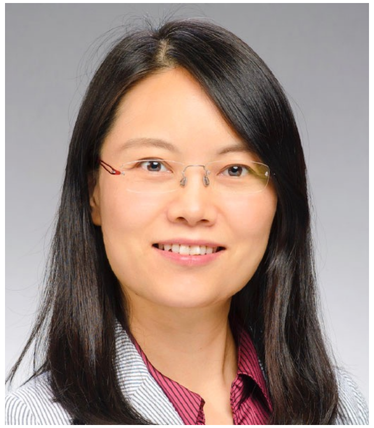
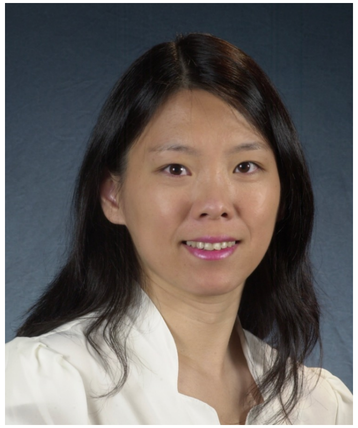
AAAI-24 Tutorial
Feb 2024, Vancouver

Knowledge-enhanced Graph Learning



Yijun Tian	Shichao Pei	Xiangliang Zhang	Wei Wang	Hanghang Tong	Nitesh Chawla
ND	UMB	ND	UCLA	UIUC	ND

UNIVERSITY OF NOTRE DAME
UMASS BOSTON
UCLA
UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Tutorial Website:

yijuntian.com/tutorial

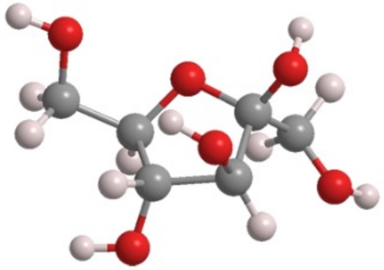
Yijun is looking for job opportunities

Tutorial Outline

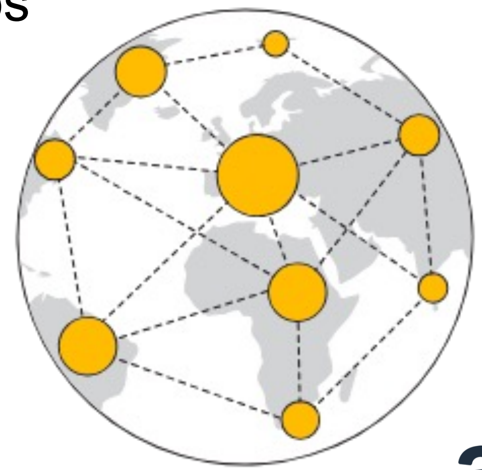
- Preliminaries and Foundations
- Graph Learning Enhanced by Knowledge from Data
- Graph Learning Enhanced by Knowledge from Models
- Graph Learning Enhanced by Knowledge from Humans and Domains
- Graph Learning Enhanced by Knowledge from External Sources
- Knowledge-enhanced Graph Learning for Real-world Applications
- Summary and Future Directions

Graph-structured data are ubiquitous

Social Media
Nodes are people
Edges are friendships



Chemistry
Nodes are atoms
Edges are chemical bonds



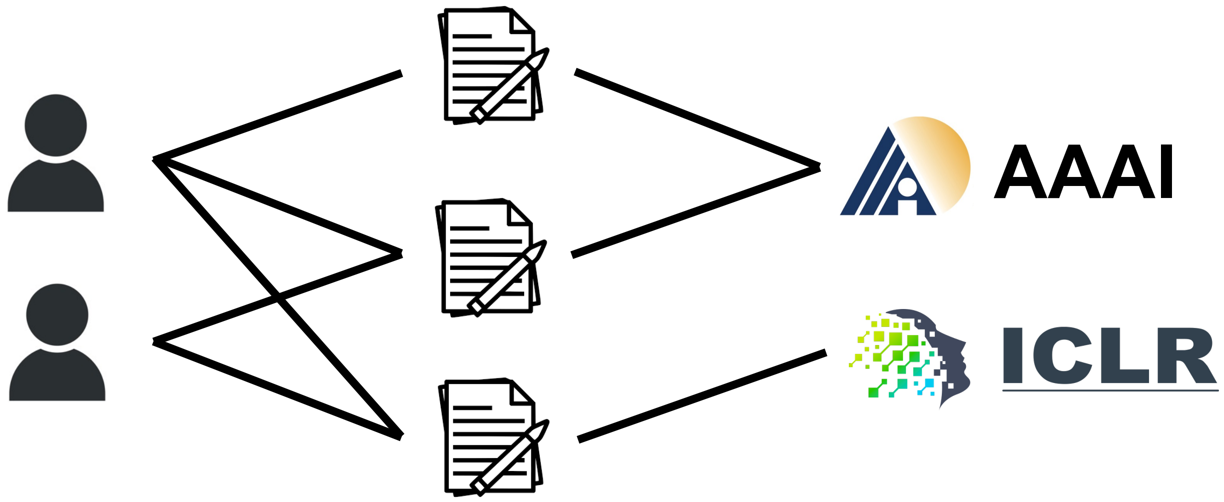
Finance
Nodes are accounts
Edges are transactions

E-commerce
Nodes are users/products
Edges are purchases

Graph-structured data are ubiquitous

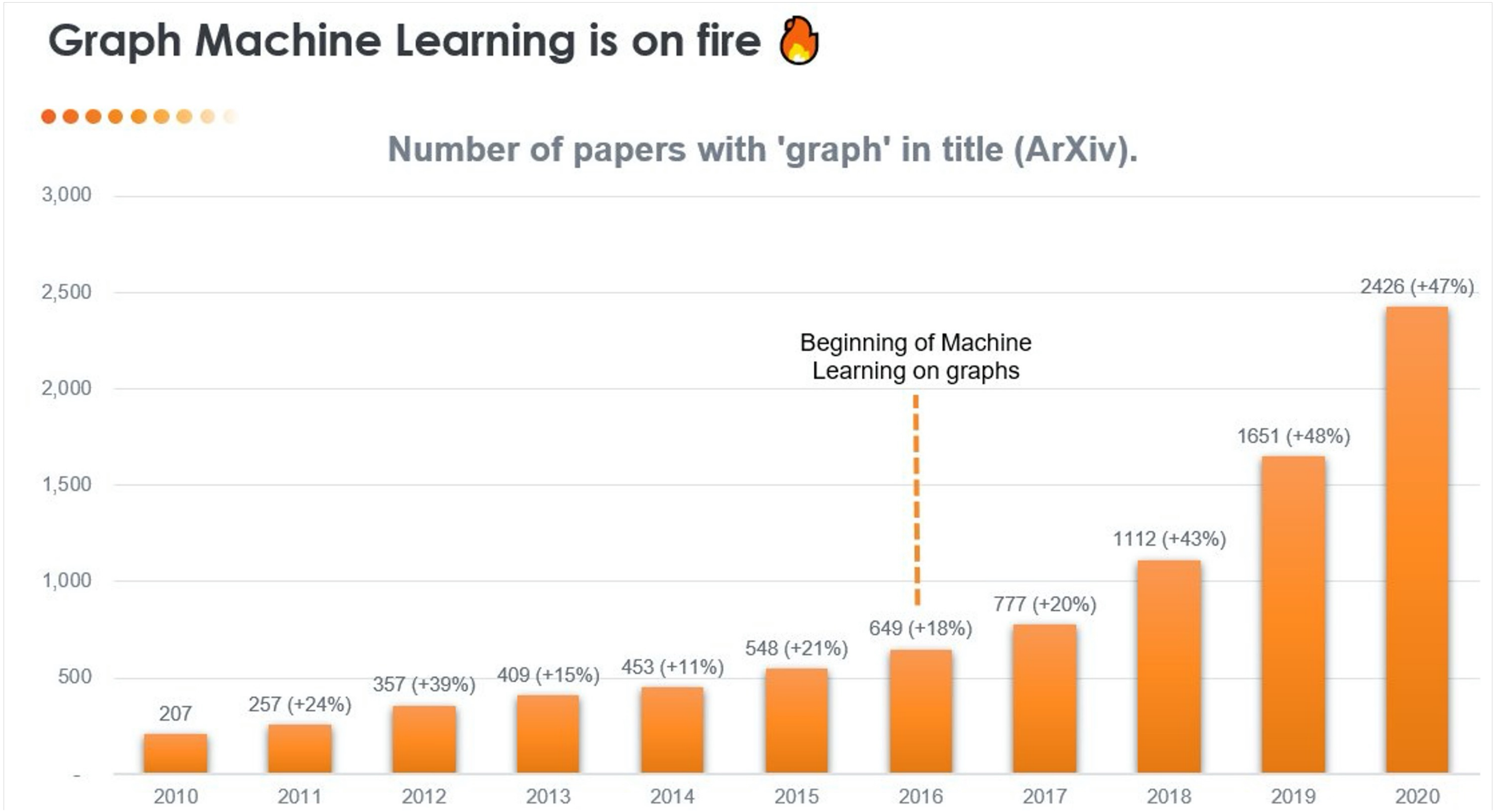
A real-world graph example:

Nodes: Authors Papers Venues



Edges: write publish at

Graph Machine Learning: Recent Trending

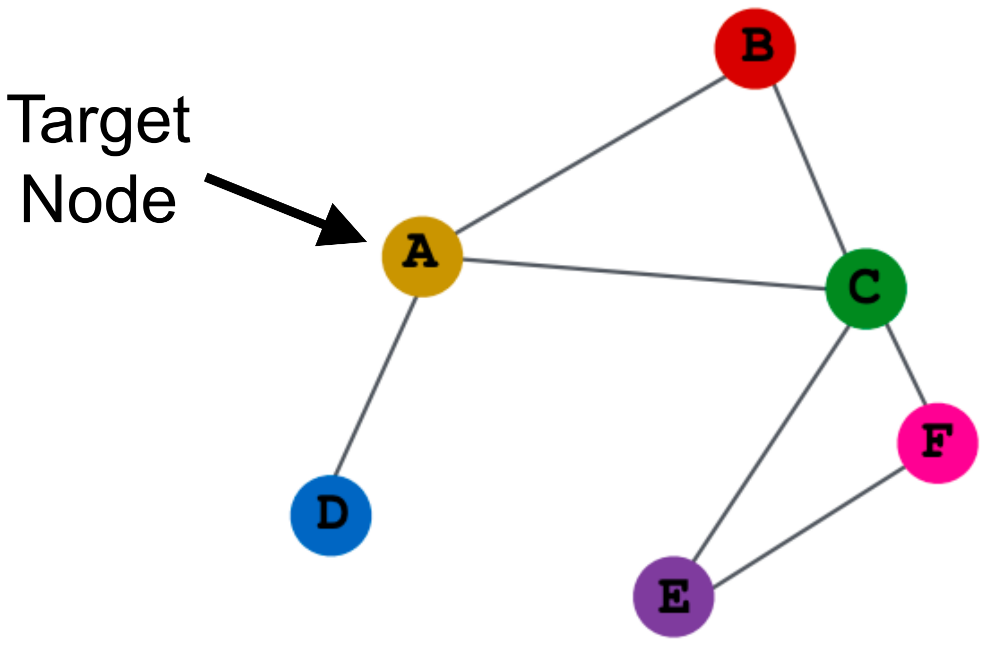


Graph Machine Learning: Previous Tutorials

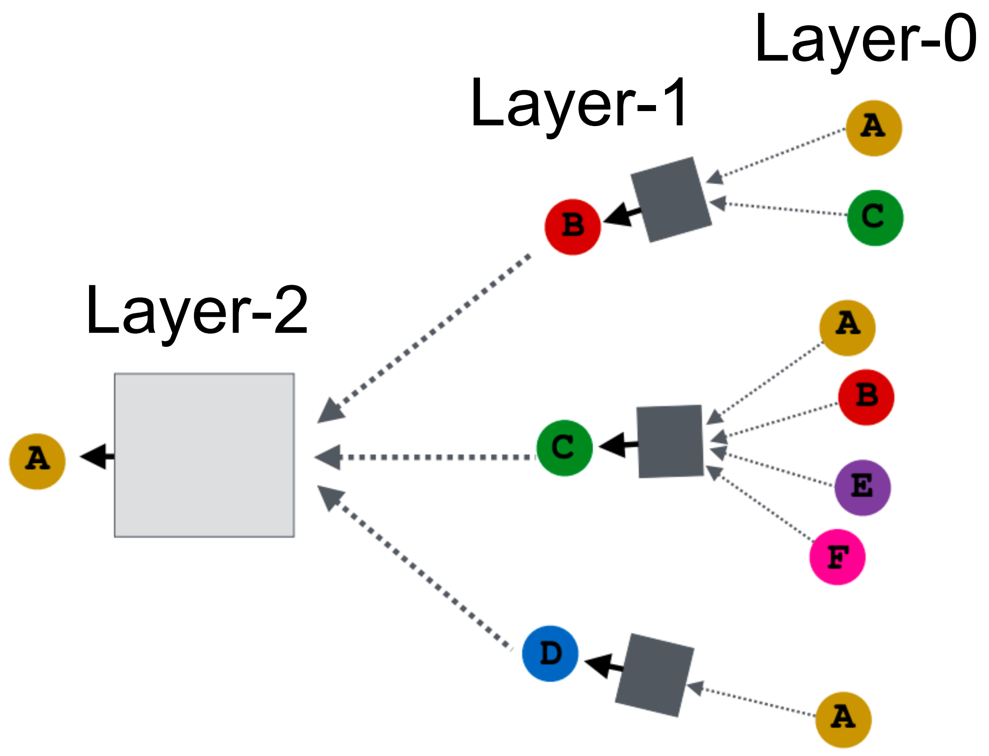
- Graph Neural Networks: Foundations, Frontiers, Applications
  - IJCAI 2022, KDD 2022, AAAI 2023, KDD 2023, WWW 2023
- Graph Neural Networks: Models and Applications
  - AAAI 2020, AAAI 2021
- Large-Scale Graph Neural Networks: The Past and New Frontiers
  - KDD 2023
- Self-supervised Learning and Pre-training on Graphs
  - WWW 2023

Graph Neural Networks: Foundations

Key idea: aggregates information from node neighborhood



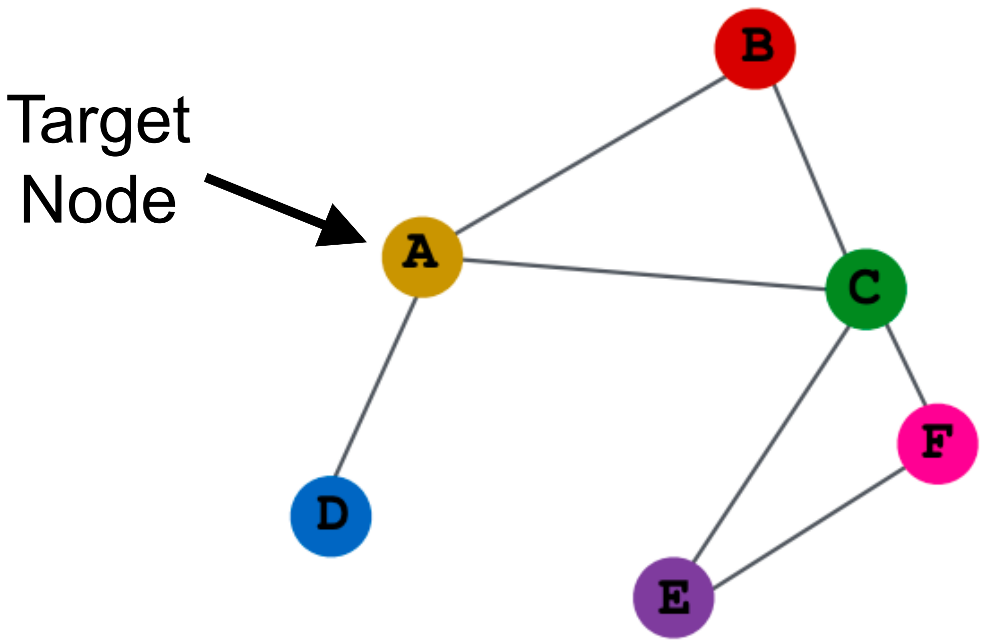
Input Graph



The Computation Process for Node A

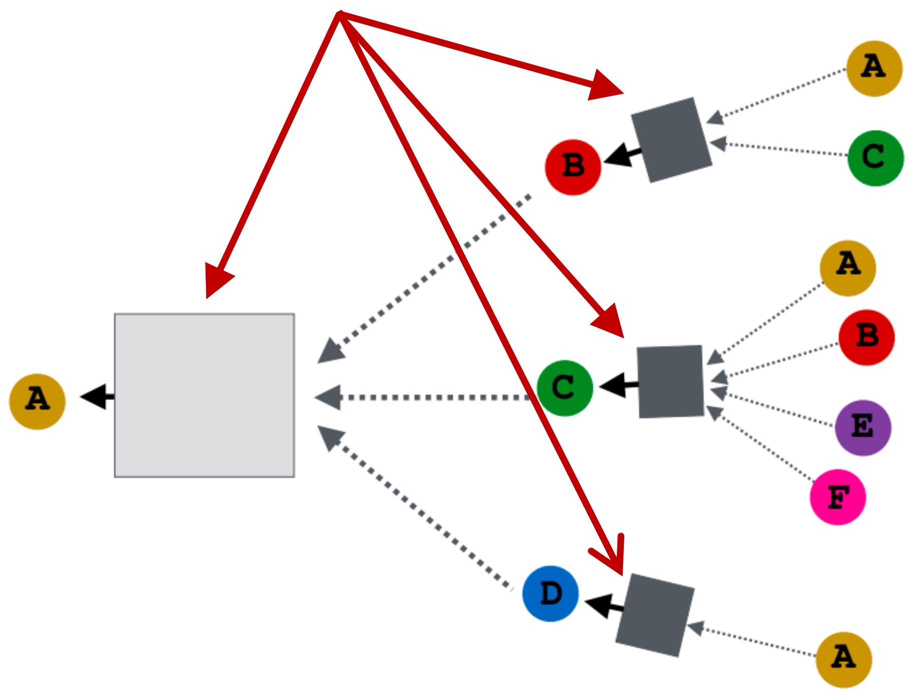
Graph Neural Networks: Foundations

Key idea: aggregates information from node neighborhood



Input Graph

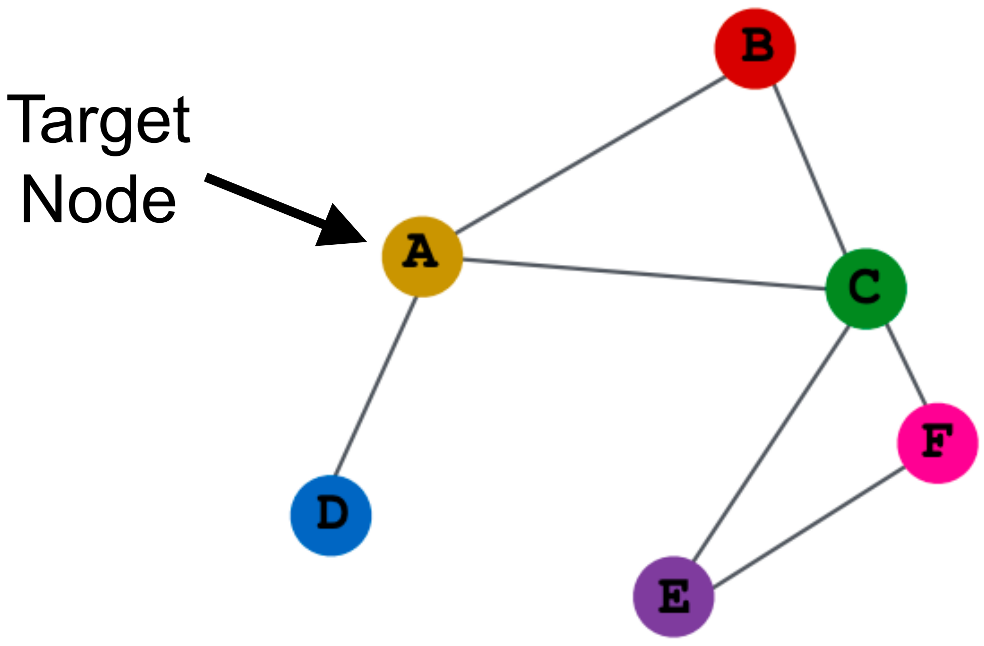
Neural networks that aggregate information



