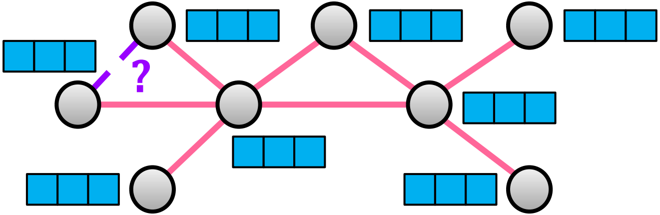
Link Prediction for Information-rich Graphs

Zexi Huang

Department of Computer Science
University of California, Santa Barbara

March 7, 2023

Link prediction

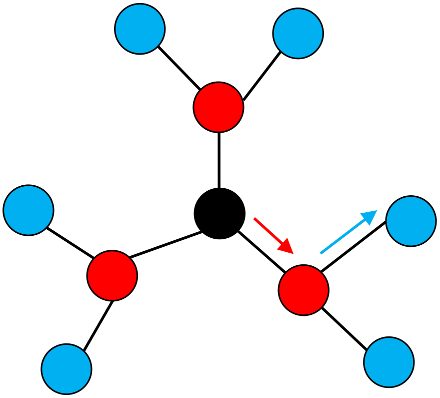


Link prediction: predicting the unobserved interactions (edges) between nodes given the observed graph structure (topology) and other information (e.g., node attributes).

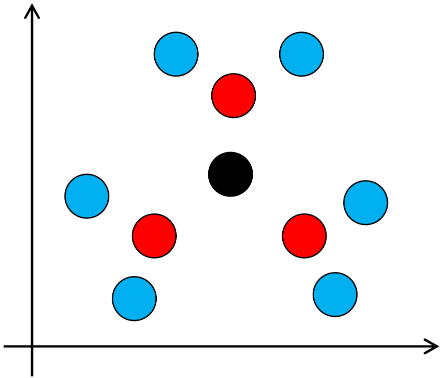
Topological link prediction approaches

- Local topological heuristics
  - Common Neighbors [1], Adamic Adar [2], and Resource Allocation [3]
- Higher-order topological heuristics
  - Katz [4], PageRank [5], and SimRank [6]
- Random-walk based node embedding
  - DeepWalk [7], node2vec [8], and NetMF [9]

Random-walk based node embedding

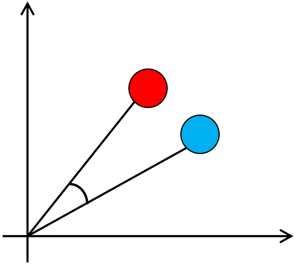


(a) Use random-walks to capture topological proximity (similarity)

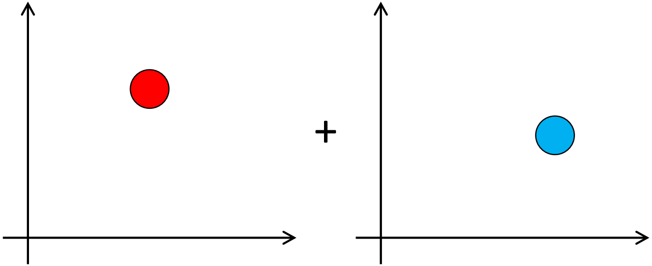


(b) Find low-dimensional latent vectors (embeddings) to preserve similarity

Link prediction based on node embedding

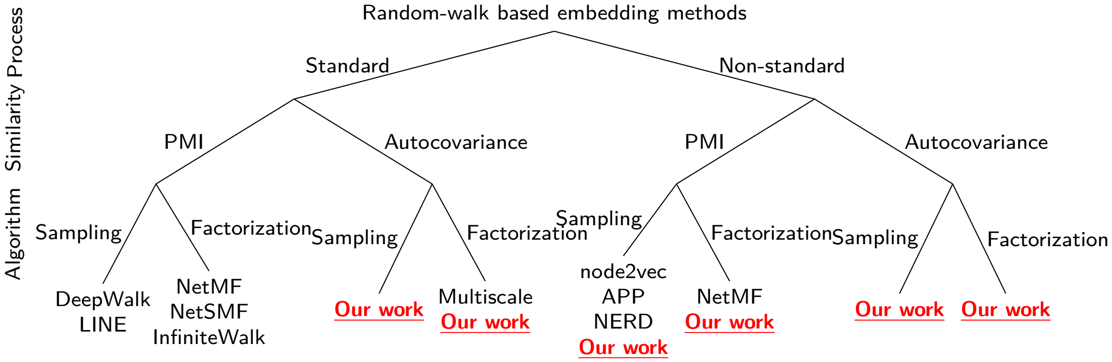


(a) Dot product
[10, 11, 12]



(b) Classification based on combined embeddings
[8, 13, 14]

An unified framework for random-walk based embedding[1]



- We provide key insights on how embedding captures structural scales.
- 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: $R = \log(\Pi M^{\tau}) - \log(\pi\pi^{T})$

Random-walk based similarity metrics

M: <u>transition matrix</u> π, Π: <u>stationary distribution</u> τ: <u>random-walk scale</u>

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

Random-walk based similarity metrics

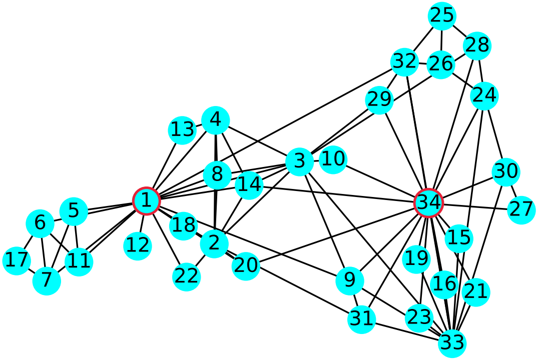
M: transition matrix π, Π: stationary distribution τ: random-walk scale

