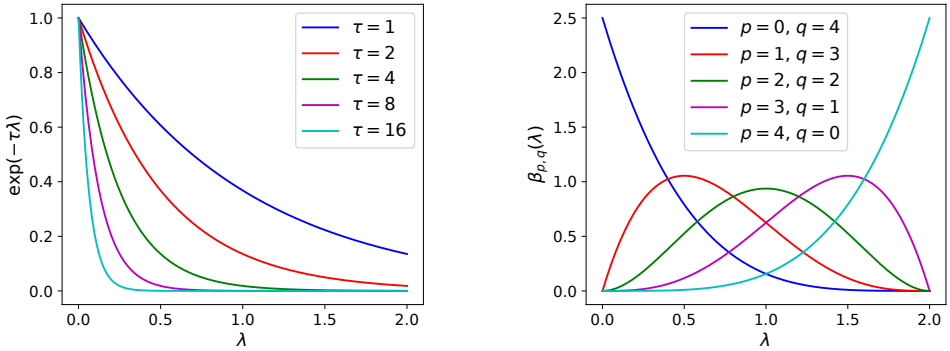


Figure: Compared to normal nodes, the spectral energy distributions of anomalous elements concentrate more on the high-frequency regions. Further, the smaller the scale of the anomalies, the higher the frequency bands they dominate.

Spectral localized Beta Wavelet GNNs



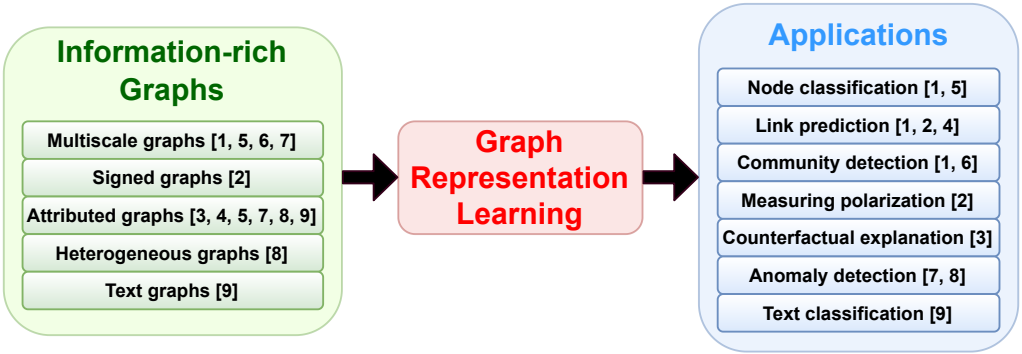
(a) Heat kernels

(b) Beta kernels

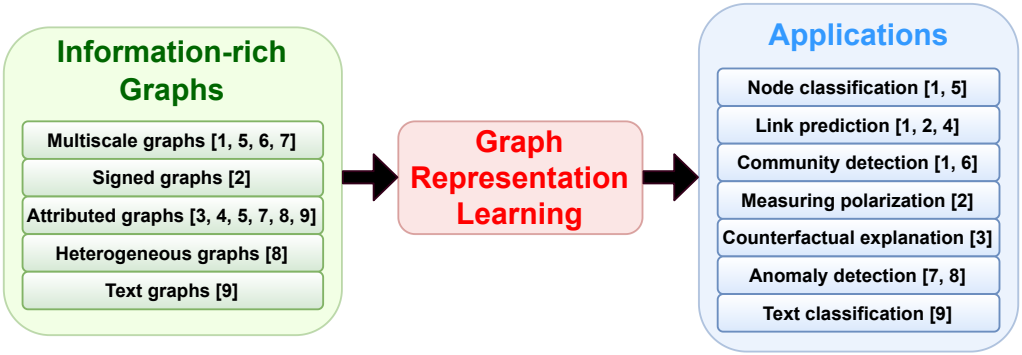
Figure: Spectral property comparison between heat kernels and Beta kernels. Beta kernels contain different band-pass filters that facilitate multiscale anomaly detection.