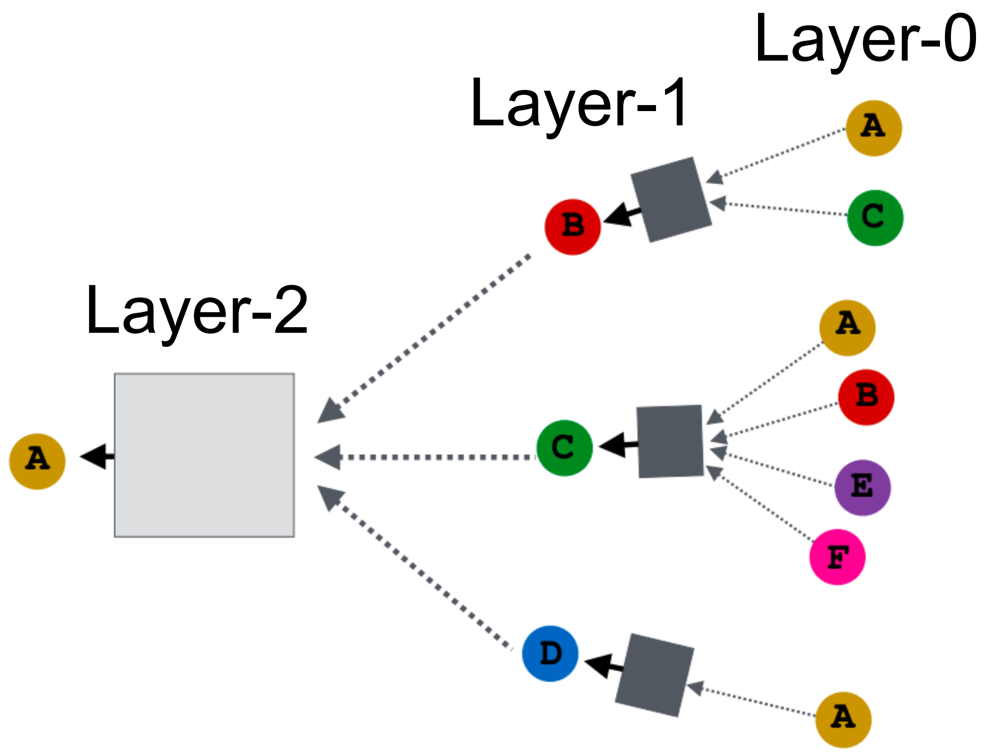
The Computation Process for Node A

Graph Neural Networks: Foundations

- Nodes have embeddings at each layer
- "Layer-0" embedding is the input feature of nodes



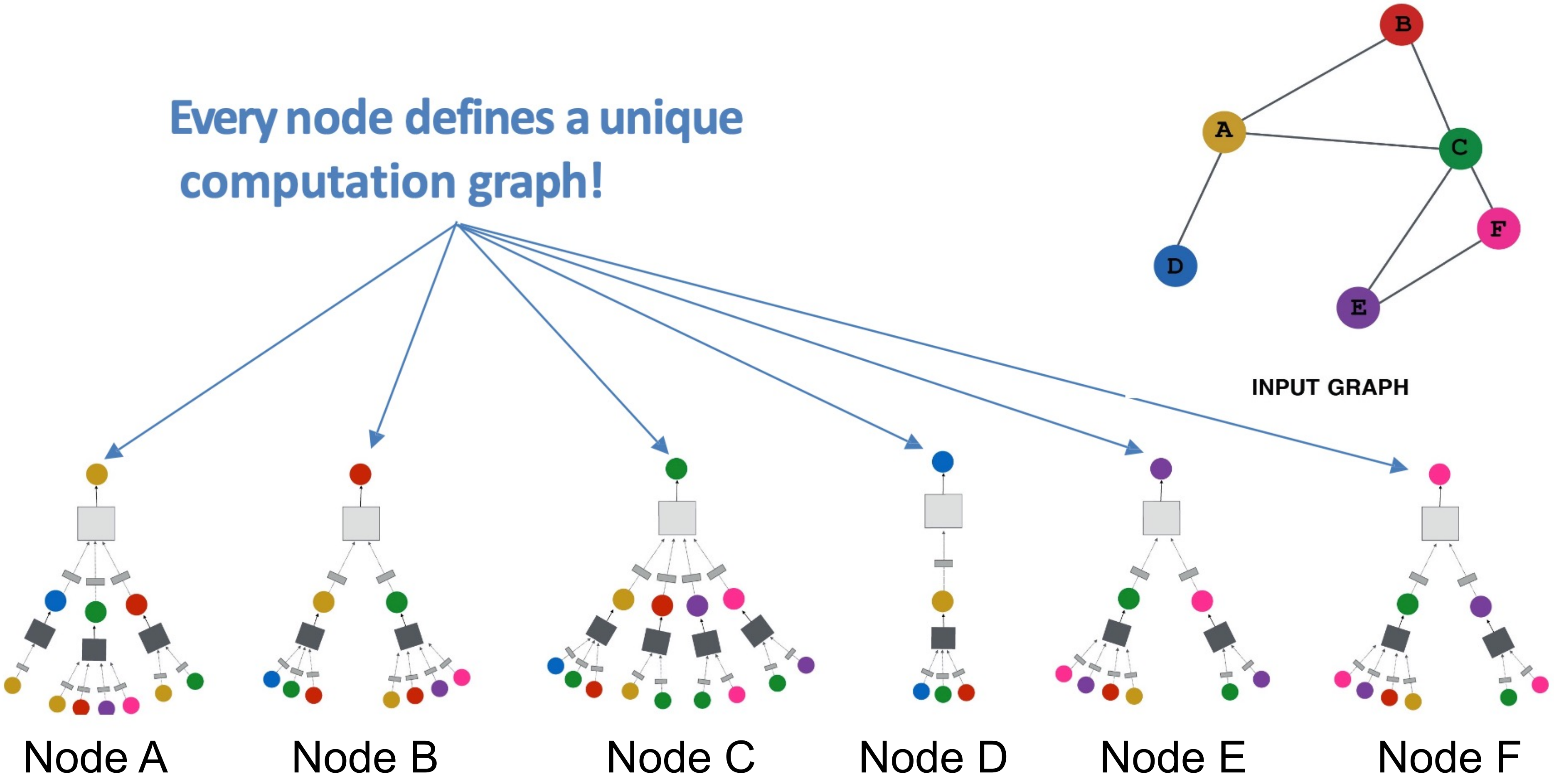
Input Graph



The Computation Process for Node A

Graph Neural Networks: Foundations

The computation graph is defined by neighborhood



Graph Neural Networks: Foundations

Basic approach: Average neighbor information and apply a neural network

Initial “layer 0” embeddings are equal to node features

$$\boxed{\mathbf{h}_v^0 = \mathbf{x}_v}$$

previous layer embedding of v

$$\boxed{\mathbf{h}_v^k} = \boxed{\sigma}\left(\mathbf{W}_k \boxed{\sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|}} + \mathbf{B}_k \boxed{\mathbf{h}_v^{k-1}}\right), \quad \forall k > 0$$

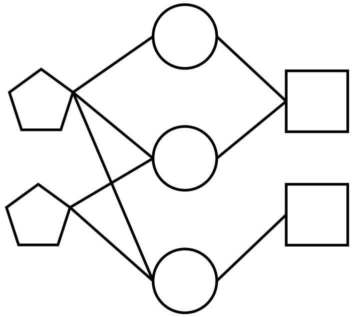
kth layer embedding of v

non-linearity (e.g., ReLU or tanh)

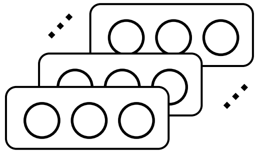
average of neighbor’s previous layer embeddings

Graph Neural Networks: Foundations

Existing GNNs primarily focus on leveraging graph structures and node features



Graph Structure: A



Node Features: X

$$GNN = f(\text{Graph Structure, Node Features})$$

Model	Conference	Citation
GCN	ICLR 2017	29000+
GraphSAGE	NeurIPS 2017	12000+
GAT	ICLR 2018	8000+
GIN	ICLR 2019	6000+

Graph Neural Networks: New Frontiers

Existing GNNs primarily focus on leveraging graph structures and node features

↓

Enhancing graph learning with auxiliary knowledge

Graph Neural Networks: New Frontiers

Existing GNNs primarily focus on leveraging graph structures and node features

Enhancing graph learning with auxiliary knowledge

Important and useful information that can be obtained, extracted, or learned from resources beyond the provided graph structures and node features

Graph Neural Networks: New Frontiers

Existing GNNs primarily focus on leveraging graph structures and node features

⇩

Enhancing graph learning with auxiliary knowledge

⇩

Knowledge-enhanced Graph Learning

Graph Neural Networks: New Frontiers

Existing GNNs primarily focus on leveraging graph structures and node features

↓

Enhancing graph learning with auxiliary knowledge

↓

Knowledge-enhanced Graph Learning

Reduce reliance on massive data and intricate model

Performance ↑ Efficiency ↑

Trustworthiness ↑

Knowledge-enhanced Graph Learning

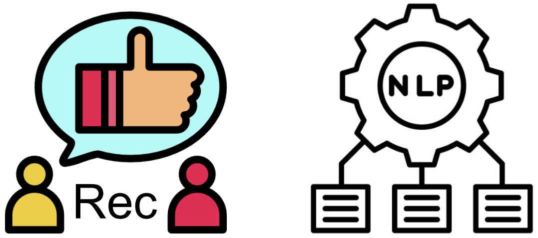
Agenda

Implicit Knowledge

Knowledge from **Data**	Knowledge from **Models**
Data → Knowledge → Model	Teacher Model → Knowledge → Student Model
Data Knowledge	Learned Knowledge

Explicit Knowledge

Knowledge from **Humans and Domains**	Knowledge from **External Sources**
Human/ Domain → Knowledge → Model	External Sources → Knowledge → Model
Domain Knowledge	External Knowledge



Real-world Applications

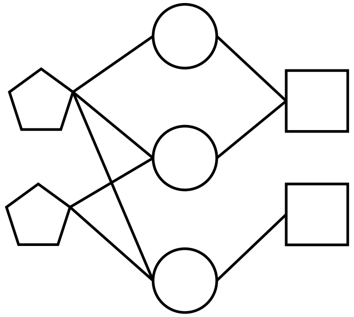
Future Directions

Tutorial Outline

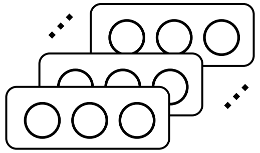
- Preliminaries and Foundations
- Graph Learning Enhanced by Knowledge from Data
- Graph Learning Enhanced by Knowledge from Models
- Graph Learning Enhanced by Knowledge from Humans and Domains
- Graph Learning Enhanced by Knowledge from External Sources
- Knowledge-enhanced Graph Learning for Real-world Applications
- Summary and Future Directions

Graph Data are Complex

Data Property

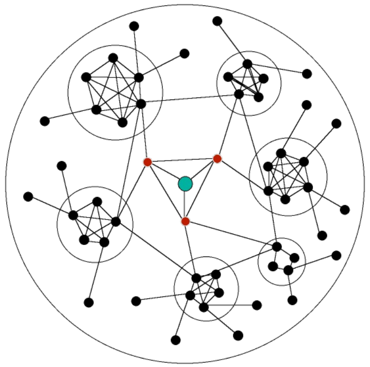


Graph Structure

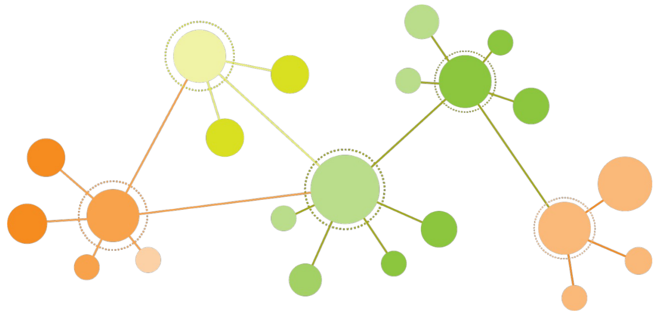


Node Features

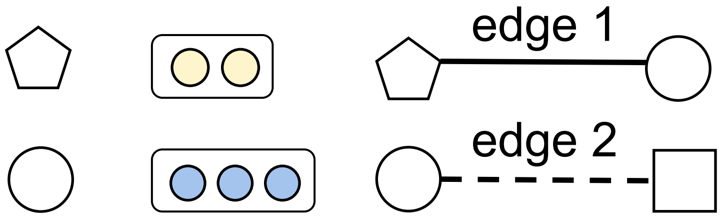
Knowledge



Local communities



Node positions



Heterogeneous Information

	p1	p2	p3
p1	1	1	0
p2	1	1	1
p3	0	1	1

Co-author

	p1	p2	p3
p1	1	1	0
p2	1	1	0
p3	0	0	1

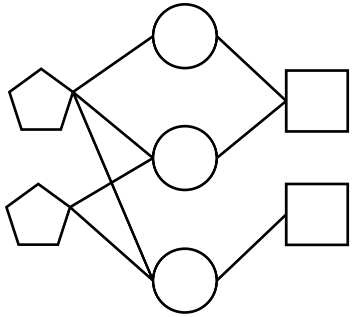
Co-venue

Higher-order Semantics

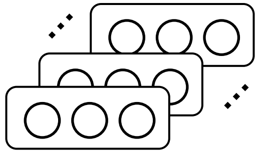
Fundamental GNNs Focus on Data Property

Data Property

$$GNN = f(\text{Graph Structure}, \text{Node Features})$$



Graph Structure



Node Features

Model	Conference	Citation
GCN	ICLR 2017	29000+
GraphSAGE	NeurIPS 2017	12000+
GAT	ICLR 2018	8000+
GIN	ICLR 2019	6000+

Knowledge is missed

Taxonomy of Knowledge from Data

Knowledge can be obtained from

1) **single-instance** level perception
   - Node sampling for node positions

2) **multiple-instance** level perception
   - Path sampling for positional and semantic information
   - Subgraph sampling for community information

Taxonomy of Knowledge from Data

Knowledge can be obtained from

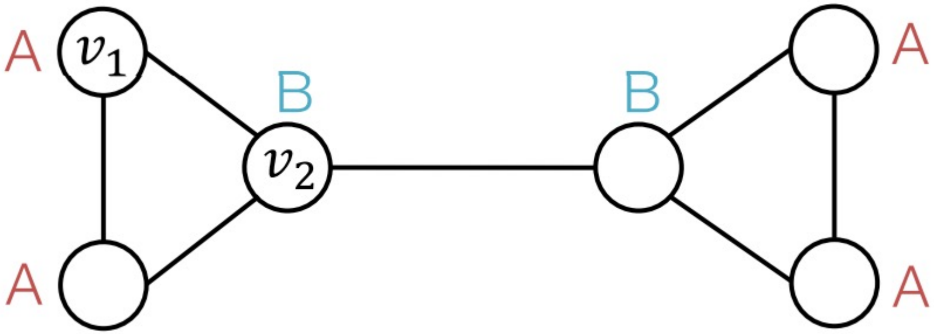
1) **single-instance** level perception
   - Node sampling for node positions

2) **multiple-instance** level perception
   - Path sampling for positional and semantic information
   - Subgraph sampling for community information

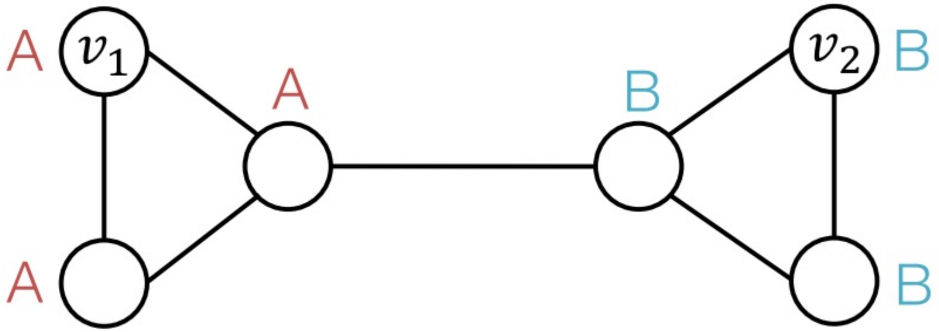
Node Sampling for Node Positions

Why do we need to emphasize node position?