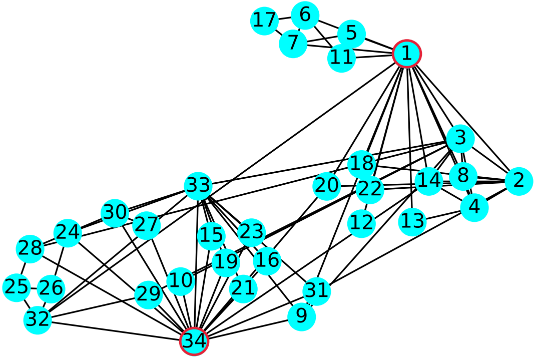
PMI: $R = \log(\Pi M^\tau) - \log(\pi\pi^T)$ Autocovariance: $R = \Pi M^\tau - \pi\pi^T$

Random-walk based similarity metrics

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

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





Link prediction performance

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

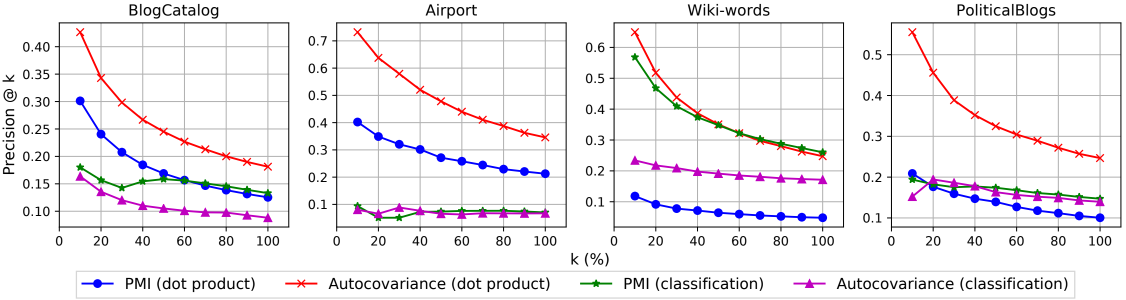


Figure: Autocovariance with dot product ranking consistently outperforms PMI (with either ranking scheme) in link prediction.

Understanding the difference

predicted degree $\propto$ embedding norm $\|\mathbf{u}\|$

Understanding the difference

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

Understanding the difference

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

Autocovariance captures heterogeneous degree distribution in graphs!

Understanding the difference

$$\text{predicted degree} \propto \text{embedding norm } \|\mathbf{u}\| \propto \begin{cases} \textcolor{red}{\text{actual degree}} & \text{for autocov.} \\ \textcolor{blue}{\text{constant}} & \text{for PMI} \end{cases}$$

Autocovariance captures heterogeneous degree distribution in graphs!

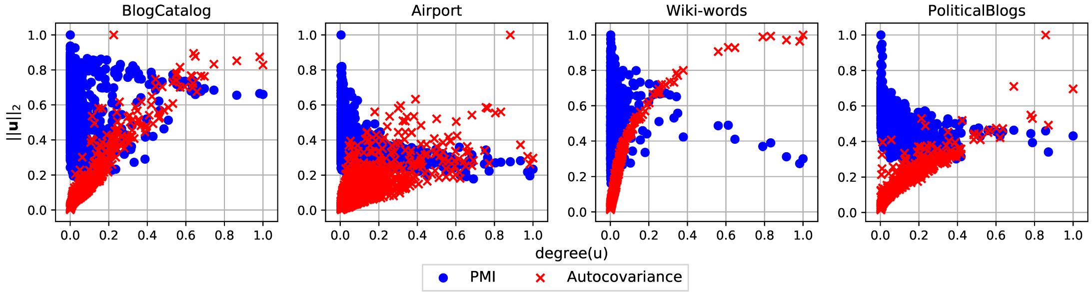


Figure: Autocovariance embedding norms correlate with actual degrees, but not PMI.

Understanding the difference

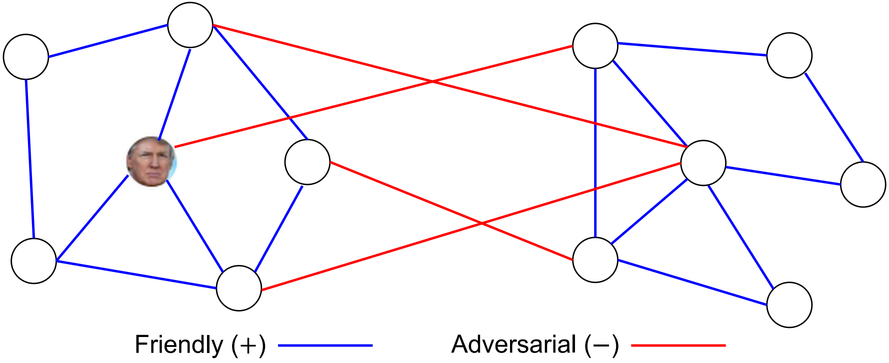
$$\text{predicted degree} \propto \text{embedding norm } \|\mathbf{u}\| \propto \begin{cases} \text{actual degree} & \text{for autocov.} \\ \text{constant} & \text{for PMI} \end{cases}$$

(a) PMI

(b) Autocovariance

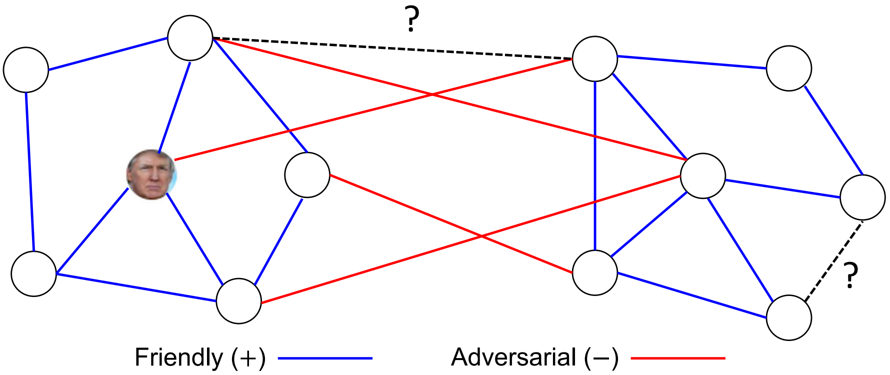
Figure: Autocovariance predicts more edges connecting to the hubs than PMI.