Outline

This dissertation demonstrates the importance of accounting for the interplay between the rich graph information and downstream task properties in graph representation learning.



This dissertation demonstrates the importance of accounting for the interplay between the rich graph information and downstream task properties in graph representation learning.



- Can we develop a universal architecture for different graphs & tasks?
- Can we understand/explain the model in the embedding space?

Acknowledgements

Advisor: Ambuj Singh

PhD Committee: Yu-Xiang Wang, Xifeng Yan

Collaborators: Arlei Silva, Mert Kosan, Sourav Medya, Sayan Ranu, Wei Ye, Manu Kondapaneni, Marianne Arriola, Saurabh Sharma

Labmates: Rachel, Richika, Chandana, Ashwini, Haraldur, Omid, Sean, Hongyuan, Furkan, Yuning, Sikun, Nikunj, Christos, Rasta, Kha-Dinh

Amazon colleagues: Kannan, Han, Alfredo, Eric, Dushyanta, Vel, ...

Roomates: Liang, Yuke, Youfu, Lianke

My family: *I miss them and this dissertation is dedicated to them.*

Learning Representations for Information-rich Graphs

PhD Defense

Zexi Huang

Committee: Ambuj Singh (Chair), Yu-Xiang Wang, Xifeng Yan

Department of Computer Science, University of California, Santa Barbara

April 5, 2023

Reference I

[1] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu.
Asymmetric transitivity preserving graph embedding.
In *SIGKDD*, 2016.

[2] Daixin Wang, Peng Cui, and Wenwu Zhu.
Structural deep network embedding.
In *SIGKDD*, 2016.

[3] Ziwei Zhang, Peng Cui, Xiao Wang, Jian Pei, Xuanrong Yao, and Wenwu Zhu.
Arbitrary-order proximity preserved network embedding.
In *SIGKDD*, 2018.

[4] Aditya Grover and Jure Leskovec.
node2vec: Scalable feature learning for networks.
In *SIGKDD*, 2016.

[5] Giang Hoang Nguyen, John Boaz Lee, Ryan A Rossi, Nesreen K Ahmed, Eunyee Koh, and Sungchul Kim.
Continuous-time dynamic network embeddings.
In *WebConf*, 2018.

Reference II

[6] Amin Javari, Tyler Derr, Pouya Esmailian, Jiliang Tang, and Kevin Chen-Chuan Chang.
Rose: Role-based signed network embedding.
In *WebConf*, 2020.

[7] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi.
Benchmark graphs for testing community detection algorithms.
Physical review E, 78(4):046110, 2008.

[8] Zexi Huang, Arlei Silva, and Ambuj Singh.
A broader picture of random-walk based graph embedding.
In *SIGKDD*, 2021.

[9] Michael T Schaub, Jean-Charles Delvenne, Renaud Lambiotte, and Mauricio Barahona.
Multiscale dynamical embeddings of complex networks.
Physical Review E, 99(6):062308, 2019.

[10] Thomas N Kipf and Max Welling.
Semi-supervised classification with graph convolutional networks.
In *ICLR*, 2017.

Reference III

[11] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio.
Graph attention networks.
arXiv preprint arXiv:1710.10903, 2017.

[12] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka.
How powerful are graph neural networks?
In *ICLR*, 2018.

[13] Thomas N Kipf and Max Welling.
Variational graph auto-encoders.
arXiv preprint arXiv:1611.07308, 2016.

[14] Muhan Zhang and Yixin Chen.
Link prediction based on graph neural networks.
In *NeurIPS*, 2018.

[15] Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec.
Hyperbolic graph convolutional neural networks.
In *NeurIPS*, 2019.

Reference IV

[16] Yiding Zhang, Xiao Wang, Chuan Shi, Nian Liu, and Guojie Song.
Lorentzian graph convolutional networks.
In *WebConf*, 2021.

[17] Lei Cai, Jundong Li, Jie Wang, and Shuiwang Ji.
Line graph neural networks for link prediction.
IEEE TPAMI, 2021.

[18] Zuoyu Yan, Tengfei Ma, Liangcai Gao, Zhi Tang, and Chao Chen.
Link prediction with persistent homology: An interactive view.
In *ICML*, 2021.