Different types of tasks



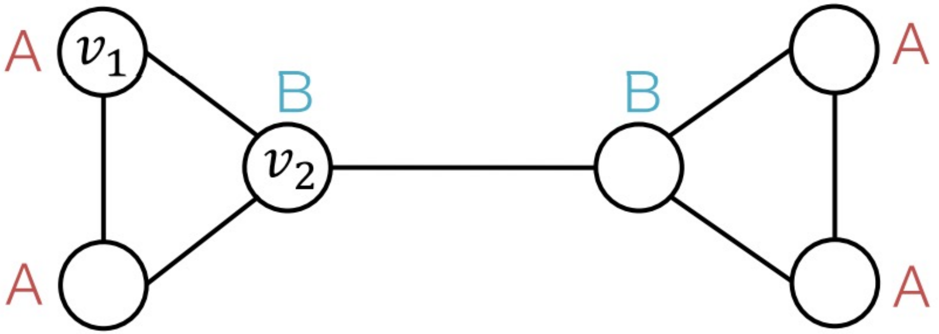
Structure-aware task: nodes labeled by structural roles



Position-aware task: nodes labeled by positions

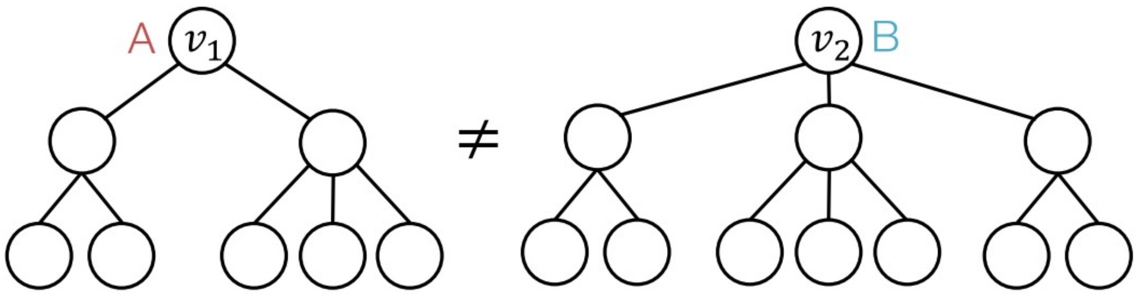
Node Sampling for Node Positions

GNNs usually work well on structure-aware task



Structure-aware task: nodes labeled by structural roles

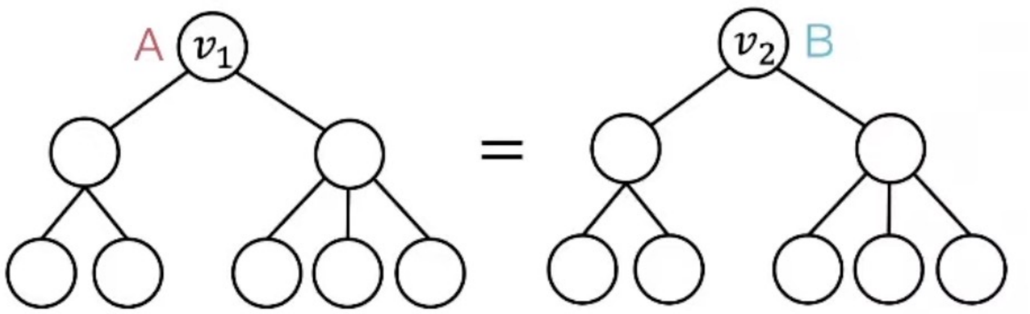
Computation graph of nodes v_1 and v_2 differs

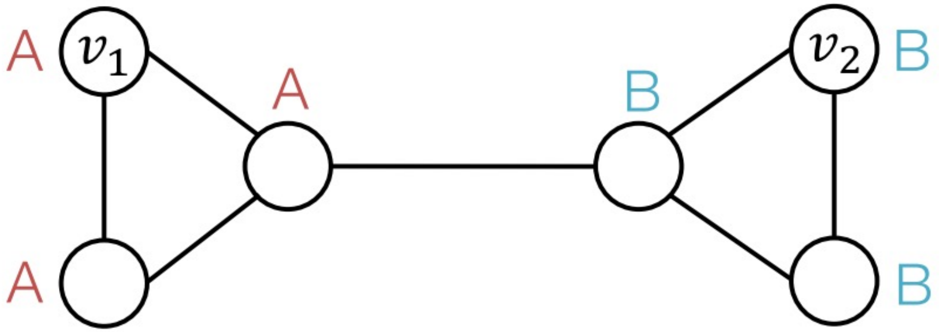


Node Sampling for Node Positions

GNNs usually perform poorly on position-aware task

Computation graph of nodes v_1 and v_2 are the same





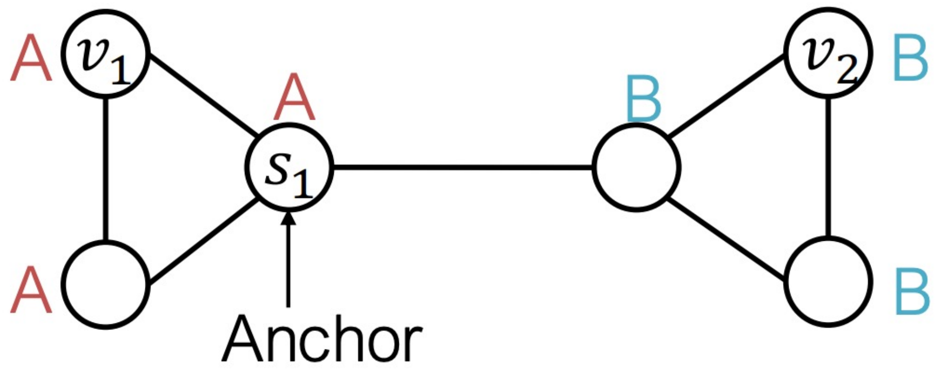
Position-aware task: nodes labeled by positions

Node Sampling for Node Positions

To encode node positions, define anchor

- Randomly pick a node as an anchor node
- Represent v_1 and v_2 with shortest distances to the anchor

	s_1
v_1	1
v_2	2



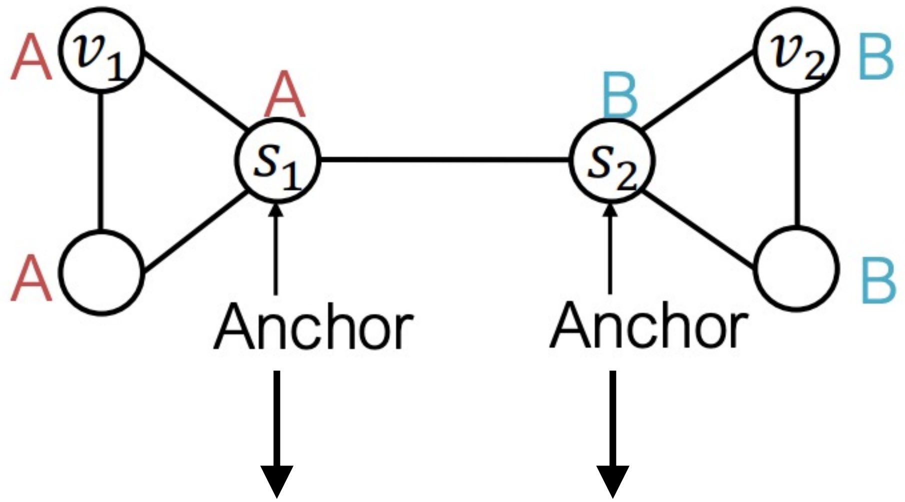
a coordinate axis

Node Sampling for Node Positions

To encode node positions, define anchor

- pick more nodes as anchors for more coordinate axes
- Represent v_1 and v_2 with shortest distances to these anchors

	s_1	s_2
v_1	1	2
v_2	2	1



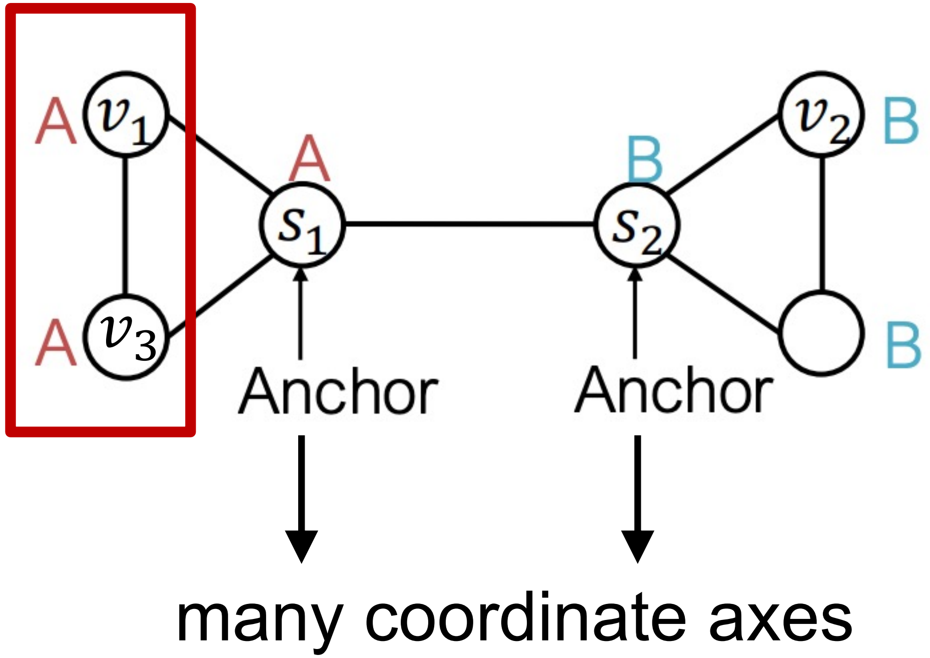
many coordinate axes

Node Sampling for Node Positions

To encode node positions, define anchor

- Problem: v_1 and v_3 have same shortest distances to these anchors

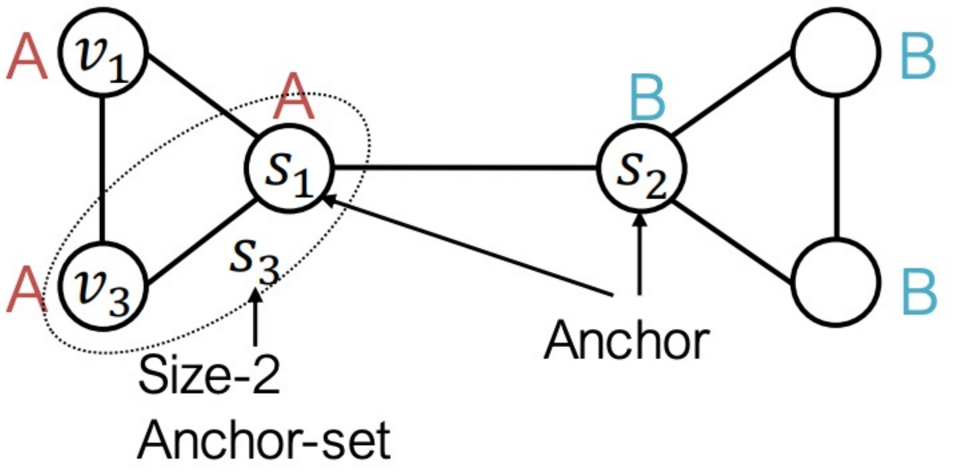
	s_1	s_2
v_1	1	2
v_3	1	2



Node Sampling for Node Positions

To encode node positions, define anchor

- Include multiple nodes in an anchor: anchor-set
- Better position estimation/encoding



	s_1	s_2	s_3
v_1	1	2	1
v_3	1	2	0

Anchor-set can distinguish v_1 and v_3, while anchor cannot

Node Sampling for Node Positions

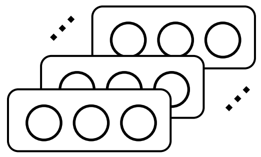
To use obtained positional encoding

We can simply concatenate it with node features and use them as usual

concatenate (

	s_1	s_2	s_3
v_1	1	2	1
v_3	1	2	0

,



)

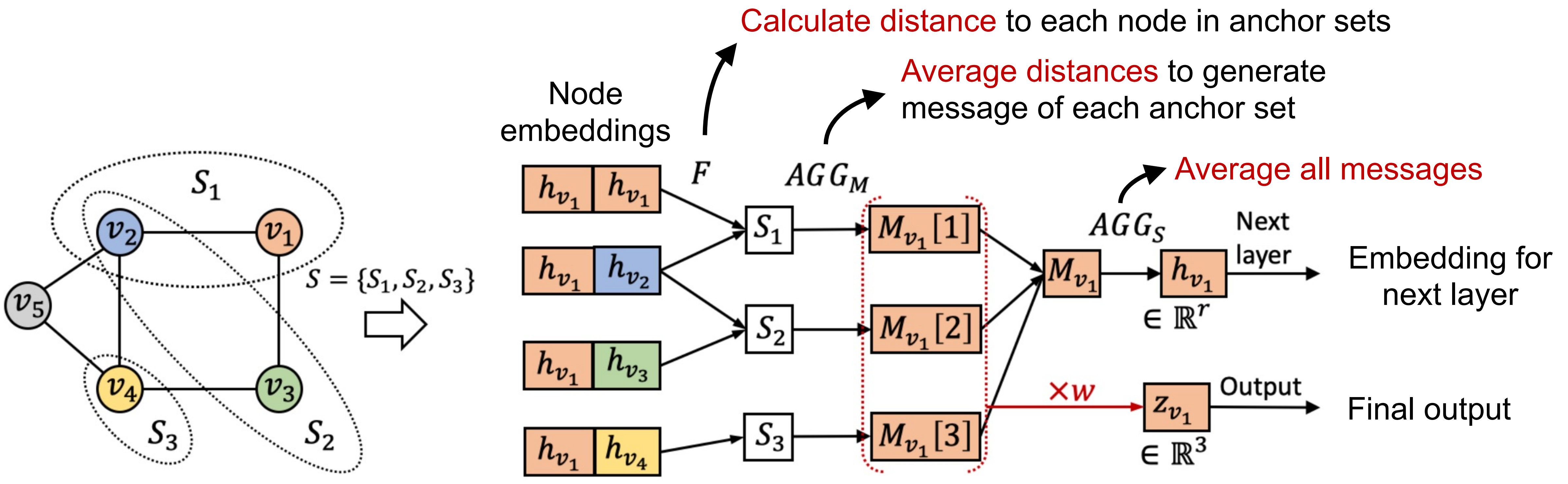
Positional encoding

Node features

Problem: sending this concatenated feature to a neural network cannot preserve the permutation invariant property of positional encoding

Node Sampling for Node Positions

To use obtained positional encoding, design P-GNN with permutation invariant aggregation (e.g., mean)



Anchor-set selection

Embedding computation for node v_1

Node Sampling for Node Positions

P-GNNs outperforms GNNs

Link Prediction task, measured in ROC AUC

	Grid-T	Communities-T	Grid	Communities	PPI
GCN	0.698 ± 0.051	0.981 ± 0.004	0.456 ± 0.037	0.512 ± 0.008	0.769 ± 0.002
GraphSAGE	0.682 ± 0.050	0.978 ± 0.003	0.532 ± 0.050	0.516 ± 0.010	0.803 ± 0.005
GAT	0.704 ± 0.050	0.980 ± 0.005	0.566 ± 0.052	0.618 ± 0.025	0.783 ± 0.004
GIN	0.732 ± 0.050	0.984 ± 0.005	0.499 ± 0.054	0.692 ± 0.049	0.782 ± 0.010
P-GNN-F-1L	0.542 ± 0.057	0.930 ± 0.093	0.619 ± 0.080	0.939 ± 0.083	0.719 ± 0.027
P-GNN-F-2L	0.637 ± 0.078	$\mathbf{0.989} \pm 0.003$	0.694 ± 0.066	$\mathbf{0.991} \pm 0.003$	0.805 ± 0.003
P-GNN-E-1L	0.665 ± 0.033	0.966 ± 0.013	0.879 ± 0.039	0.985 ± 0.005	0.775 ± 0.029
P-GNN-E-2L	$\mathbf{0.834} \pm 0.099$	0.988 ± 0.003	$\mathbf{0.940} \pm 0.027$	0.985 ± 0.008	$\mathbf{0.808} \pm 0.003$

Different variants of P-GNN with fast or regular way of calculating shortest path

+66%

When graphs come with rich features (e.g., PPI dataset), the performance improvement is smaller, because node features may already capture positional information

Taxonomy of Knowledge from Data

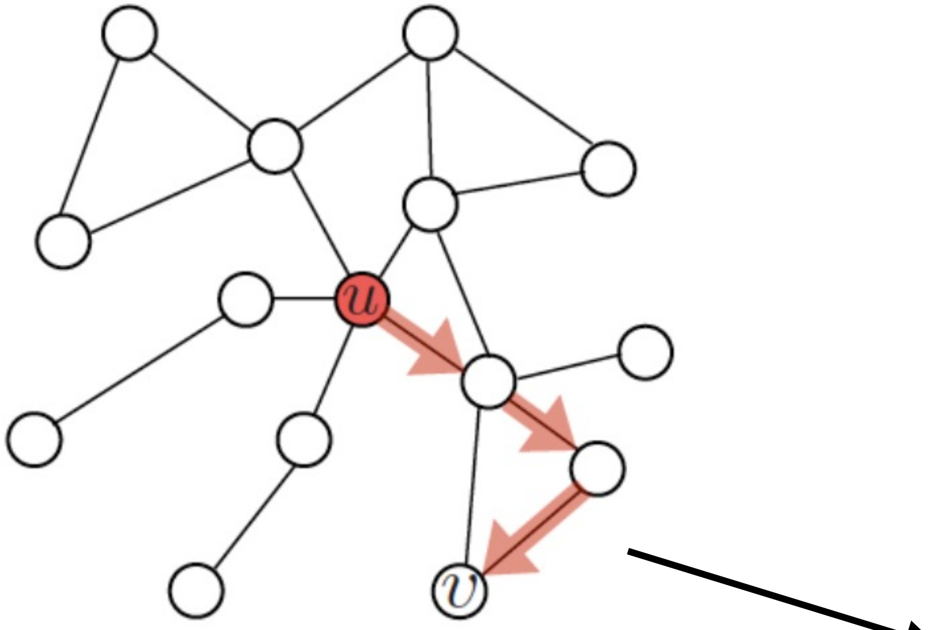
Knowledge can be obtained from

1) **single-instance** level perception
   - Node sampling for node positions

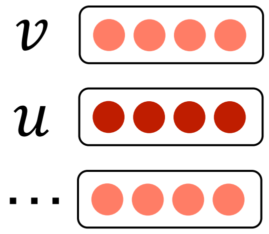
2) **multiple-instance** level perception
   - Path sampling for positional and semantic information
   - Subgraph sampling for community information