Polarized signed embedding for effective signed link prediction[2]



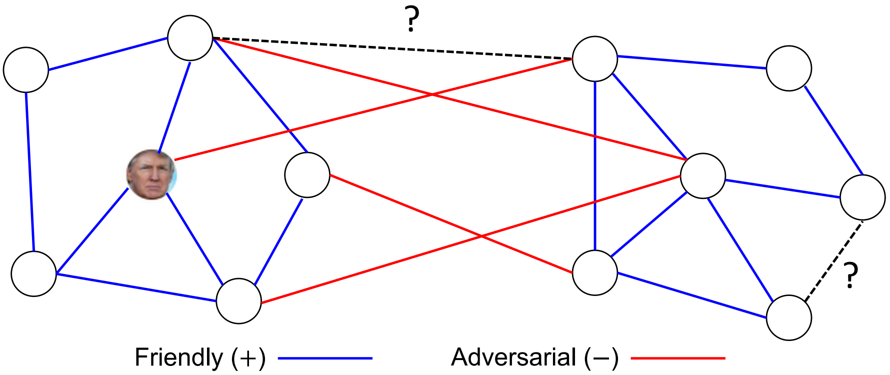
[2] **Huang**, Silva, Singh. POLE: Polarized embedding for signed networks. WSDM'22.

Polarized signed embedding for effective signed link prediction[2]



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

Polarized signed 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

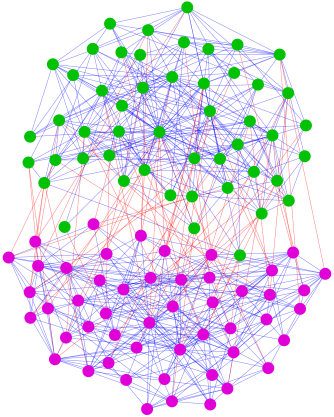
- Predicting signs of links:
  - Signed embedding [15, 16] (signed similarity)

Signed link prediction in polarized networks

- Predicting signs of links:
  - Signed embedding [15, 16] (signed similarity)
- What about predicting link existence?
  - Unsigned embedding [7, 8] (connectivity)

Signed link prediction in polarized networks

- Predicting signs of links:
  - Signed embedding [15, 16] (signed similarity)
- What about predicting link existence?
  - Unsigned embedding [7, 8] (connectivity)
- However, they **cannot** predict negative links between polarized communities!
  - Because topology and link signs are interdependent
  - Need to capture signed/unsigned similarities jointly



Intra-community: dense, positive
Inter-community: sparse, negative

Signed random walk

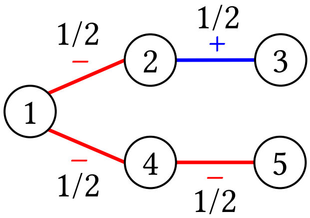
- Unsigned RW: $|M|_{uv}(t) = \sum_{\text{all length-}t\text{ paths }l\text{ between }u\text{ and }v} \text{Prob}(l)$
  - $\text{Prob}(l)$ captures the unsigned similarity.

Signed random walk

- Unsigned RW: $|M|_{uv}(t) = \sum_{\text{all length-}t\text{ paths } l \text{ between } u \text{ and } v} \text{Prob}(l)$
  - $\text{Prob}(l)$ captures the unsigned similarity.
- Signed RW: $M_{uv}(t) = \sum_{\text{all length-}t\text{ paths } l \text{ between } u \text{ and } v} \text{Prob}(l)\text{Sign}(l)$
  - $\text{Sign}(l)$ based on the social balance theory captures the signed similarity.

Signed random walk

- Unsigned RW: $|M|_{uv}(t) = \sum_{\text{all length-}t\text{ paths } l \text{ between } u \text{ and } v} \text{Prob}(l)$
  - $\text{Prob}(l)$ captures the unsigned similarity.
- Signed RW: $M_{uv}(t) = \sum_{\text{all length-}t\text{ paths } l \text{ between } u \text{ and } v} \text{Prob}(l)\text{Sign}(l)$
  - $\text{Sign}(l)$ 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:

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

- POLE: extends autocovariance similarity [17, 18] to signed RW

$$R(t) = M(t)^T W M(t)$$

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: extends autocovariance similarity [17, 18] to signed RW

$$R(t) = M(t)^T W M(t)$$

- Desired properties:
  - Positive links: large positive similarity;
  - Negative links: large negative similarity;
  - Non-links: small similarity.

Comparison of similarity distributions

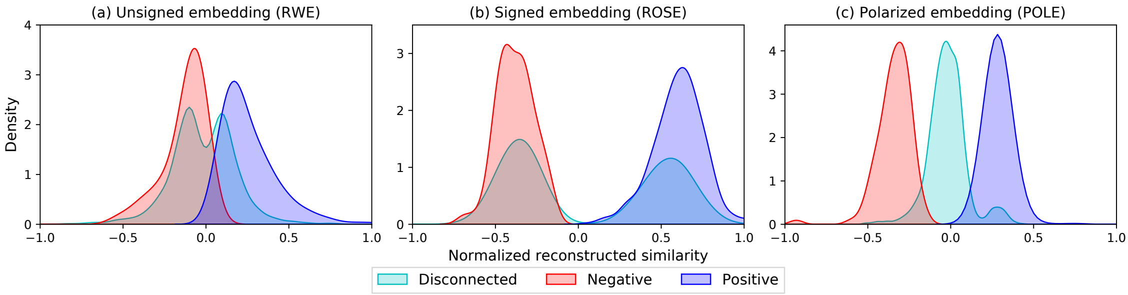


Figure: Distributions of the reconstructed similarity for different types of node pairs in a polarized graph using (a) unsigned [18], (b) signed [14], and (c) polarized embedding.

Comparison of similarity distributions

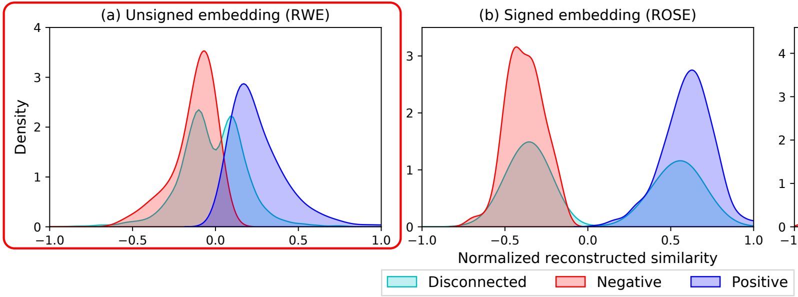


Figure: Distributions of the reconstructed similarity for different types of node pairs in a polarized graph using (a) unsigned [18], (b) signed [14], and (c) polarized embedding.