[19] Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang.
Neural bellman-ford networks: A general graph neural network framework for link prediction.
In *NeurIPS*, 2021.

[20] Yuzhou Chen, Yulia R Gel, and H Vincent Poor.
Bscnets: Block simplicial complex neural networks.
In *AAAI*, 2022.

Reference V

[21] Liming Pan, Cheng Shi, and Ivan Dokmanić.
Neural link prediction with walk pooling.
In *ICLR*, 2022.

[22] J-C Delvenne, Sophia N Yaliraki, and Mauricio Barahona.
Stability of graph communities across time scales.
PNAS, 107(29):12755–12760, 2010.

[23] Kihyuk Sohn.
Improved deep metric learning with multi-class n-pair loss objective.
NeurIPS, 2016.

[24] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin.
"why should i trust you?" explaining the predictions of any classifier.
In *SIGKDD*, 2016.

[25] Scott M Lundberg and Su-In Lee.
A unified approach to interpreting model predictions.
In *NeurIPS*, 2017.

Reference VI

[26] Sandra Wachter, Brent Mittelstadt, and Chris Russell.
Counterfactual explanations without opening the black box: Automated decisions and the gdpr.
Harvard Journal of Law & Technology, 31(2), 2018.

[27] Berk Ustun, Alexander Spangher, and Yang Liu.
Actionable recourse in linear classification.
In *FAT*, 2019.

[28] Kaivalya Rawal and Himabindu Lakkaraju.
Beyond individualized recourse: Interpretable and interactive summaries of actionable recourses.
In *NeurIPS*, 2020.

[29] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec.
Gnnexplainer: Generating explanations for graph neural networks.
In *NeurIPS*, 2019.

[30] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang.
Parameterized explainer for graph neural network.
In *NeurIPS*, 2020.

Reference VII

[31] Mohit Bajaj, Lingyang Chu, Zi Yu Xue, Jian Pei, Lanjun Wang, Peter Cho-Ho Lam, and Yong Zhang.
Robust counterfactual explanations on graph neural networks.
In *NeurIPS*, 2021.

[32] Ana Lucic, Maartje A Ter Hoeve, Gabriele Tolomei, Maarten De Rijke, and Fabrizio Silvestri.
Cf-gnnexplainer: Counterfactual explanations for graph neural networks.
In *AISTATS*, 2022.

[33] Qiaozhu Mei, Jian Guo, and Dragomir Radev.
Divrank: the interplay of prestige and diversity in information networks.
In *SIGKDD*, 2010.

[34] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann.
Predict then propagate: Graph neural networks meet personalized pagerank.
In *ICLR*, 2018.

[35] Johannes Klicpera, Stefan Weißenberger, and Stephan Günnemann.
Diffusion improves graph learning.
In *NeurIPS*, 2019.

Reference VIII

[36] Ninghao Liu, Xiao Huang, and Xia Hu.
Accelerated local anomaly detection via resolving attributed networks.
In *IJCAI*, 2017.

[37] Kaize Ding, Jundong Li, and Huan Liu.
Interactive anomaly detection on attributed networks.
In *WSDM*, 2019.

[38] Zhiming Xu, Xiao Huang, Yue Zhao, Yushun Dong, and Jundong Li.
Contrastive attributed network anomaly detection with data augmentation.
In *PAKDD*, 2022.

[39] Jianheng Tang, Jiajin Li, Zi-Chao Gao, and Jia Li.
Rethinking graph neural networks for anomaly detection.
In *ICLR*, 2022.

[40] Leonardo Gutiérrez-Gómez, Alexandre Bovet, and Jean-Charles Delvenne.
Multi-scale anomaly detection on attributed networks.
In *AAAI*, 2020.

Reference IX

[41] Xuexiong Luo, Jia Wu, Amin Beheshti, Jian Yang, Xiankun Zhang, Yuan Wang, and Shan Xue.
Comga: Community-aware attributed graph anomaly detection.
In *WSDM*, 2022.