Path Sampling for Positions and Semantics

Sample a path of nodes for positions, e.g., via random walk



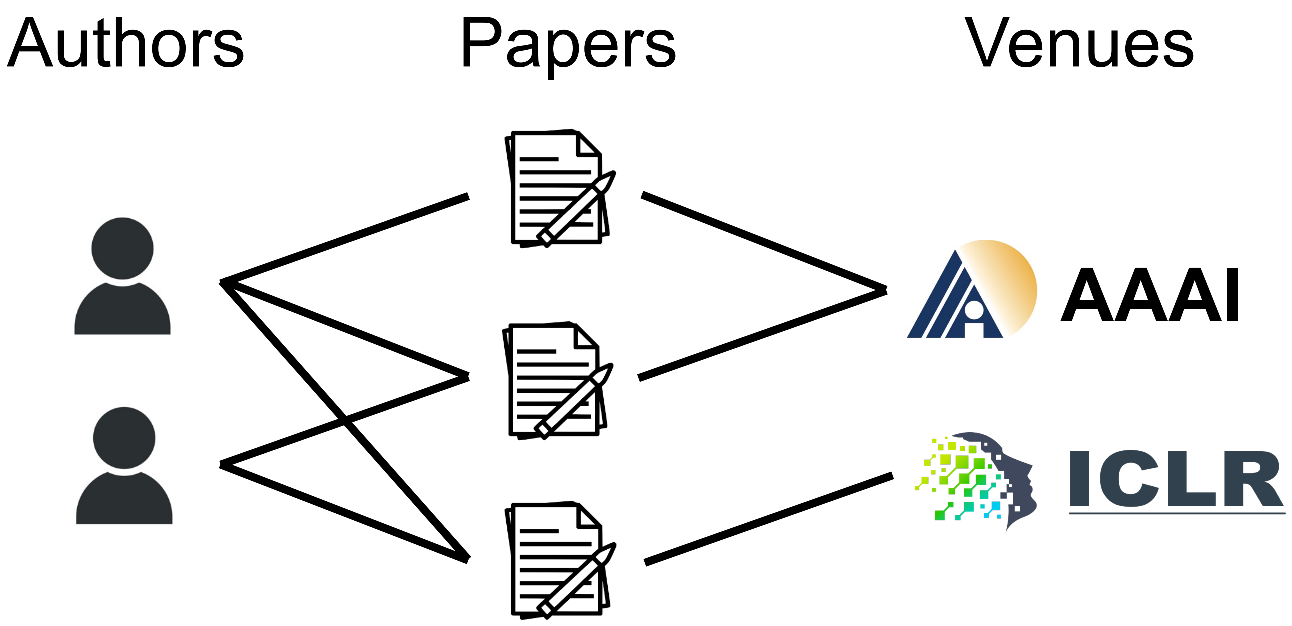
- Nodes appeared in a sampled path are similar
- Similarity between nodes in a graph translated to closeness in embedding



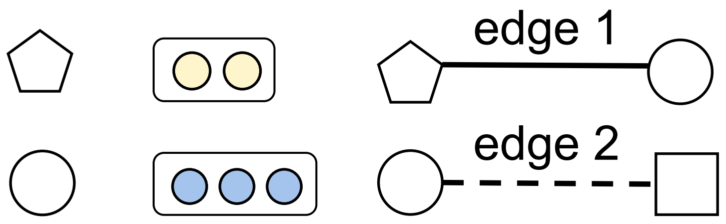
Positional embedding

Path Sampling for Positions and Semantics

Heterogeneous graphs contain rich information with semantics



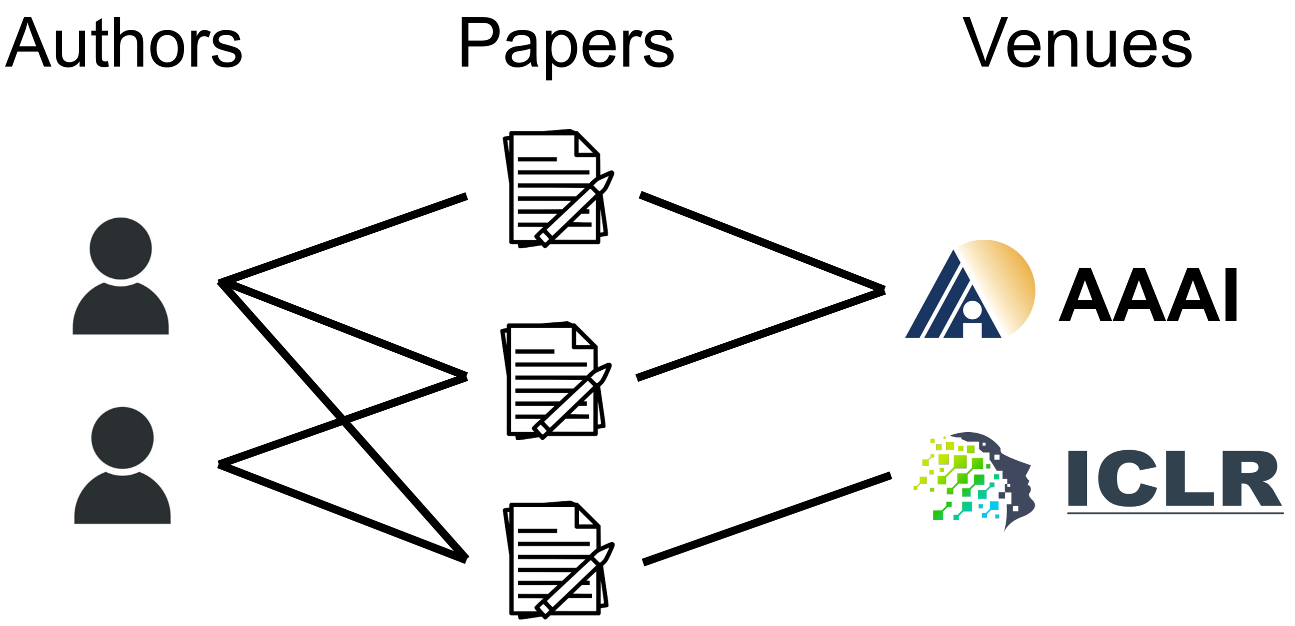
Heterogeneous graph



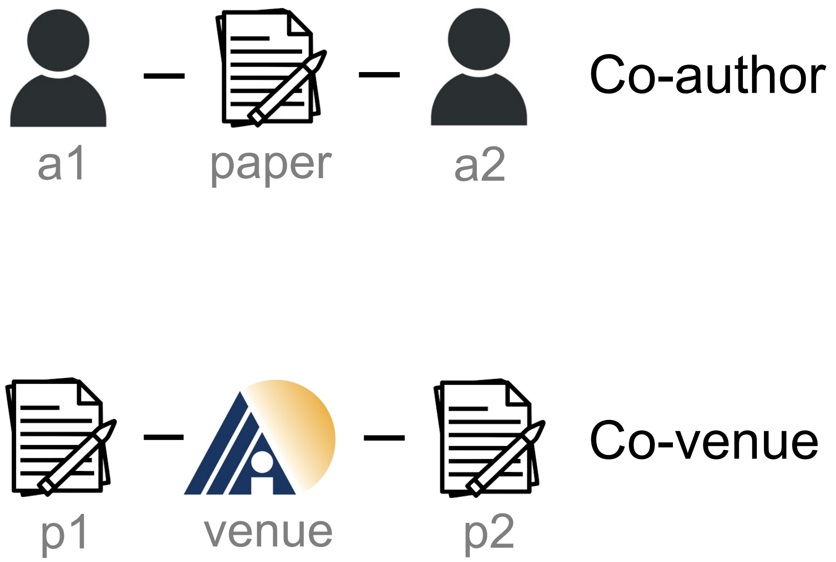
Heterogeneous Information

Path Sampling for Positions and Semantics

Sample a path of nodes for semantics, e.g., via metapaths



Heterogeneous graph



metapaths

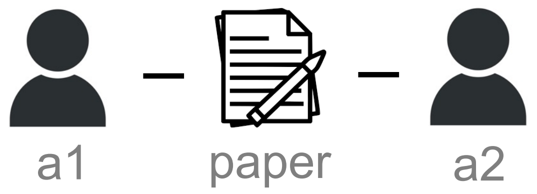
metapath2vec: Scalable Representation Learning for Heterogeneous Networks

Path Sampling for Positions and Semantics

Sample a path of nodes for semantics, e.g., via metapaths

Co-author

	p1	p2	p3
p1	1	1	0
p2	1	1	1
p3	0	1	1



Co-author

Co-venue

	p1	p2	p3
p1	1	1	0
p2	1	1	0
p3	0	0	1

Co-venue

Higher-order Semantics

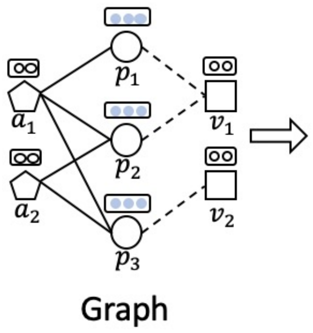
metapaths

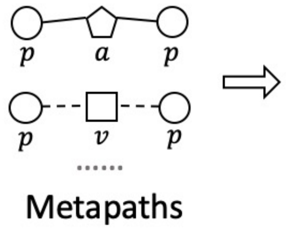
metapath2vec: Scalable Representation Learning for Heterogeneous Networks

Path Sampling for Positions and Semantics

To encode the positions and semantics, introducing HGMAE

Data | Knowledge Extraction | Learning Strategies | Regular GNN | Learning Objectives



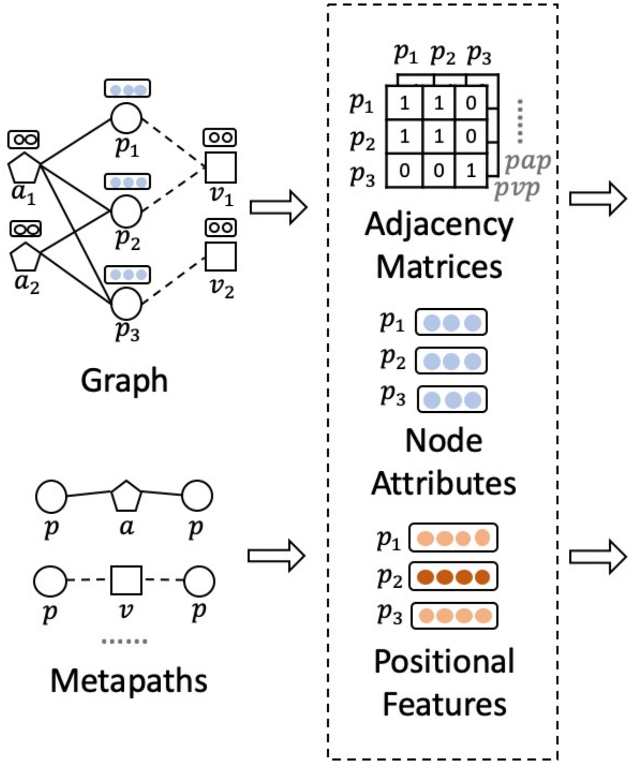


Metapaths involve the semantic knowledge

Path Sampling for Positions and Semantics

To encode the positions and semantics, introducing HGMAE

Data | Knowledge Extraction | Learning Strategies | Regular GNN | Learning Objectives



Semantics

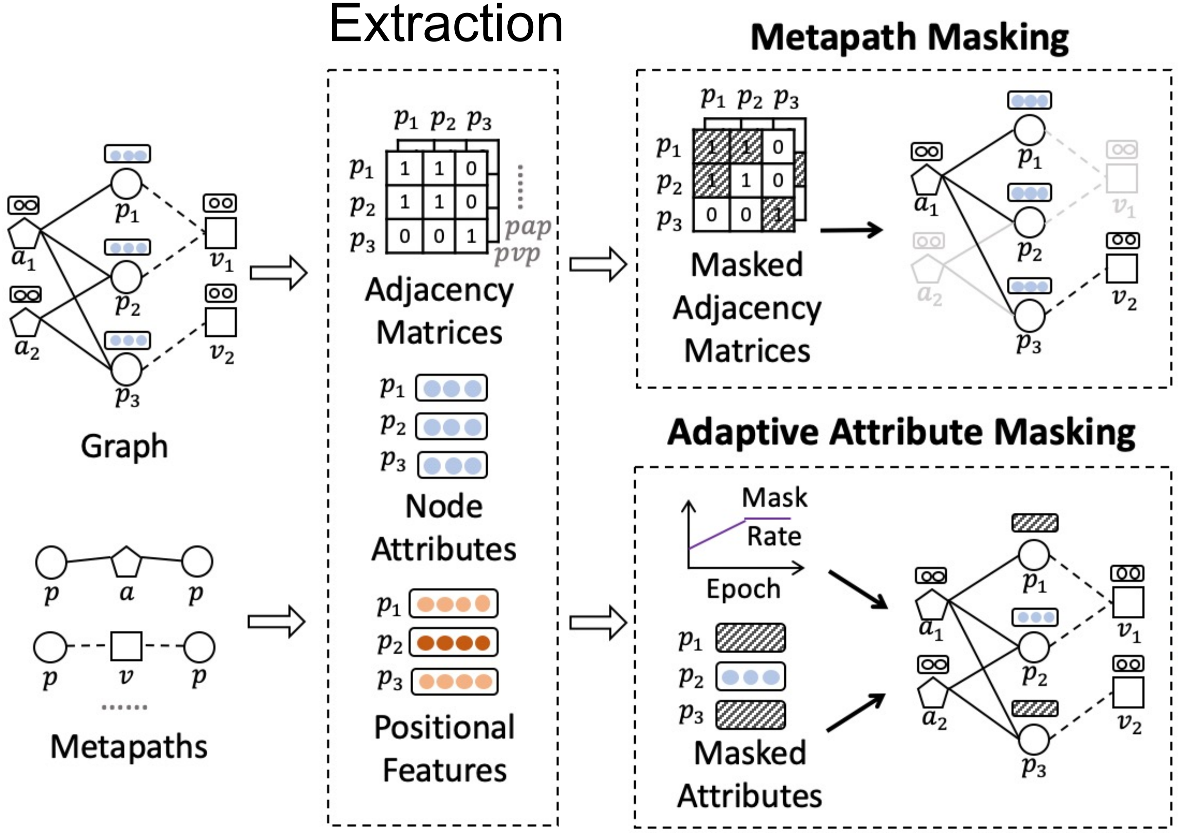
Heterogeneous info

Node Positions

Path Sampling for Positions and Semantics

To encode the positions and semantics, introducing HGMAE

Data | Knowledge Extraction | Learning Strategies | Regular GNN | Learning Objectives

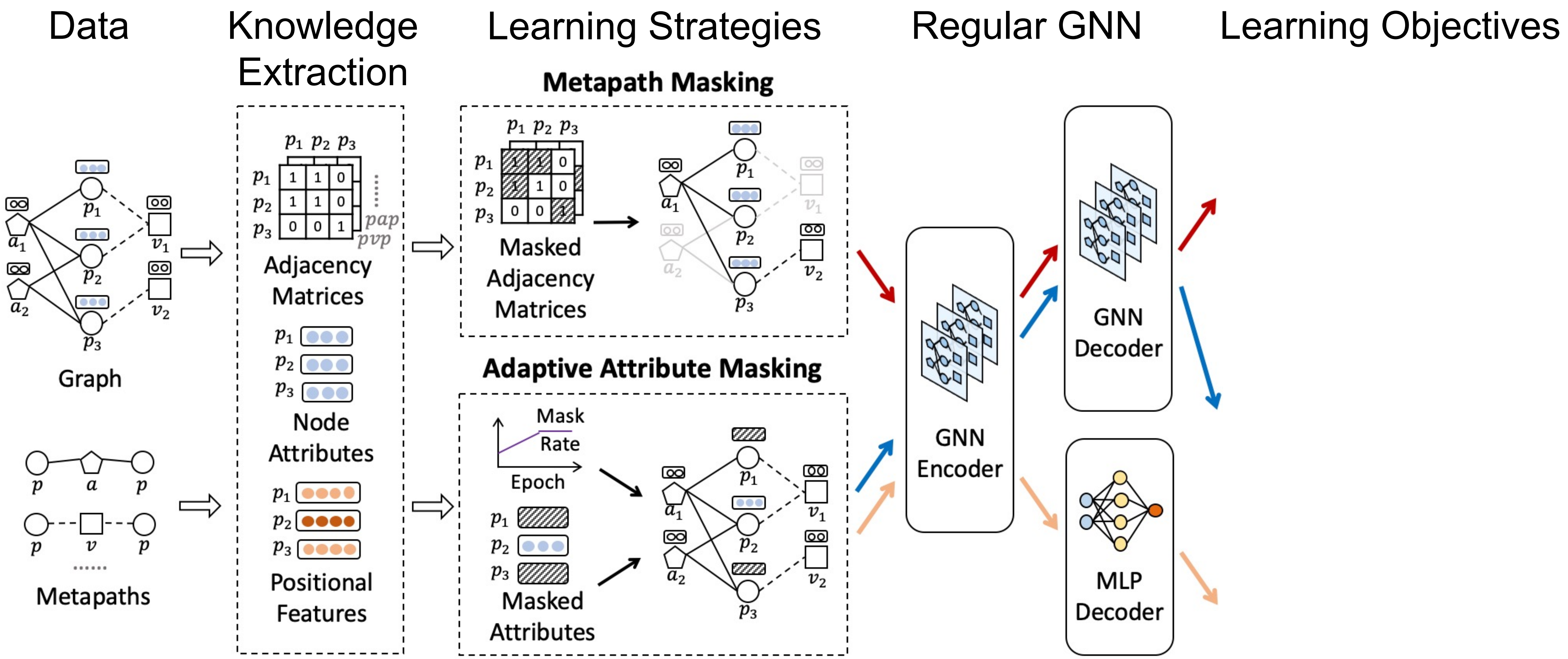


Capture semantic knowledge

Capture heterogeneous info and node positions

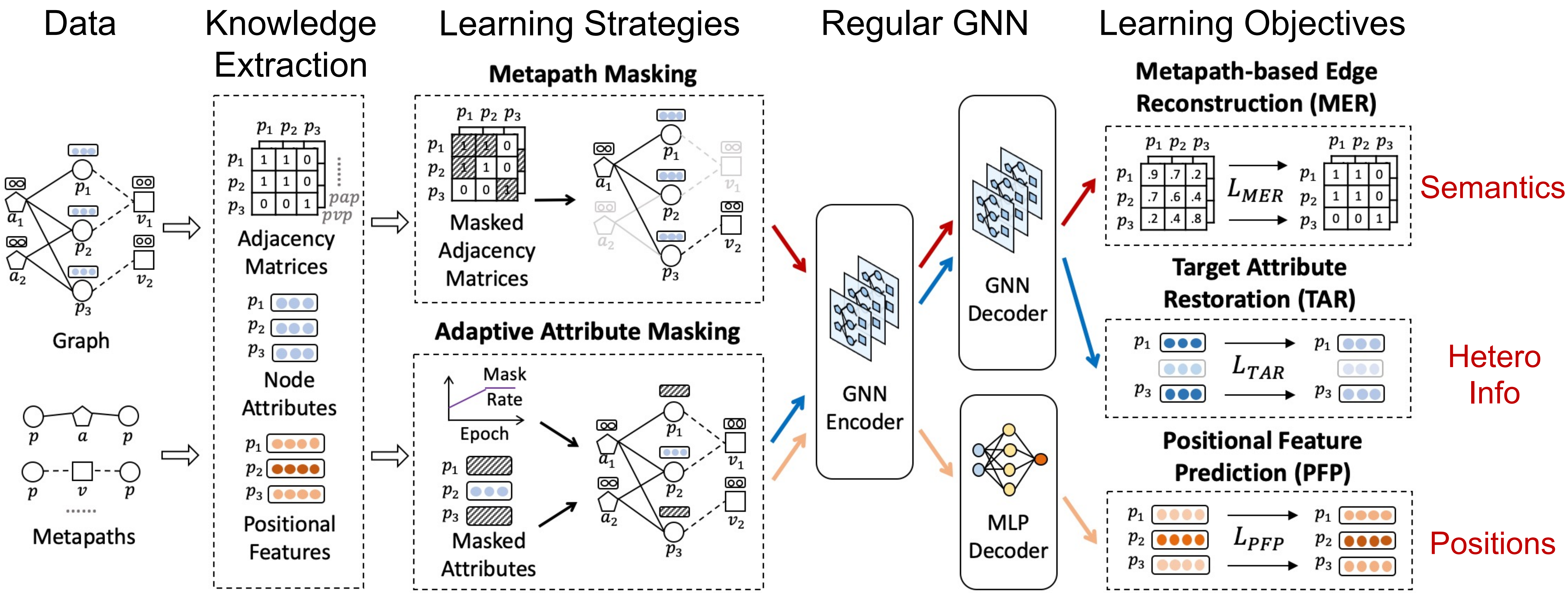
Path Sampling for Positions and Semantics