Comparison of similarity distributions

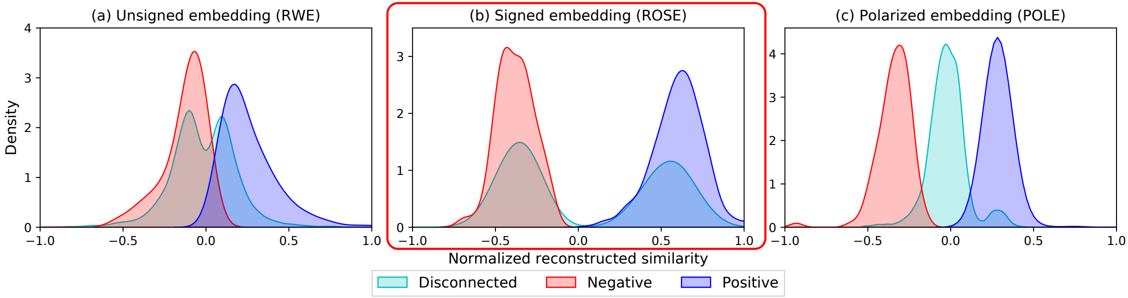


Figure: Distributions of the reconstructed similarity for different types of node pairs in a polarized graph using (a) unsigned [18], (b) signed [14], and (c) polarized embedding.

Comparison of similarity distributions

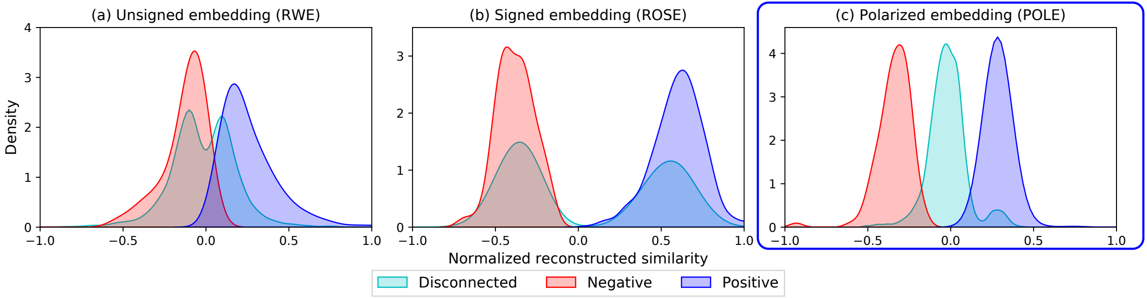


Figure: Distributions of the reconstructed similarity for different types of node pairs in a polarized graph using (a) unsigned [18], (b) signed [14], and (c) polarized embedding.

Signed link prediction results

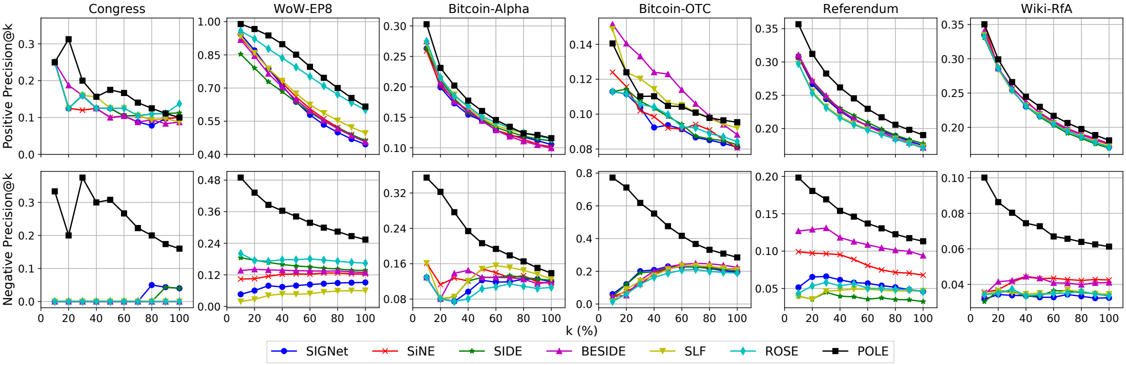
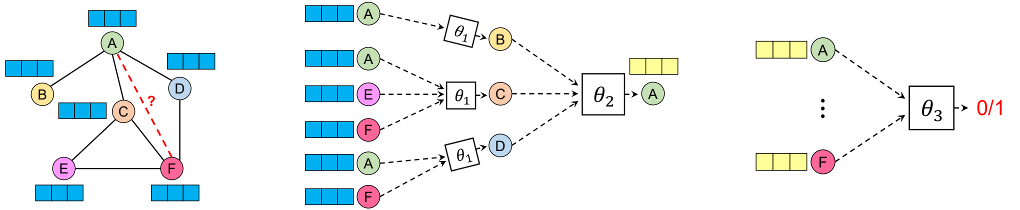


Figure: Signed link prediction with link existence information performance comparison. POLE outperforms all baselines in almost all datasets, especially for the negative links.

Attributed graph embedding and graph neural networks

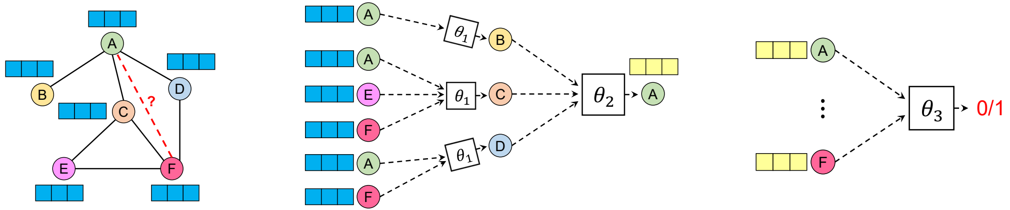
- GNNs [19, 20, 21] are a powerful DL paradigm that learns to generate better node features (embeddings) using structure information.



(a) Input attributed graph (b) GNN message-passing (c) Pooling and classification

Attributed graph embedding and graph neural networks

- GNNs [19, 20, 21] are a powerful DL paradigm that learns to generate better node features (embeddings) using structure information.