To encode the positions and semantics, introducing HGMAE



Path Sampling for Positions and Semantics

To encode the positions and semantics, introducing HGMAE



Path Sampling for Positions and Semantics

To encode the positions and semantics, introducing HGMAE

Performance Comparison: Node Classification

Datasets	Metric	Split	GraphSAGE	GAE	Mp2vec	HERec	HetGNN	HAN	DGI	DMGI	HeCo	GraphMAE	HGMAE
DBLP	Mi-F1	20	71.44±8.7	91.55±0.1	89.67±0.1	90.24±0.4	90.11±1.0	90.16±0.9	88.72±2.6	90.78±0.3	91.97±0.2	89.31±0.7	**92.71±0.5**
		40	73.61±8.6	90.00±0.3	89.14±0.2	90.15±0.4	89.03±0.7	89.47±0.9	89.22±0.5	89.92±0.4	90.76±0.3	87.80±0.5	**92.43±0.3**
		60	74.05±8.3	90.95±0.2	91.17±0.1	91.01±0.3	90.43±0.6	90.34±0.8	90.35±0.8	90.66±0.5	91.59±0.2	89.82±0.4	**93.05±0.3**
	Ma-F1	20	71.97±8.4	90.90±0.1	88.98±0.2	89.57±0.4	89.51±1.1	89.31±0.9	87.93±2.4	89.94±0.4	91.28±0.2	87.94±0.7	**92.28±0.5**
		40	73.69±8.4	89.60±0.3	88.68±0.2	89.73±0.4	88.61±0.8	88.87±1.0	88.62±0.6	89.25±0.4	90.34±0.3	86.85±0.7	**92.12±0.3**
		60	73.86±8.1	90.08±0.2	90.25±0.1	90.18±0.3	89.56±0.5	89.20±0.8	89.19±0.9	89.46±0.6	90.64±0.3	88.07±0.6	**92.33±0.3**
	AUC	20	90.59±4.3	98.15±0.1	97.69±0.0	98.21±0.2	97.96±0.4	98.07±0.6	96.99±1.4	97.75±0.3	98.32±0.1	92.23±3.0	**98.90±0.1**
		40	91.42±4.0	97.85±0.1	97.08±0.0	97.93±0.1	97.70±0.3	97.48±0.6	97.12±0.4	97.23±0.2	98.06±0.1	91.76±2.5	**98.55±0.1**
		60	91.73±3.8	98.37±0.1	98.00±0.0	98.49±0.1	97.97±0.2	97.96±0.5	97.76±0.5	97.72±0.4	98.59±0.1	91.63±2.5	**98.89±0.1**
Freebase	Mi-F1	20	54.83±3.0	55.20±0.7	56.23±0.8	57.92±0.5	56.85±0.9	57.24±3.2	58.16±0.9	58.26±0.9	61.72±0.6	64.88±1.8	**65.15±1.3**
		40	57.08±3.2	56.05±2.0	61.01±1.3	62.71±0.7	53.96±1.1	63.74±2.7	57.82±0.8	54.28±1.6	64.03±0.7	62.34±1.0	**67.23±0.8**
		60	55.92±3.2	53.85±0.4	58.74±0.8	58.57±0.5	56.84±0.7	61.06±2.0	57.96±0.7	56.69±1.2	63.61±1.6	59.48±6.2	**67.44±1.2**
	Ma-F1	20	45.14±4.5	53.81±0.6	53.96±0.7	55.78±0.5	52.72±1.0	53.16±2.8	54.90±0.7	55.79±0.9	59.23±0.7	59.04±1.0	**62.06±1.0**
		40	44.88±4.1	52.44±2.3	57.80±1.1	59.28±0.6	48.57±0.5	59.63±2.3	53.40±1.4	49.88±1.9	61.19±0.6	56.40±1.1	**64.64±0.9**
		60	45.16±3.1	50.65±0.4	55.94±0.7	56.50±0.4	52.37±0.8	56.77±1.7	53.81±1.1	52.10±0.7	60.13±1.3	51.73±2.3	**63.84±1.0**
	AUC	20	67.63±5.0	73.03±0.7	71.78±0.7	73.89±0.4	70.84±0.7	73.26±2.1	72.80±0.6	73.19±1.2	76.22±0.8	72.60±0.2	**78.36±1.1**
		40	66.42±4.7	74.05±0.9	75.51±0.8	76.08±0.4	69.48±0.2	77.74±1.2	72.97±1.1	70.77±1.6	78.44±0.5	72.44±1.6	**79.69±0.7**
		60	66.78±3.5	71.75±0.4	74.78±0.4	74.89±0.4	71.01±0.5	75.69±1.5	73.32±0.9	73.17±1.4	78.04±0.4	70.66±1.6	**79.11±1.3**

+7.5%

Path Sampling for Positions and Semantics

To encode the positions and semantics, introducing HGMAE

Performance Comparison: Node Clustering

Datasets	DBLP		Freebase		ACM		AMiner	
Metrics	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI
GraphSage	51.50	36.40	9.05	10.49	29.20	27.72	15.74	10.10
GAE	72.59	77.31	19.03	14.10	27.42	24.49	28.58	20.90
Mp2vec	73.55	77.70	16.47	17.32	48.43	34.65	30.80	25.26
HERec	70.21	73.99	19.76	19.36	47.54	35.67	27.82	20.16
HetGNN	69.79	75.34	12.25	15.01	41.53	34.81	21.46	26.60
DGI	59.23	61.85	18.34	11.29	51.73	41.16	22.06	15.93
DMGI	70.06	75.46	16.98	16.91	51.66	46.64	19.24	20.09
GraphMAE	65.86	69.75	19.43	20.05	47.03	46.48	17.98	21.52
HeCo	74.51	80.17	20.38	20.98	56.87	56.94	32.26	28.64
HGMAE	**76.92**	**82.34**	**22.05**	**22.84**	**66.68**	**71.51**	**41.10**	**38.27**

+17% +25% +27% +33%

Path Sampling for Positions and Semantics

To encode the positions and semantics, introducing HGMAE

Ablation Study

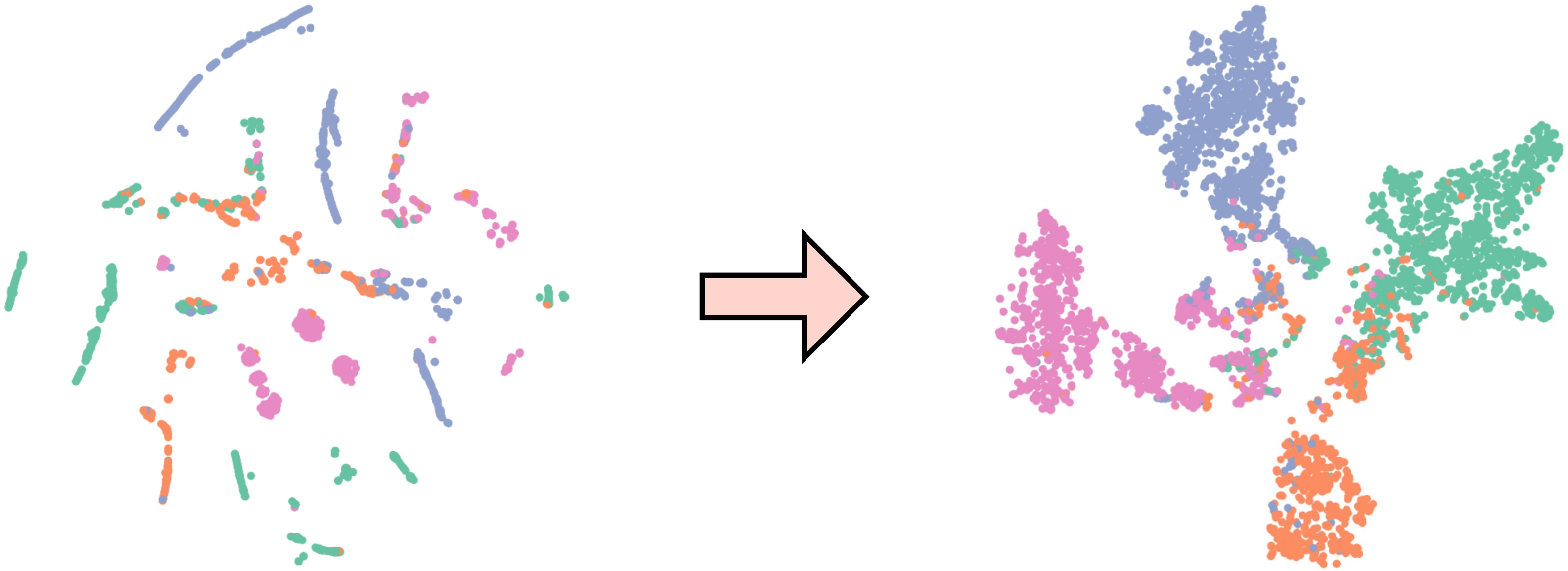
		Semantics	Hetero Info	Positions	
Datasets	Metric	w/o MER	w/o TAR	w/o PFP	HGMAE
ACM	Mi-F1	76.85±0.2	88.54±0.4	89.81±0.5	**90.59±0.5**
	Ma-F1	71.93±0.4	88.82±0.4	89.94±0.4	**90.80±0.5**
	AUC	84.84±1.5	96.47±0.1	97.22±0.1	**97.69±0.1**

- Removing the learning of each knowledge results in decreased performance
- By learning all knowledge, HGMAE achieves the best results

Path Sampling for Positions and Semantics

To encode the positions and semantics, introducing HGMAE

How valuable is knowledge?



w/o Knowledge
(GraphMAE)

w/ Knowledge
(HGMAE)

Taxonomy of Knowledge from Data

Knowledge can be obtained from

1) **single-instance** level perception
   - Node sampling for node positions

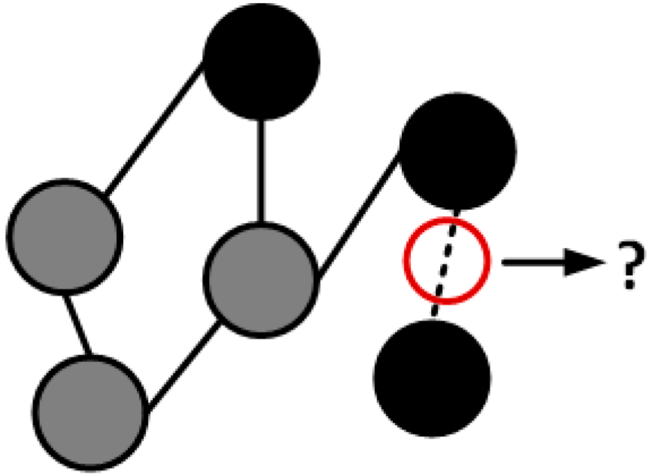
2) **multiple-instance** level perception
   - Path sampling for positional and semantic information
   - Subgraph sampling for community information

Subgraph Sampling for Community Information

What is link prediction?

Predicting missing or future links between pairs of nodes is important in social networks, recommender systems, biological networks, and academic graphs

- Friend Recommendation
- Product Recommendation
- Protein-protein Interaction
- Co-authorship Prediction
- …



Subgraph Sampling for Community Information

Heuristic methods for link prediction

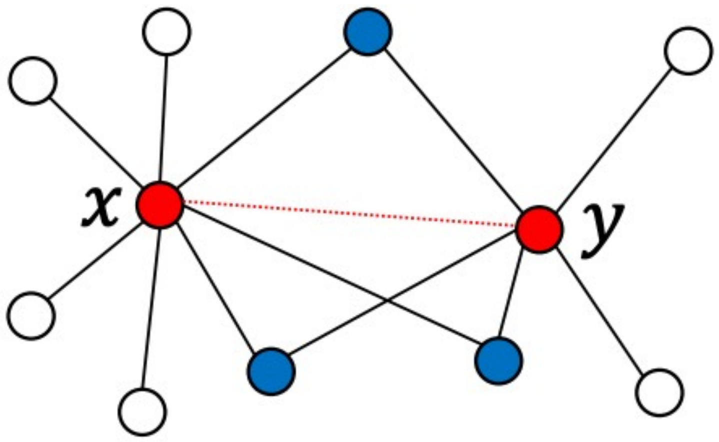
Name	Formula	Order
common neighbors	$\lvert\Gamma(x) \cap \Gamma(y)\rvert$	first
Jaccard	$\frac{\lvert\Gamma(x)\cap\Gamma(y)\rvert}{\lvert\Gamma(x)\cup\Gamma(y)\rvert}$	first
preferential attachment	$\lvert\Gamma(x)\rvert \cdot \lvert\Gamma(y)\rvert$	first
Adamic-Adar	$\sum_{z\in\Gamma(x)\cap\Gamma(y)} \frac{1}{\log\lvert\Gamma(z)\rvert}$	second
resource allocation	$\sum_{z\in\Gamma(x)\cap\Gamma(y)} \frac{1}{\lvert\Gamma(z)\rvert}$	second
Katz	$\sum_{l=1}^{\infty} \beta^l \lvert\text{walks}^{\langle l\rangle}(x,y)\rvert$	high
PageRank	$[\pi_x]_y + [\pi_y]_x$	high
SimRank	$\gamma \frac{\sum_{a\in\Gamma(x)}\sum_{b\in\Gamma(y)}\text{score}(a,b)}{\lvert\Gamma(x)\rvert\cdot\lvert\Gamma(y)\rvert}$	high

Pros:

- Easy calculation
- Interpretable
- Not dependent on training

Subgraph Sampling for Community Information

Heuristic methods for link prediction: Common Neighbor



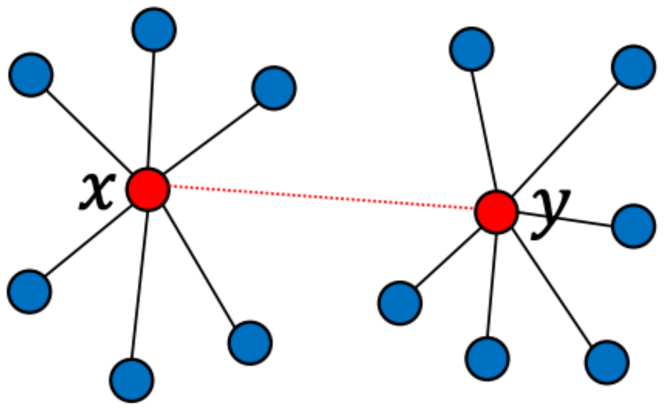
common neighbors (CN):
$|\Gamma(x) \cap \Gamma(y)|$

$\Gamma(x)$ denotes the neighbor set of node x

x and y are likely to have a link if they have many common neighbors

Subgraph Sampling for Community Information

Heuristic methods for link prediction: Preferential Attachment



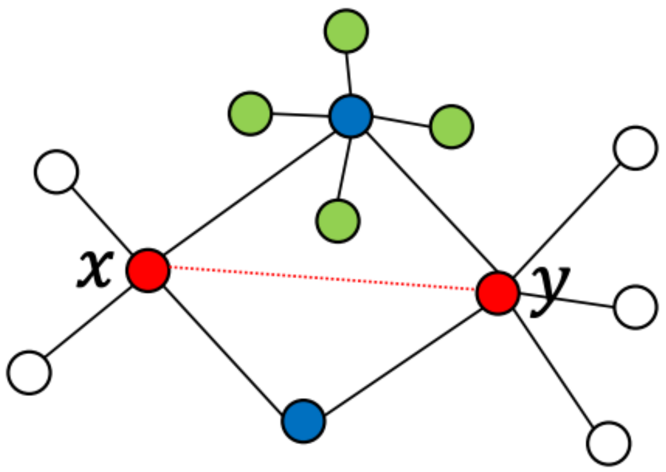
preferential attachment (PA):
$|\Gamma(x)| \cdot |\Gamma(y)|$

$\Gamma(x)$ denotes the neighbor set of node x

x is likely to connect to y if y has many connections

Subgraph Sampling for Community Information

Heuristic methods for link prediction: Adamic-Adar



Adamic-Adar (AA):

$$\sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log |\Gamma(z)|}$$

$\Gamma(x)$ denotes the neighbor set of node x

- Weighted common neighbors
- Popular common neighbors contribute less

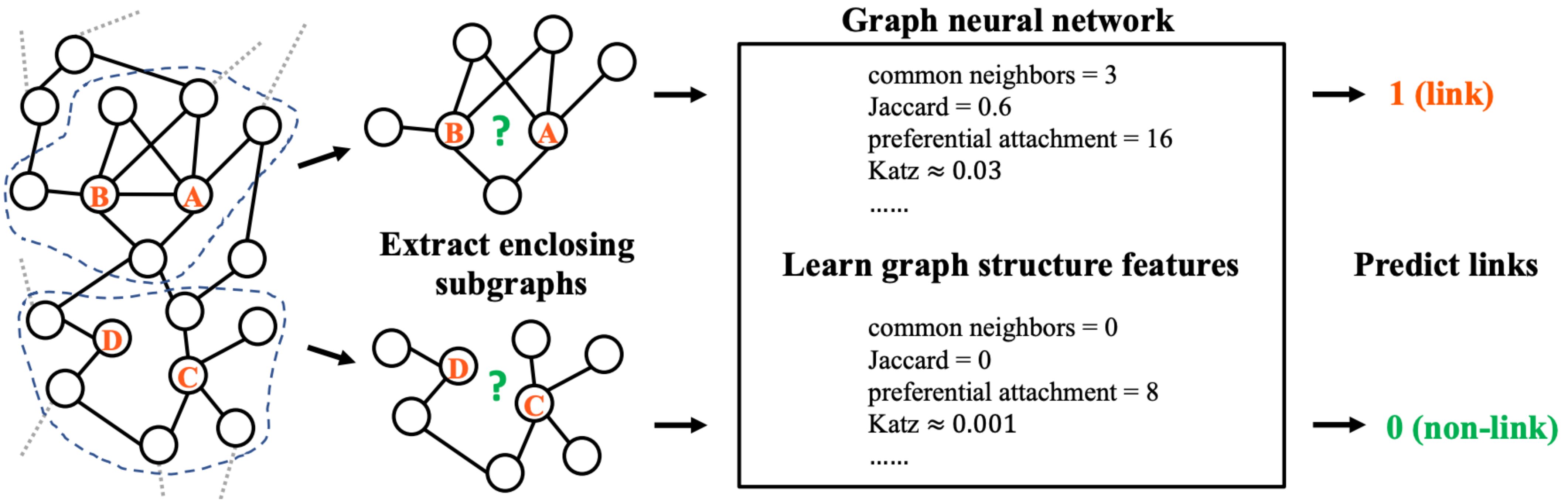
Subgraph Sampling for Community Information

Heuristic methods for link prediction: Problems

- Rule-based strategy, not learning-based
- Strong assumption on link formation mechanisms, which might only work well on certain graphs

Subgraph Sampling for Community Information

Sample a subgraph to enhance link prediction: SEAL



Subgraph Sampling for Community Information

Sample a subgraph to enhance link prediction: SEAL

Comparison with heuristic methods (AUC)

Data	CN	Jaccard	PA	AA	RA	Katz	PR	SR	ENS	WLK	WLNM	SEAL
USAir	93.80±1.22	89.79±1.61	88.84±1.45	95.06±1.03	95.77±0.92	92.88±1.42	94.67±1.08	78.89±2.31	88.96±1.44	**96.63**±0.73	95.95±1.10	**96.62**±0.72
NS	94.42±0.95	94.43±0.93	68.65±2.03	94.45±0.93	94.45±0.93	94.85±1.10	94.89±1.08	94.79±1.08	97.64±0.25	98.57±0.51	98.61±0.49	**98.85**±0.47
PB	92.04±0.35	87.41±0.39	90.14±0.45	92.36±0.34	92.46±0.37	92.92±0.35	93.54±0.41	77.08±0.80	90.15±0.45	93.83±0.59	93.49±0.47	**94.72**±0.46
Yeast	89.37±0.61	89.32±0.60	82.20±1.02	89.43±0.62	89.45±0.62	92.24±0.61	92.76±0.55	91.49±0.57	82.36±1.02	95.86±0.54	95.62±0.52	**97.91**±0.52
C.ele	85.13±1.61	80.19±1.64	74.79±2.04	86.95±1.40	87.49±1.41	86.34±1.89	**90.32**±1.49	77.07±2.00	74.94±2.04	89.72±1.67	86.18±1.72	**90.30**±1.35
Power	58.80±0.88	58.79±0.88	44.33±1.02	58.79±0.88	58.79±0.88	65.39±1.59	66.00±1.59	76.15±1.06	79.52±1.78	82.41±3.43	84.76±0.98	**87.61**±1.57
Router	56.43±0.52	56.40±0.52	47.58±1.47	56.43±0.51	56.43±0.51	38.62±1.35	38.76±1.39	37.40±1.27	47.58±1.48	87.42±2.08	94.41±0.88	**96.38**±1.45
E.coli	93.71±0.39	81.31±0.61	91.82±0.58	95.36±0.34	95.95±0.35	93.50±0.44	95.57±0.44	62.49±1.43	91.89±0.58	96.94±0.29	97.21±0.27	**97.64**±0.22

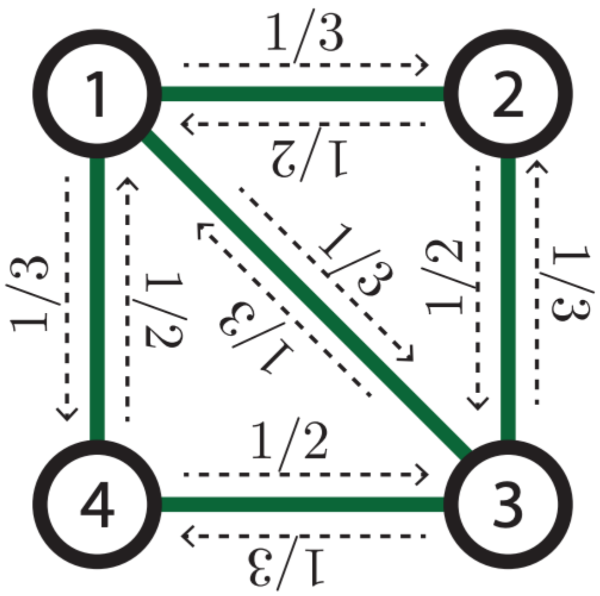
SEAL outperforms heuristic methods

Subgraph Sampling for Community Information

One step further: WalkPool

SEAL average the node embedding in subgraph for link prediction

WalkPool proposes that average pooling is suboptimal. Instead, using random walk to extract structural information

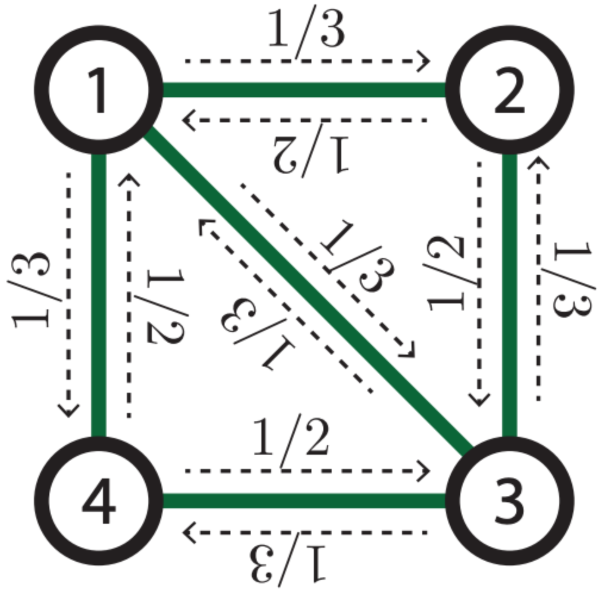


$$\mathbf{P} = \begin{pmatrix} 0 & 1/3 & 1/3 & 1/3 \\ 1/2 & 0 & 1/2 & 0 \\ 1/3 & 1/3 & 0 & 1/3 \\ 1/2 & 0 & 1/2 & 0 \end{pmatrix}$$

Random walk probabilities

Subgraph Sampling for Community Information

One step further: WalkPool

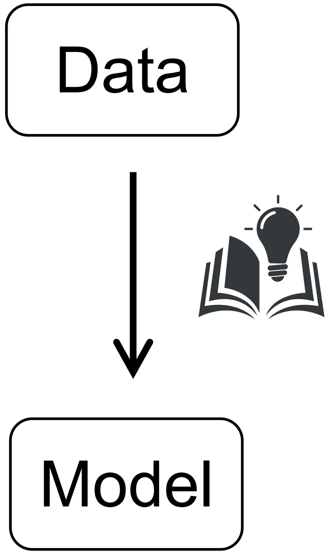


Features	$\tau = 2$ walk length
$\mathtt{node}_1^\tau$	$\left(\frac{1}{3}\cdot\frac{1}{2} + \frac{1}{3}\cdot\frac{1}{2} + \frac{1}{3}\cdot\frac{1}{3}\right)$ 1→2→1 1→4→1 1→3→1
$\mathtt{link}_{1,2}^\tau$	$\left(\frac{1}{3}\cdot\frac{1}{2}\right)$ 1→3→2
$\mathtt{graph}^\tau$	$\mathtt{node}_1^2 + \mathtt{node}_2^2 + \mathtt{node}_3^2 + \mathtt{node}_4^2$

Use these random walk features for link prediction via an MLP

Graph Learning Enhanced by Knowledge from Data

Chapter summary



Knowledge can be obtained from

1) **single-instance** level perception
   - Node sampling for node positions

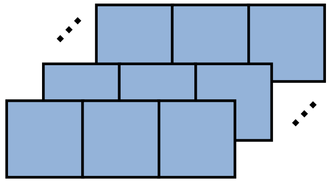
2) **multiple-instance** level perception
   - Path sampling for positional and semantic information
   - Subgraph sampling for community information

Tutorial Outline

- Preliminaries and Foundations
- Graph Learning Enhanced by Knowledge from Data
- Graph Learning Enhanced by Knowledge from Models
- Graph Learning Enhanced by Knowledge from Humans and Domains
- Graph Learning Enhanced by Knowledge from External Sources
- Knowledge-enhanced Graph Learning for Real-world Applications
- Summary and Future Directions

Models Contain Rich Knowledge

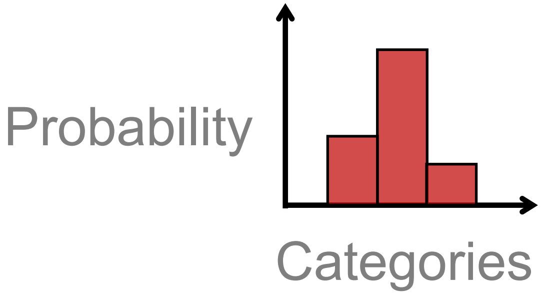
What is knowledge in models?



Learned Embeddings

	p1	p2	p3
p1	1	.3	.7
p2	.3	1	.2
p3	.7	.2	1

Data Similarity

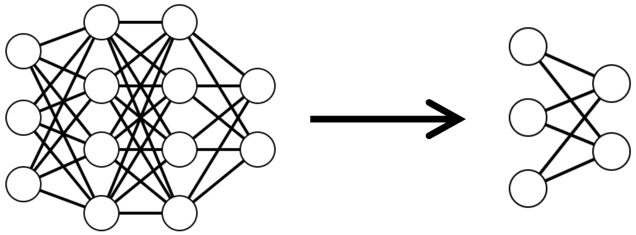


Logits

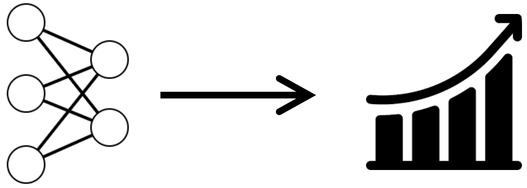
Why we need to emphasize model knowledge?

Two goals:

- Model Compression

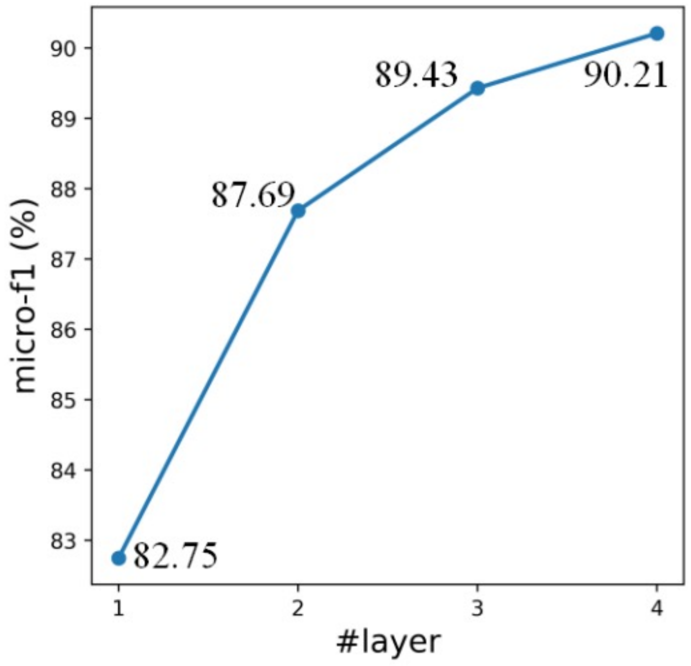


- Performance Improvement

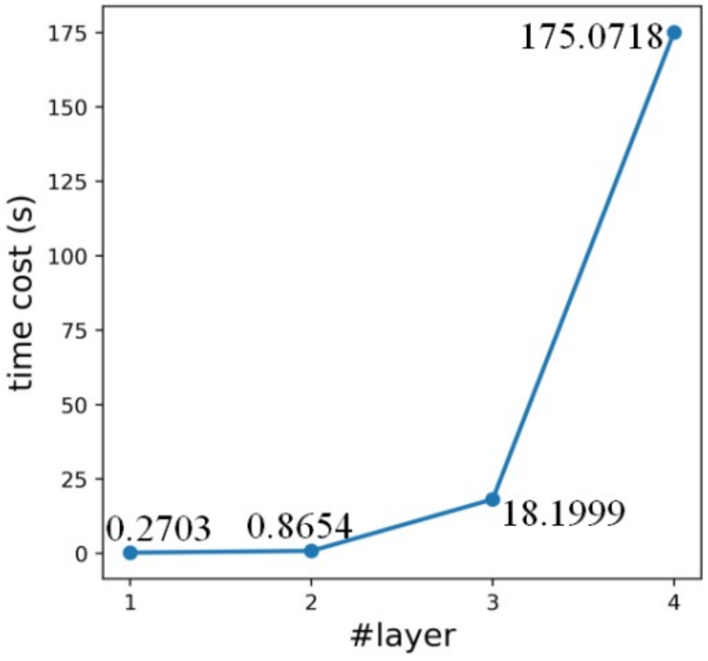


Models Contain Rich Knowledge

Deep GNNs are expensive



Micro-f1 of GAT on Facebook



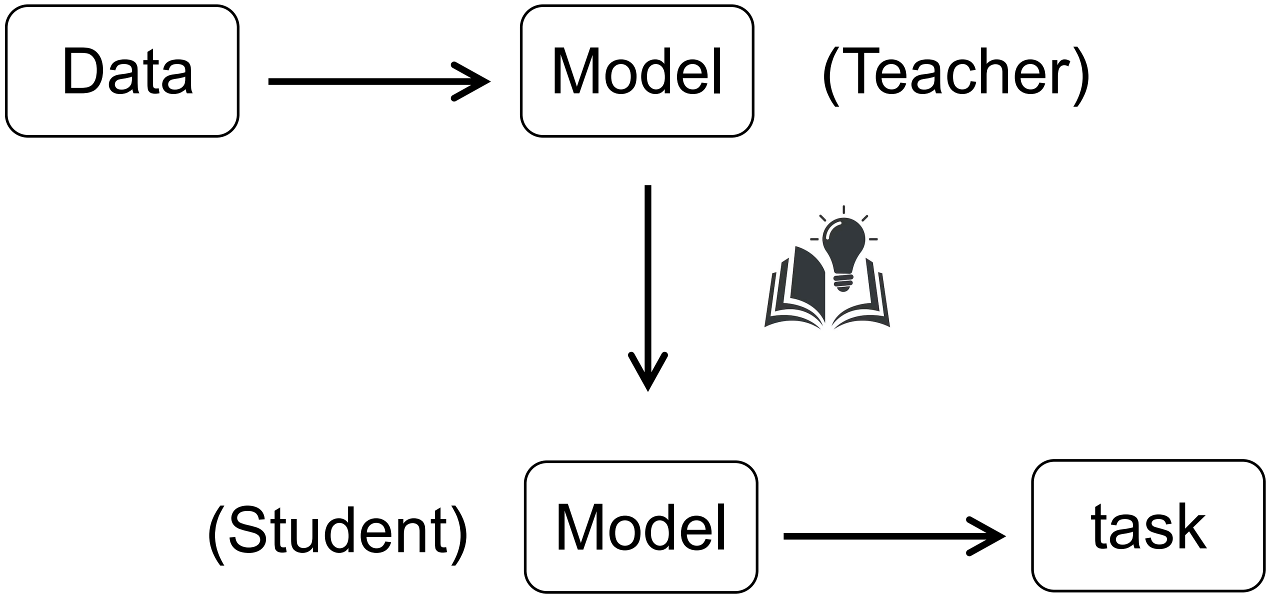
Time cost of GAT on Facebook

The more layers,
the higher performance,
the more expensive

Models Contain Rich Knowledge

How can we emphasize model knowledge?

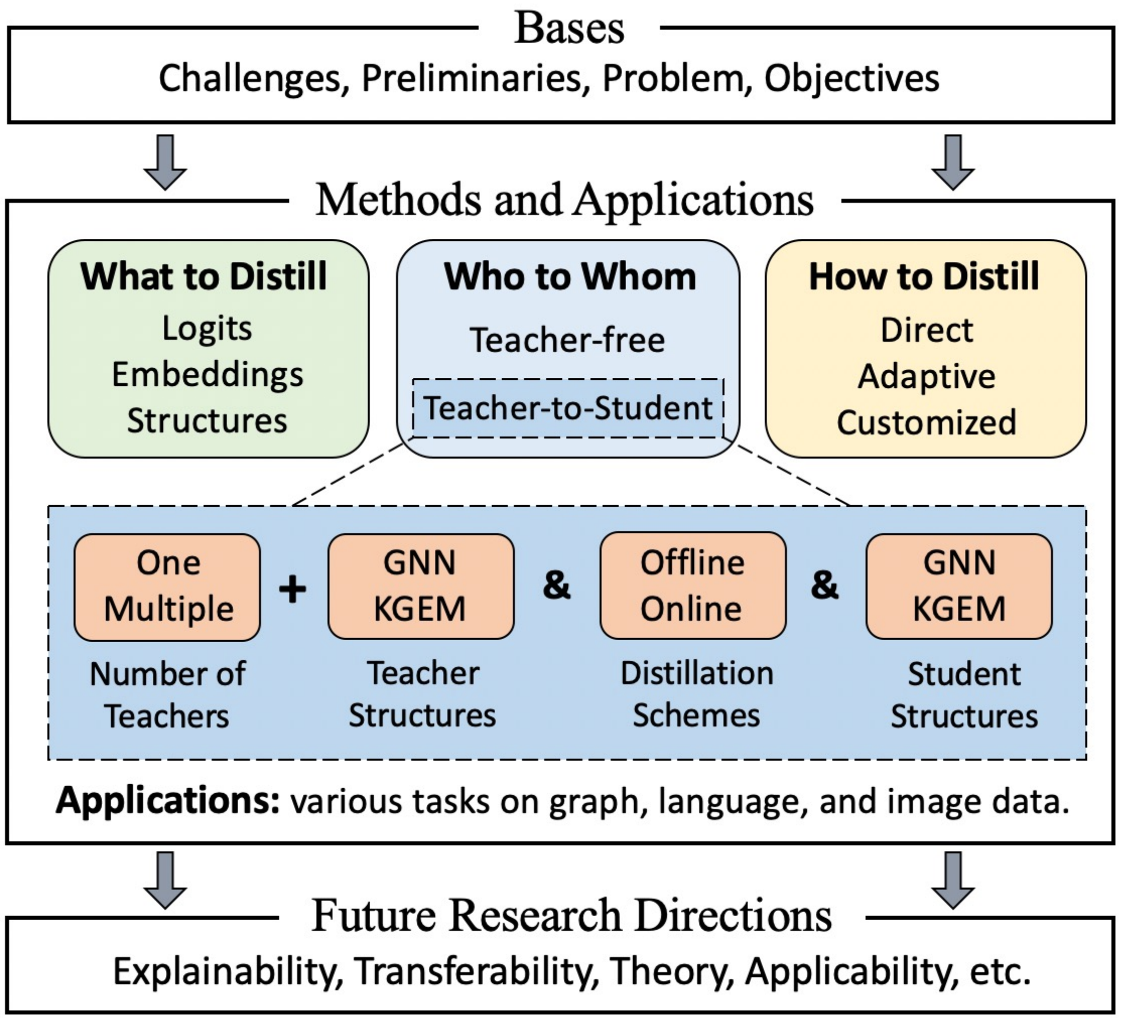
Distilling knowledge from models!