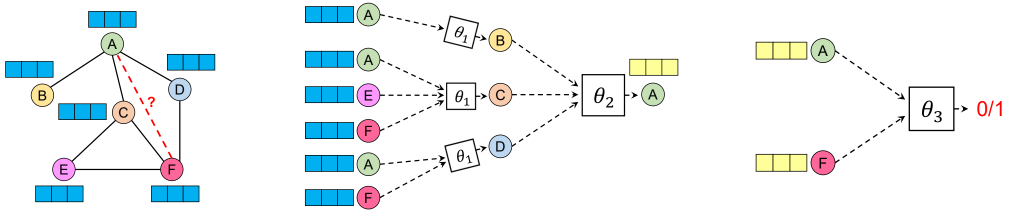


(a) Input attributed graph (b) GNN message-passing (c) Pooling and classification

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

Graph autoencoder [22]

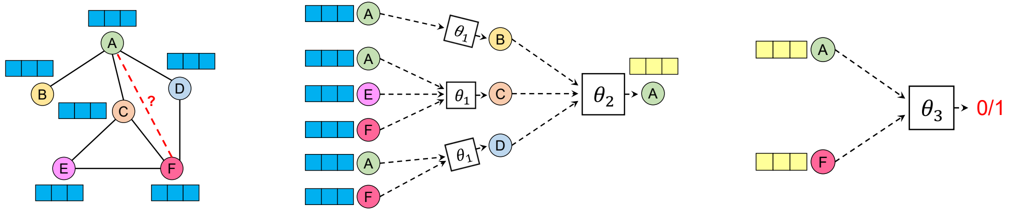
- Train graph convolutional networks (GCNs) to generate node embeddings and use the pairwise node embeddings to predict edges.



(a) Input attributed graph (b) GCN message-passing (c) Pairwise classification

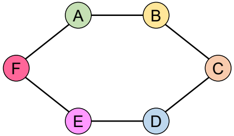
Graph autoencoder [22]

- Train graph convolutional networks (GCNs) to generate node embeddings and use the pairwise node embeddings to predict edges.



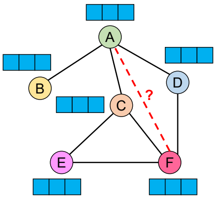
(a) Input attributed graph (b) GCN message-passing (c) Pairwise classification

- Not enough expressive power to capture topological proximity.

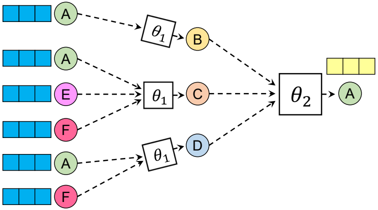


Graph autoencoder [22]

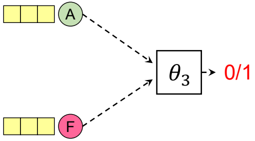
- Train graph convolutional networks (GCNs) to generate node embeddings and use the pairwise node embeddings to predict edges.



(a) Input attributed graph

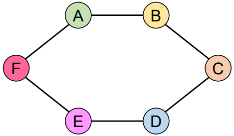


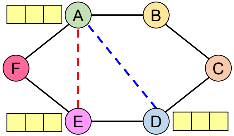
(b) GCN message-passing



(c) Pairwise classification

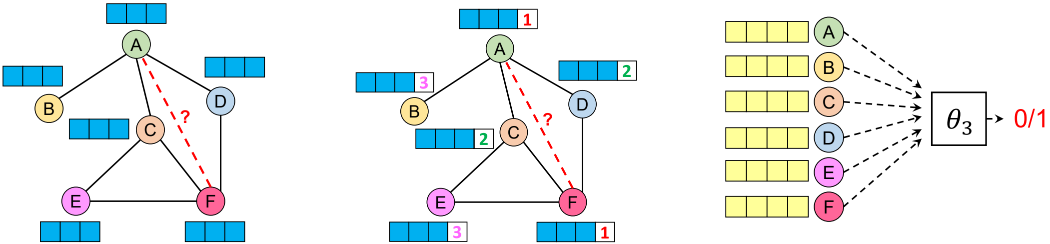
- Not enough expressive power to capture topological proximity.





SEAL: learning from subgraphs [23]

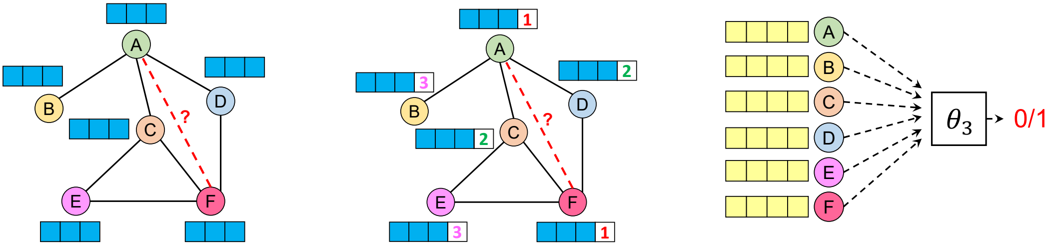
- Address the problem of expressive power of GAE by
  - explicitly adding topological features via labeling trick
  - modeling link prediction as a subgraph classification problem



(a) Input attributed graph (b) Labeling trick (c) Subgraph classification

SEAL: learning from subgraphs [23]

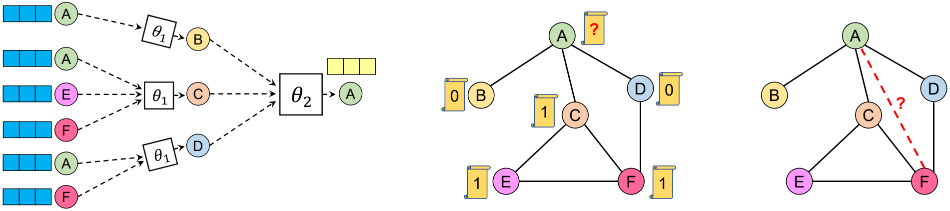
- Address the problem of expressive power of GAE by
  - explicitly adding topological features via labeling trick
  - modeling link prediction as a subgraph classification problem



(a) Input attributed graph (b) Labeling trick (c) Subgraph classification

- Follow-up work investigates other topological features [29, 31] and different pooling strategies [24, 25].