A Systematic Framework

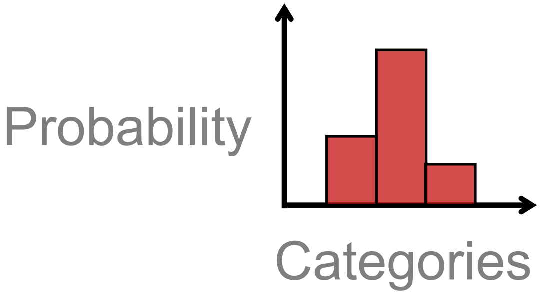
Three Main Aspects:

- What to Distill
- Who to Whom
- How to Distill

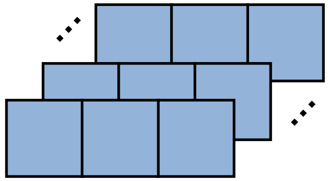


A Systematic Framework

What to distill?



Logits denote the inputs to the final SoftMax function and represent the soft label prediction



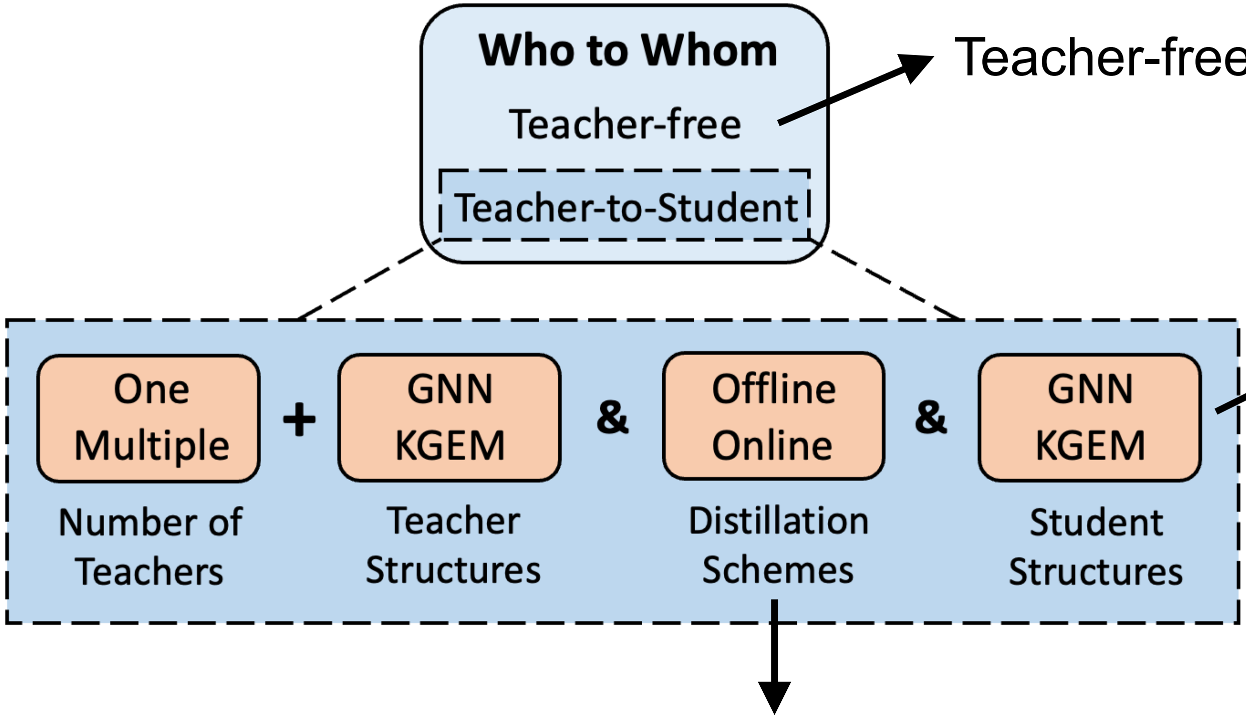
Embeddings are learned node embeddings from the intermediate layers of teacher models

1	.3	.7
.3	1	.2
.7	.2	1

Structures depict the connectivity and relationships between the elements in graph

A Systematic Framework

Who to whom?



Teacher-free: model can learn knowledge from itself

Teacher/Student Structures:

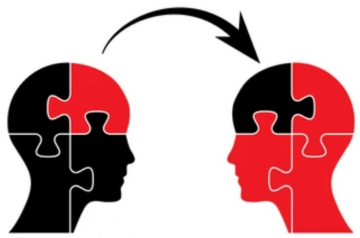
- Graph Neural Networks
- Knowledge Graph Embedding Methods
- Multi-layer Perceptron

Offline: teacher model is pre-trained and only student model is updated
Online: both the teacher model and the student model are trained end-to-end

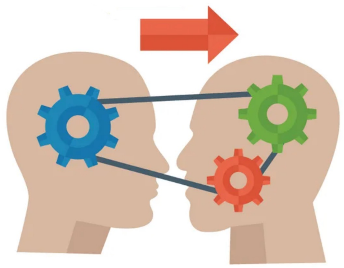
A Systematic Framework

How to distill?

Direct distillation minimizes the divergences between the knowledge of the teacher and the student directly



Adaptive distillation considers the significance of teacher knowledge adaptively



Customized distillation enables diversified designs according to the distinct objective of tasks in different scenarios

A Systematic Framework

A list of representative methods

Method	Objective	What to Distillate	Who to Whom	How to Distillate	Venue
TinyGNN[1]	Compression	Logits	GNN $\Rightarrow$ GNN	Direct	KDD'20
LSP[2]	Compression	Structures	GNN $\Rightarrow$ GNN	Direct	CVPR'20
RDD[3]	Performance	Logits, Embs	Teacher-free	Adaptive	SIGMOD'20
ROD[4]	Performance	Logits; Structures	Multi. GNNs $\Longleftrightarrow$ GNN	Adaptive	KDD'21
......					
GKD[20]	Compression	Structures	GNN $\Rightarrow$ GNN	Direct	NeurIPS'22
SAIL[21]	Performance	Embs	GNN $\Rightarrow$ GNN	Direct	AAAI'22
LTE4G[22]	Performance	Logits	Multi. GNNs $\Rightarrow$ GNN	Direct	CIKM'22
T2-GNN[23]	Performance	Logits; Embs	Multi. GNNs $\Rightarrow$ GNN	Direct	AAAI'23
RELIANT[24]	Compression	Logits	GNN $\Rightarrow$ GNN	Direct	SDM'23
BGNN[25]	Performance	Logits	Multi. GNNs $\Rightarrow$ GNN	Adaptive	AAAI'23
NOSMOG[26]	Compression	Logits; Structures	GNN $\Rightarrow$ MLP	Direct	ICLR'23

Covered Topics for Knowledge from Models

How to obtain knowledge from

1) Logits
2) Embeddings
3) Structures

A recent learning strategy

4) Distilling an MLP to replace GNNs

Covered Topics for Knowledge from Models

How to obtain knowledge from

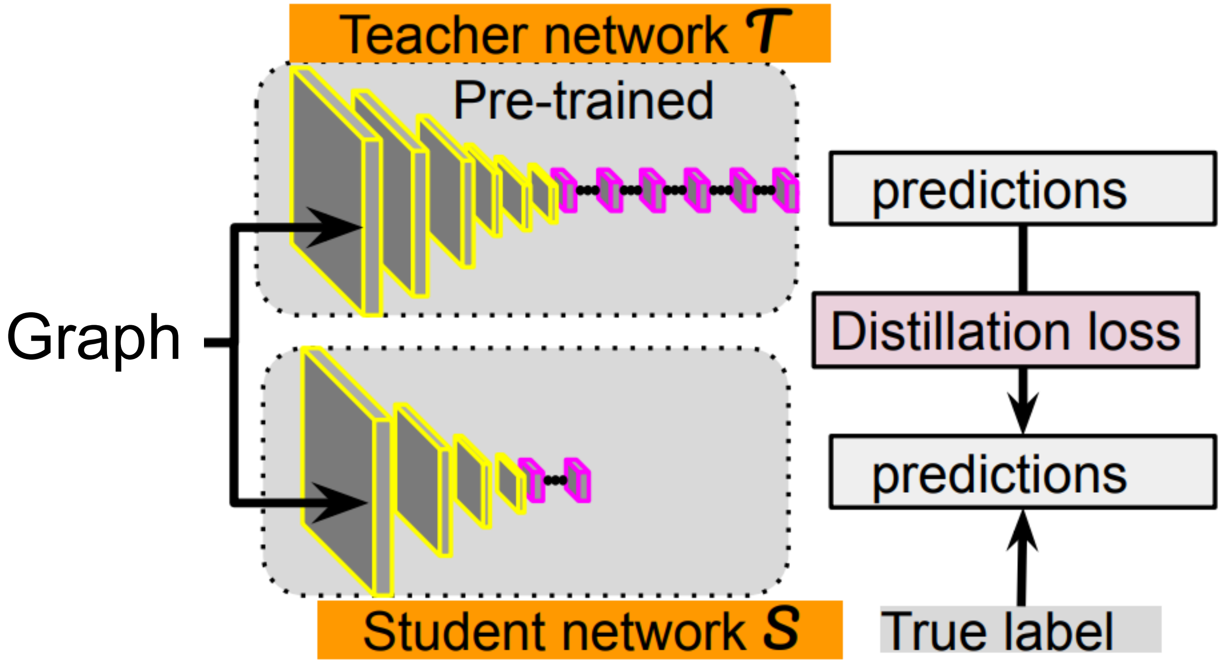
1) Logits
2) Embeddings
3) Structures

A recent learning strategy

4) Distilling an MLP to replace GNNs

Knowledge Distillation from Logits

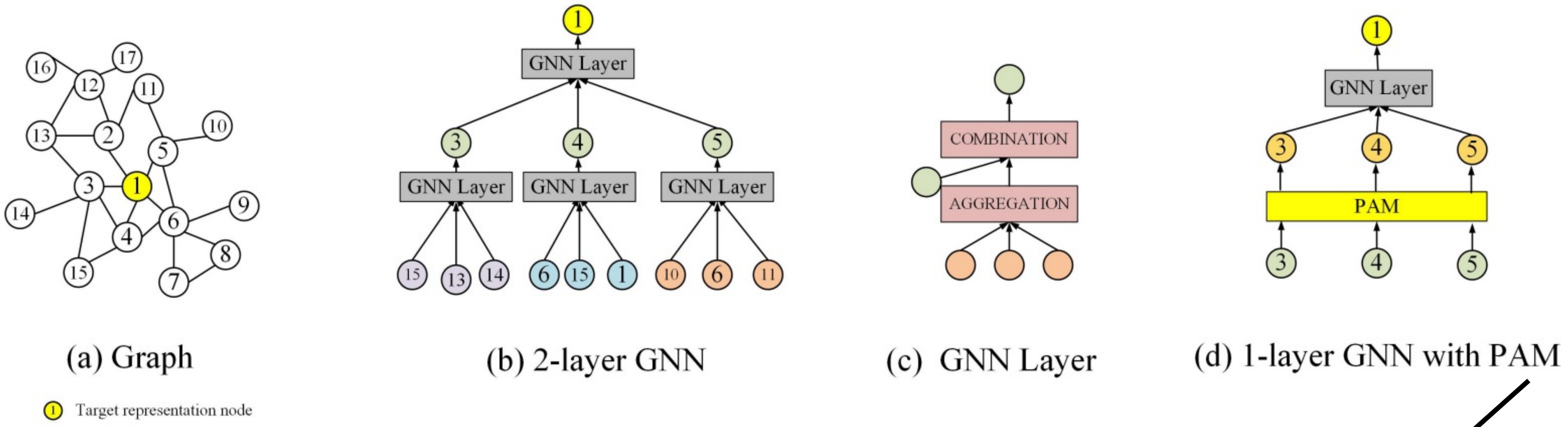
Illustration



Learning a smaller student network by mimicking the predictions of the teacher

Knowledge Distillation from Logits

Learn a small GNN with local structure enhanced: TinyGNN

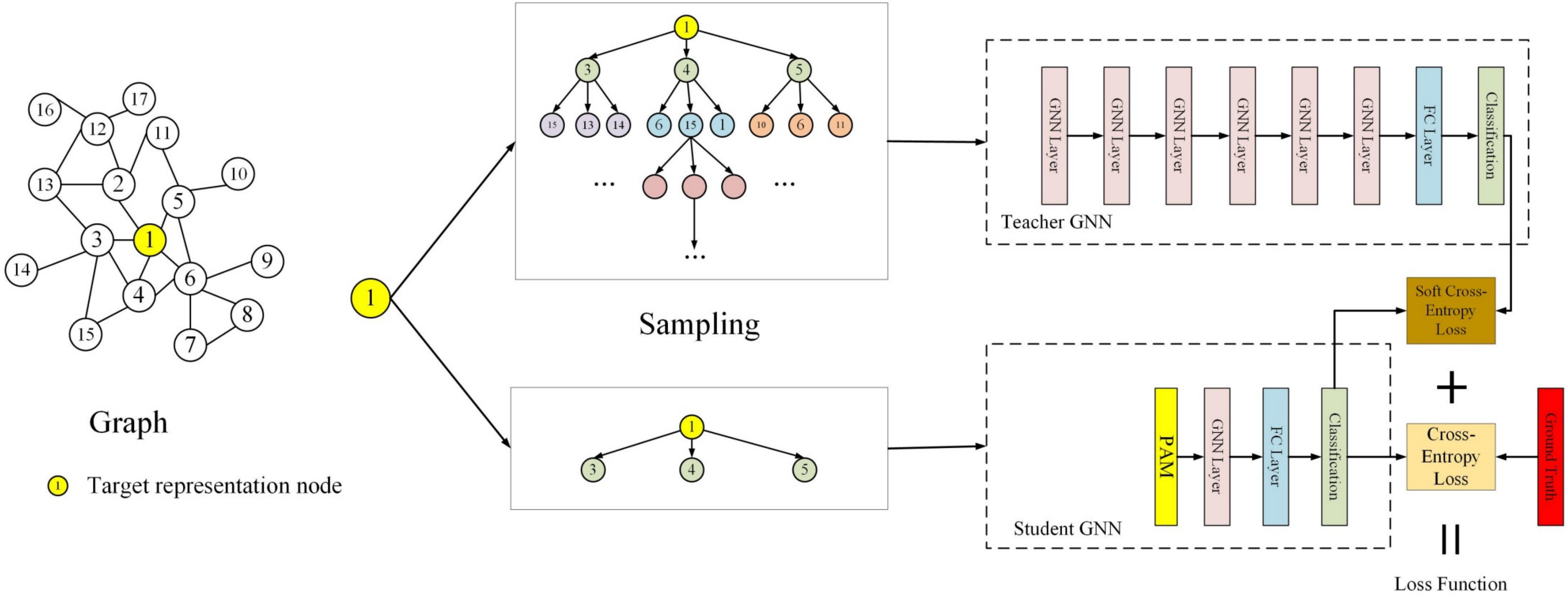


(a) Graph (b) 2-layer GNN (c) GNN Layer (d) 1-layer GNN with PAM

Peer-Aware Module captures the local structure

Knowledge Distillation from Logits

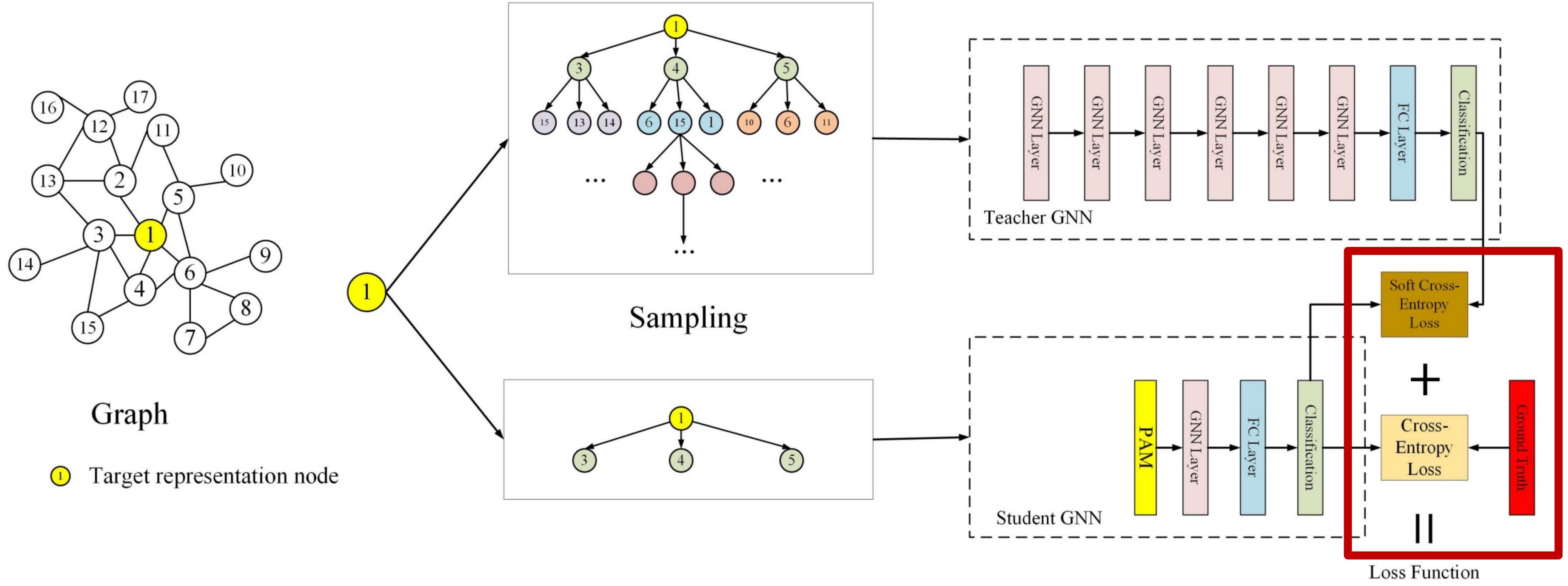
Learn a small GNN with local structure enhanced: TinyGNN