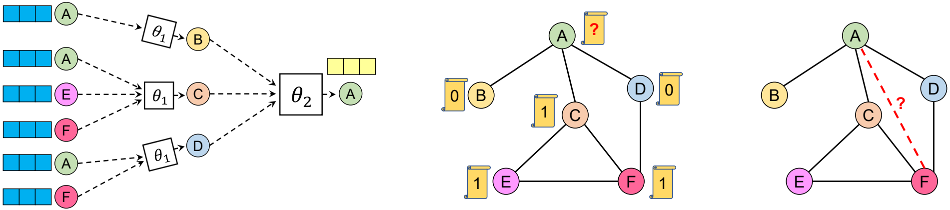
Challenges of GNNs for link prediction



(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?

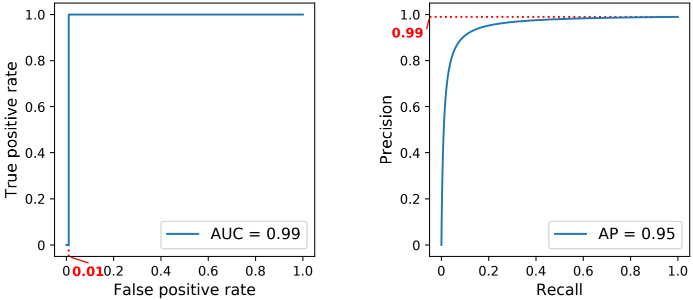
Supervised link prediction evaluation

- Existing work [22, 23, 26, 27, 28, 29, 30, 31, 25] adopts AUC and AP with *biased testing* (downsampling negative/disconnected pairs).

Supervised link prediction evaluation

- Existing work [22, 23, 26, 27, 28, 29, 30, 31, 25] adopts AUC and AP with *biased testing* (downsampling negative/disconnected pairs).

Example: Consider a graph with 10k nodes, 100k edges, and 99.9M disconnected (or negative) pairs. A bad link prediction model that predicts **1M** false positives (**1k** with *biased testing*) higher than the **100k** true edges achieves **0.99** AUC and **0.95** AP.

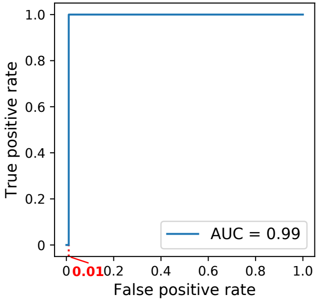


(a) ROC (b) PR curve (biased) curve

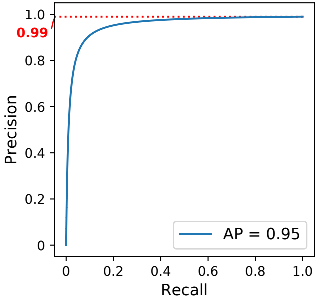
Supervised link prediction evaluation

- Existing work [22, 23, 26, 27, 28, 29, 30, 31, 25] adopts AUC and AP with *biased testing* (downsampling negative/disconnected pairs).

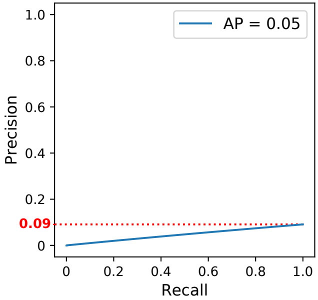
Example: Consider a graph with 10k nodes, 100k edges, and 99.9M disconnected (or negative) pairs. A bad link prediction model that predicts **1M** false positives (**1k** with *biased testing*) higher than the **100k** true edges achieves **0.99** AUC and **0.95** AP.



(a) ROC



(b) PR curve (biased) curve



(c) PR curve (unbiased)

Supervised link prediction evaluation

- Existing work [22, 23, 26, 27, 28, 29, 30, 31, 25] 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 [10, 12, 18] and IR.

Supervised link prediction evaluation

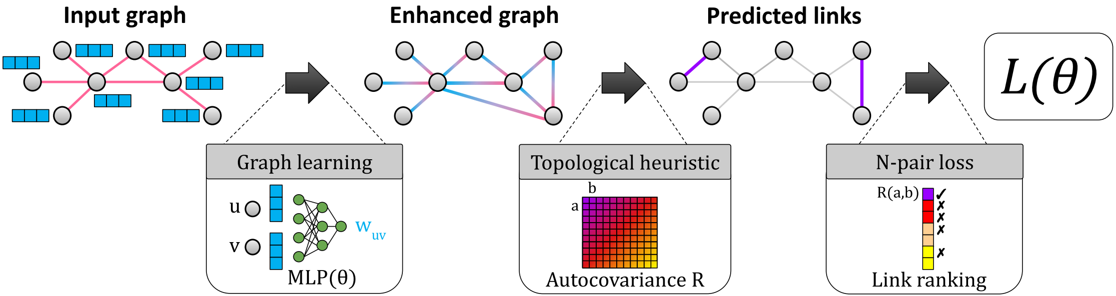
- Existing work [22, 23, 26, 27, 28, 29, 30, 31, 25] adopts AUC and AP with *biased testing* (downsampling negative/disconnected pairs), which pictures an **overly optimistic** view of model performance.
- We argue for evaluation under *unbiased testing*, which has been widely applied in unsupervised link prediction [10, 12, 18] and IR.

Supervised link prediction training

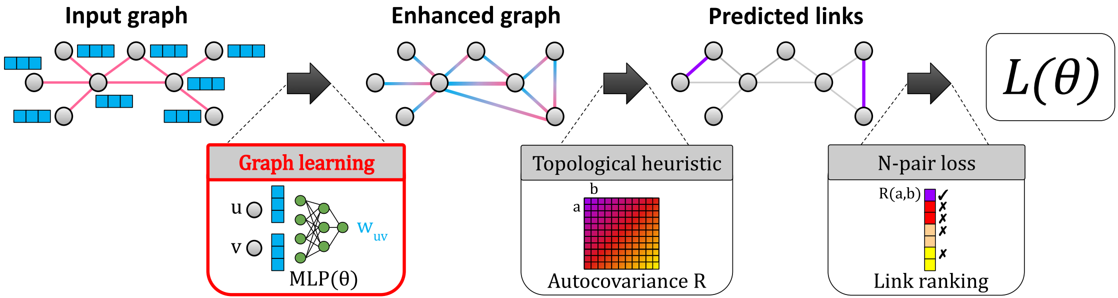
- Existing work uses binary cross entropy loss with *biased training*.
  - It **discards** potentially useful evidence from negative pairs.
  - It induces the model to **overestimate** the probability of positive pairs.

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

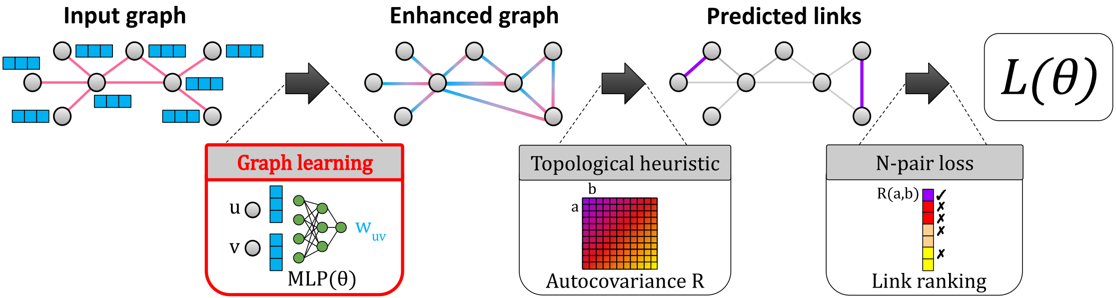
- Explore alternative frameworks to combine topology and attributes
- Address the problem of class imbalance in training and testing



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

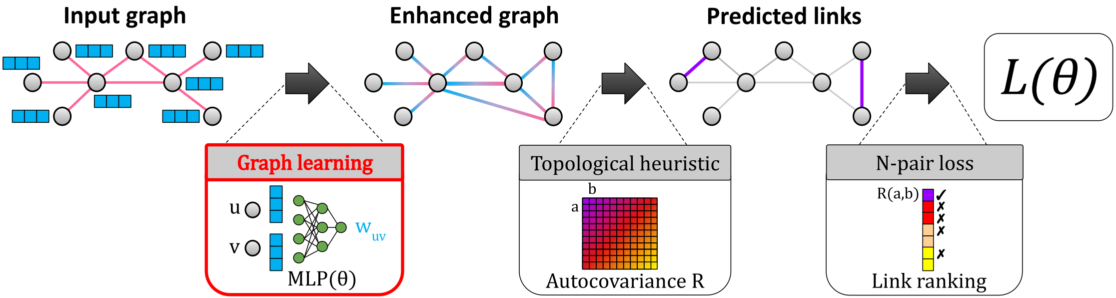


Graph learning



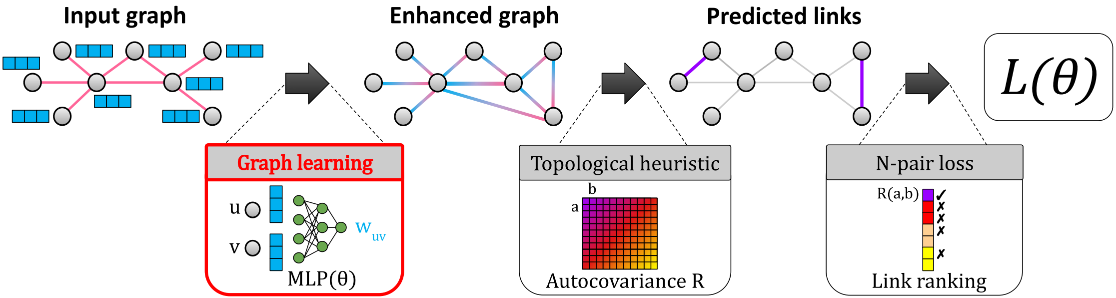
Graph learning

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



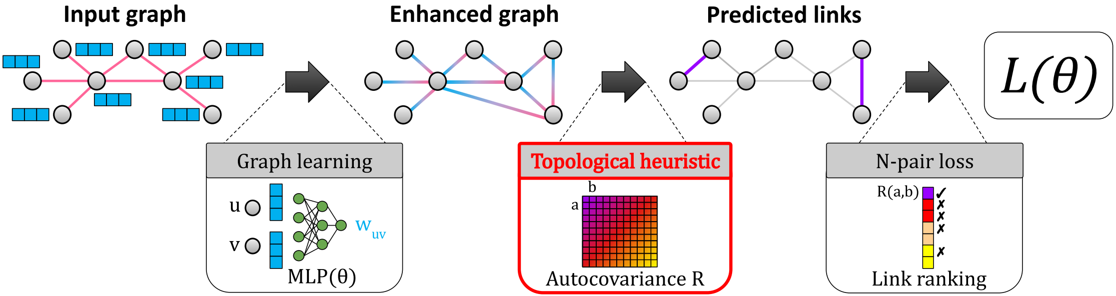
Graph learning

- 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

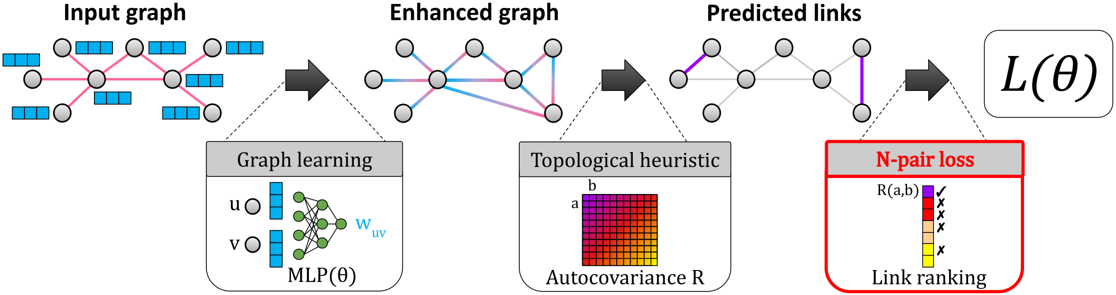
- Graph augmentation: $\widetilde{E} = E + \{(u,v) \mid s(x_u, x_v) > s_\eta\}$
- Trained weighting: $w_{uv} = \text{MLP}([x_u + x_v; |x_u - x_v|]; \theta)$
- Combined weights: $\widetilde{A}_{uv} = \alpha A_{uv} + (1-\alpha)(\beta w_{uv} + (1-\beta) s(x_u, x_v))$



Topological heuristic

- Applying Autocovariance [32, 18] to the enhanced graph $\widetilde{A}$:

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



N-pair loss [33]

- 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		$\mathbf{3.90 \pm 0.03}$	$\mathbf{4.55 \pm 0.02}$	$\mathbf{2.88 \pm 0.09}$	$\mathbf{25.68 \pm 0.53}$	$\mathbf{18.77 \pm 0.19}$

Link prediction results (cont.)

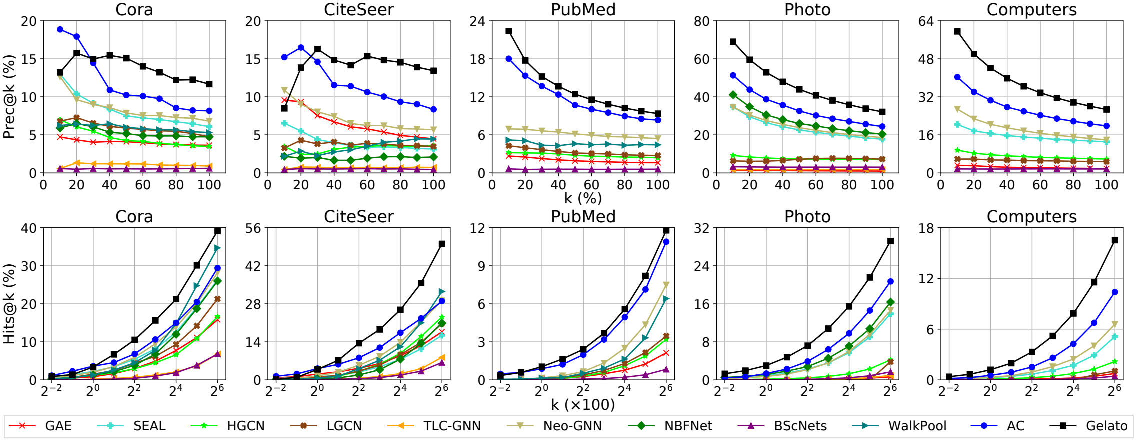
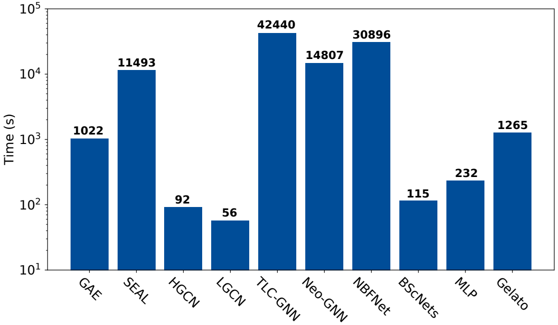
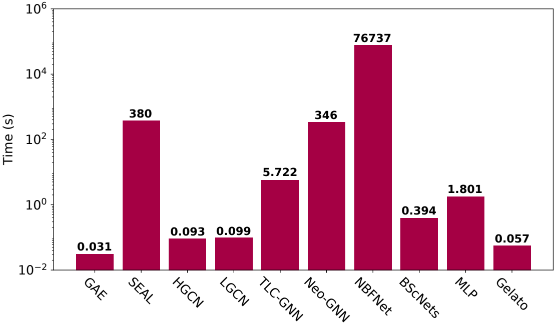


Figure: Link prediction performance in terms of precision@k and hits@k for varying values of k. With few exceptions, Gelato outperforms the baselines across different k.

Running time comparison



(a) Training time until convergence



(b) Inference time per *unbiased testing*

Figure: Even under *unbiased training*, Gelato has competitive training time (**11**×compared to the best GNN-based method, Neo-GNN) and is significantly faster than most baselines for inference (**6,000**×).

Summary

- We investigated the problem of representation learning for effective link prediction in simple, signed, and attributed networks.
- We scrutinized popular representation learning paradigms (node embedding and GNNs) and discussed novel solutions leveraging random-walk dynamics, social theories, and graph learning.

Reference I

[1] Mark EJ Newman.
Clustering and preferential attachment in growing networks.
Physical review E, 64(2):025102, 2001.

[2] Lada A Adamic and Eytan Adar.
Friends and neighbors on the web.
Social networks, 25(3):211–230, 2003.

[3] Tao Zhou, Linyuan Lü, and Yi-Cheng Zhang.
Predicting missing links via local information.
The European Physical Journal B, 71(4):623–630, 2009.

[4] Leo Katz.
A new status index derived from sociometric analysis.
Psychometrika, 18(1):39–43, 1953.

[5] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd.
The pagerank citation ranking: Bringing order to the web.
Technical report, Stanford InfoLab, 1999.

Reference II

[6] Glen Jeh and Jennifer Widom.
Simrank: a measure of structural-context similarity.
In *SIGKDD*, 2002.

[7] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena.
Deepwalk: Online learning of social representations.
In *SIGKDD*, 2014.

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

[9] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang.
Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec.
In *WSDM*, 2018.

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

Reference III

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

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

[13] Giang Hoang Nguyen, John Boaz Lee, Ryan A Rossi, Nesreen K Ahmed, Eunyee Koh, and Sungchul Kim.
Continuous-time dynamic network embeddings.
In *Companion Proceedings of the The Web Conference 2018*, pages 969–976, 2018.

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

[15] Junghwan Kim, Haekyu Park, Ji-Eun Lee, and U Kang.
Side: representation learning in signed directed networks.
In *WebConf*, 2018.

Reference IV

[16] Yiqi Chen, Tieyun Qian, Huan Liu, and Ke Sun.
"bridge" enhanced signed directed network embedding.
In *CIKM*, 2018.

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

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

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

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

Reference V

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

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

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

[24] Zheyi Liu, Darong Lai, Chuanyou Li, and Meng Wang.
Feature fusion based subgraph classification for link prediction.
In *CIKM*, 2020.

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

Reference VI

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

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

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

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

[30] 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.

Reference VII

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

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

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