- Nodes 3, 4, 5 are peer nodes to each other
- Calculate attention between peer nodes to encode local structure

Knowledge Distillation from Logits

Learn a small GNN with local structure enhanced: TinyGNN



Learning knowledge from teacher logits and ground truth labels

Knowledge Distillation from Logits

Learn a small GNN with local structure enhanced: TinyGNN

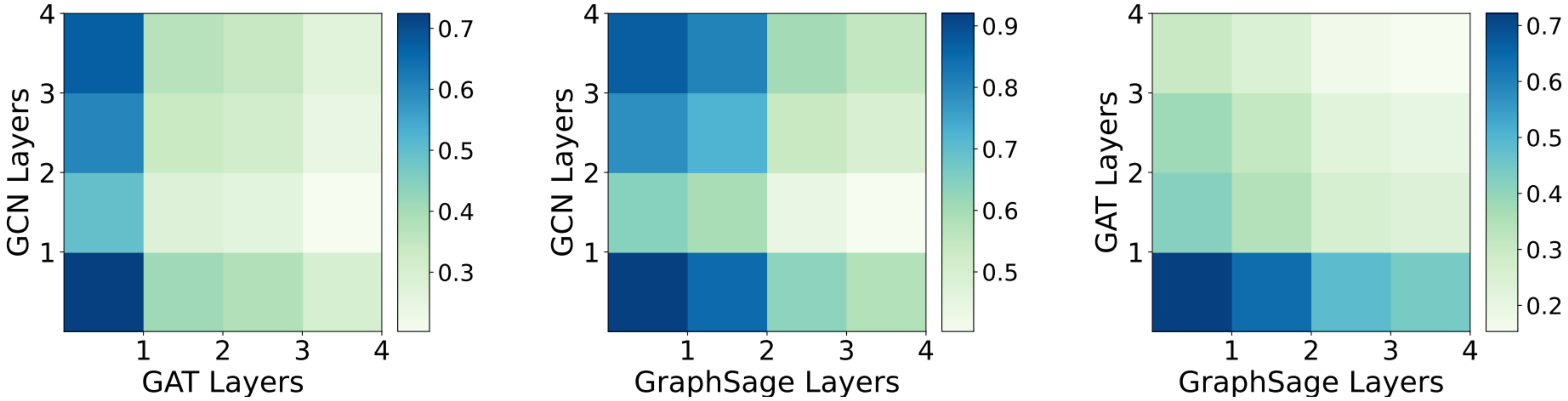
Model	Facebook		Chameleon		Squirrel		AliGraph	
	Micro-f1(%)	Macro-f1(%)	Micro-f1(%)	Macro-f1(%)	Micro-f1(%)	Macro-f1(%)	Micro-f1(%)	Macro-f1(%)
GNN_3 (Teacher)	89.43	88.69	39.12	38.16	30.68	30.41	60.34	53.75
GNN_2 (Base)	87.69	86.83	38.73	36.92	30.47	30.08	49.98	43.45
GNN_1 (Base)	82.75	81.65	36.88	35.79	29.83	29.24	33.77	25.22
GNN_1-*NDS*	83.32	82.11	37.46	36.42	30.73	30.38	35.30	24.88
GNN_1-*PAM*	83.15	81.79	38.05	36.56	31.32	31.08	40.19	32.05
$TinyGNN_1$	**84.56**	**83.41**	**39.71**	**38.46**	**32.09**	**32.00**	**42.34**	**32.37**
GNN_2-*NDS*	88.90	88.15	39.51	38.21	30.98	30.52	54.72	46.97
GNN_2-*PAM*	88.56	87.72	40.98	39.62	32.69	31.95	57.81	50.40
$TinyGNN_2$	**89.40**	**88.66**	**41.17**	**40.01**	**33.33**	**33.17**	**61.12**	**54.17**

- 1 layer TinyGNN performs poorly
- 2 layers TinyGNN can outperform the 3 layers GNN teacher

TinyGNN: Learning Efficient Graph Neural Networks

Knowledge Distillation from Logits

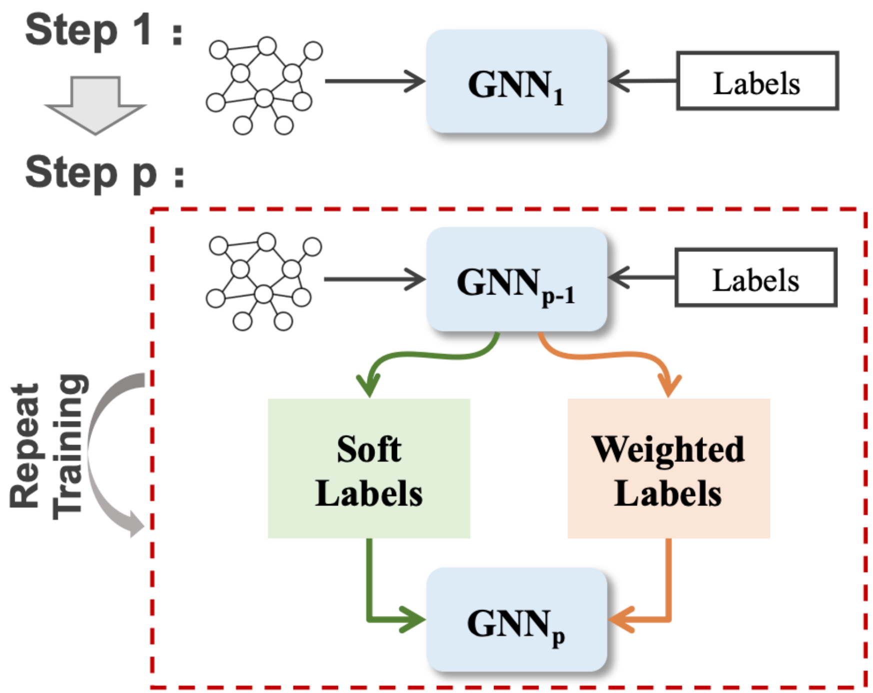
Different GNNs learn different knowledge



- GCN aggregates neighborhoods with predefined weights
- GAT aggregates neighborhoods using learnable weights
- GraphSage randomly samples neighbors during aggregation

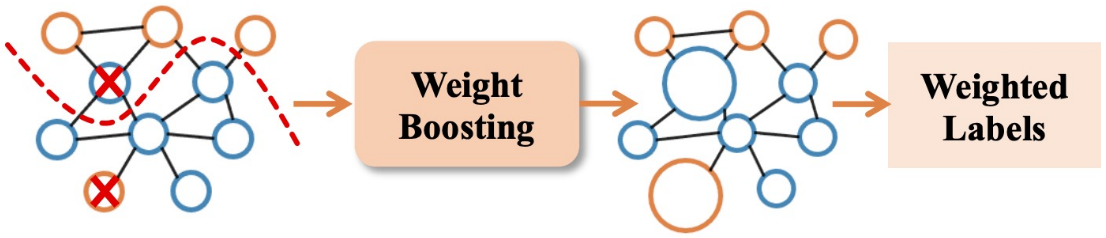
Knowledge Distillation from Logits

Learn a GNN from multiple GNNs: BGNN



* GNN_p is different from any GNN_q (q<p)

- Distill several times, once a different GNN
- Weighted Labels encourage student to learn adaptively, by focusing more on the misclassified samples



Knowledge Distillation from Logits

Learn a GNN from multiple GNNs: BGNN

		Graph Classification			Node Classification			
Student	Method	COLLAB	IMDB	ENZYMES	CORA	CITESEER	PUBMED	A-COMPUTERS
GCN	BAN (Furlanello et al. 2018)	81.60±0.40	78.50±1.00	66.67±1.67	83.17±0.26	72.47±0.21	79.87±0.21	88.78±0.05
	MulDE (Wang et al. 2021)	80.86±1.18	77.50±1.20	67.33±0.90	82.53±0.05	72.33±0.09	78.73±0.09	88.41±0.12
	BGNN(s)	82.73±0.34	79.33±0.47	69.44±0.79	83.97±0.17	73.87±0.24	80.73±0.25	**89.58±0.03**
	BGNN(m)-ST	82.87±0.09	**79.67±0.27**	**71.12±2.45**	**84.83±0.25**	73.60±0.14	80.20±0.08	89.03±0.02
	BGNN(m)-TS	**83.40±0.15**	79.00±0.00	70.00±1.36	84.40±0.22	**74.87±0.25**	**80.90±0.00**	89.02±0.03

BGNN outperforms existing methods on multi-teacher KD

Covered Topics for Knowledge from Models

How to obtain knowledge from

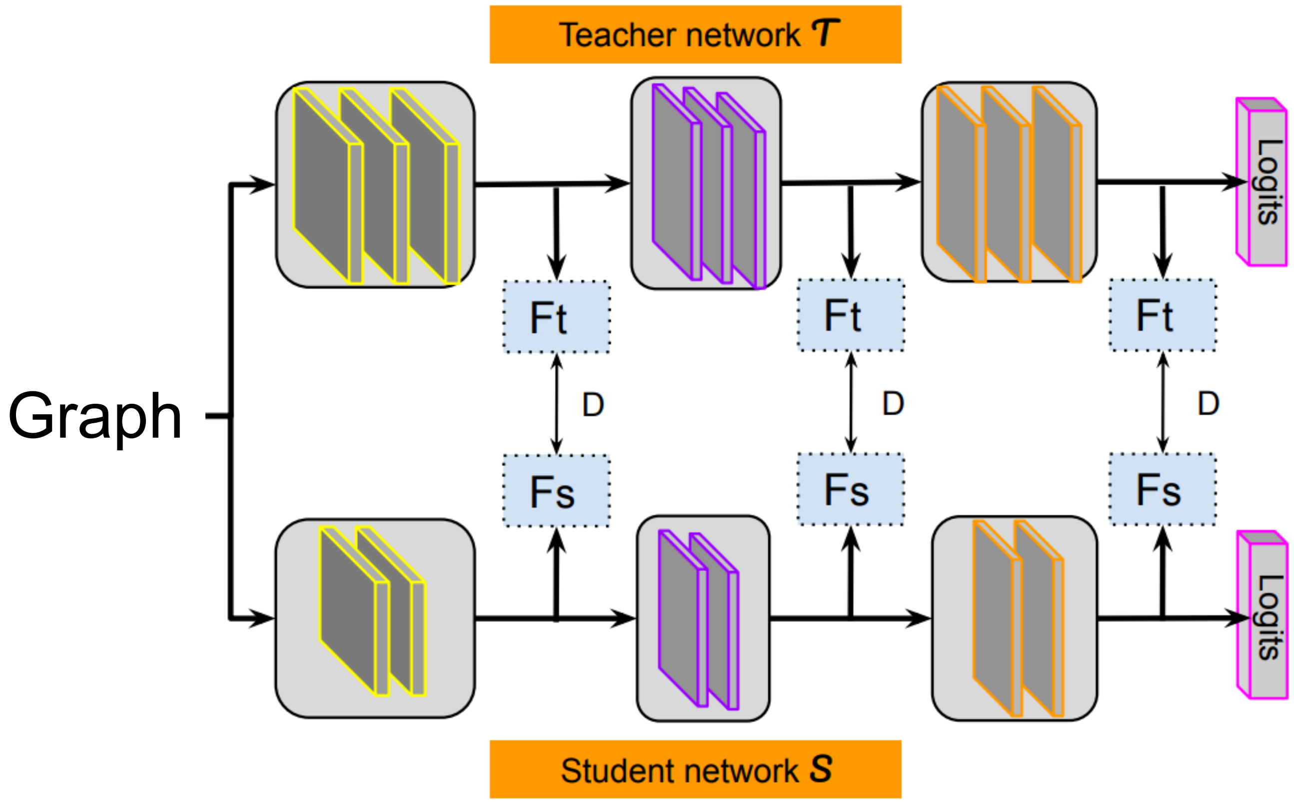
1) Logits
2) Embeddings
3) Structures

A recent learning strategy

4) distilling an MLP to replace GNNs

Knowledge Distillation from Embeddings

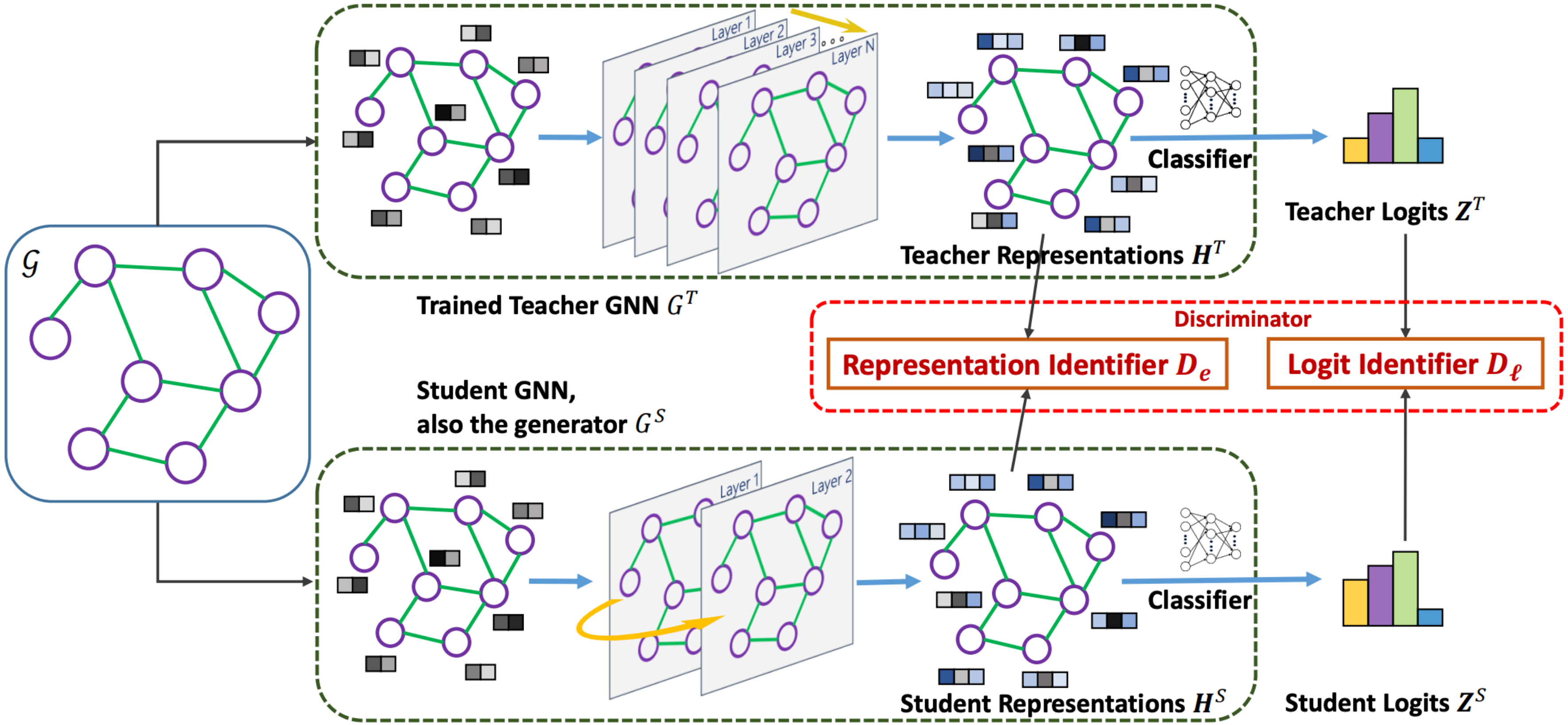
Illustration



- Beyond learning teacher’s logits, the student should also learn teacher’s embeddings
- Logits contain inter-class correlations
- Embeddings contain inter-node correlations

Knowledge Distillation from Embeddings

Enable the student to learn embeddings: GraphAKD



- GNN Generator: produce similar embeddings and logits
- Discriminator: distinguish the output of teacher and student

Knowledge Distillation from Embeddings

Enable the student to learn embeddings: GraphAKD

	Teacher			Vanilla Student			Student trained with GraphAKD			
Datasets	Model	Perf.	#Params	Model	O. Perf.	R. Perf.	Perf.	#Params	Perf. Impv. (%)	#Params Decr.
Cora	GCNII	85.5	616,519	GCN	81.5	78.3 ±0.9	83.6 ±0.8	96,633	2.1	84.3%
CiteSeer	GCNII	73.4	5,144,070	GCN	71.1	68.6 ±1.1	72.9 ±0.4	1,016.156	1.8	80.2%
PubMed	GCNII	80.3	1,177,603	GCN	79.0	78.1 ±1.0	81.3 ±0.4	195,357	2.3	83.4%
Flickr	GCNII	56.20	1,182,727	GCN	49.20	49.63 ±1.19	52.95 ±0.24	196,473	3.32	83.4%
Arxiv	GCNII	72.74	2,148,648	GCN	71.74	71.43 ±0.13	73.05 ±0.22	242,426	1.31	88.7%
Reddit	GCNII	96.77	691,241	GCN	93.30	94.12 ±0.04	95.15 ±0.02	234,655	1.03	66.1%
Yelp	GCNII	65.14	2,306,660	Cluster-GCN	59.15	59.63 ±0.51	60.63 ±0.42	431,950	1.00	81.3%
Products	GAMLP	84.59	3,335,831	Cluster-GCN	76.21	74.99 ±0.76	81.45 ±0.47	682,449	5.24	79.5%

- Better performance than vanilla student, but perform poorly than the teacher
- Smaller model size (parameters decreased by ~80%)

Knowledge Distillation from Embeddings

Enable the student to learn embeddings: GraphAKD

	#Params		GPU Memory		Inference time	
Datasets	Teacher	Student	Teacher	Student	Teacher	Student
Cora	0.6M	0.1M	0.22G	0.03G	40.3ms	4.1ms
PubMed	1.2M	0.2M	1.23G	0.33G	57.3ms	5.7ms
Flickr	1.2M	0.2M	2.79G	1.49G	309.7ms	11.9ms
Yelp	2.3M	0.4M	6.28G	4.73G	3.0s	1.5s
Products	3.3M	0.7M	6.25G	6.20G	16.1s	7.0s

Reduced GPU memory; Faster inference time

Knowledge Distillation from Embeddings

Enable the student to learn embeddings: GraphAKD

Ablation studies on the learning impacts

	Datasets	Cora	PubMed	Flickr	Yelp	Products	Molhiv
	Teacher	85.5	80.3	56.20	65.14	84.59	84.03
	Student	81.5	79.0	49.20	59.15	76.21	75.58
Embeddings	Only D_e	82.9	80.6	52.20	59.63	81.13	78.28
Logits	Only D_ℓ	82.3	81.0	52.52	60.03	79.76	78.09
	GraphAKD	**83.6**	**81.3**	**52.95**	**60.63**	**81.45**	**79.16**

- Learning from embeddings is beneficial
- Combining the learning from embeddings and logits is even better

Covered Topics for Knowledge from Models

How to obtain knowledge from

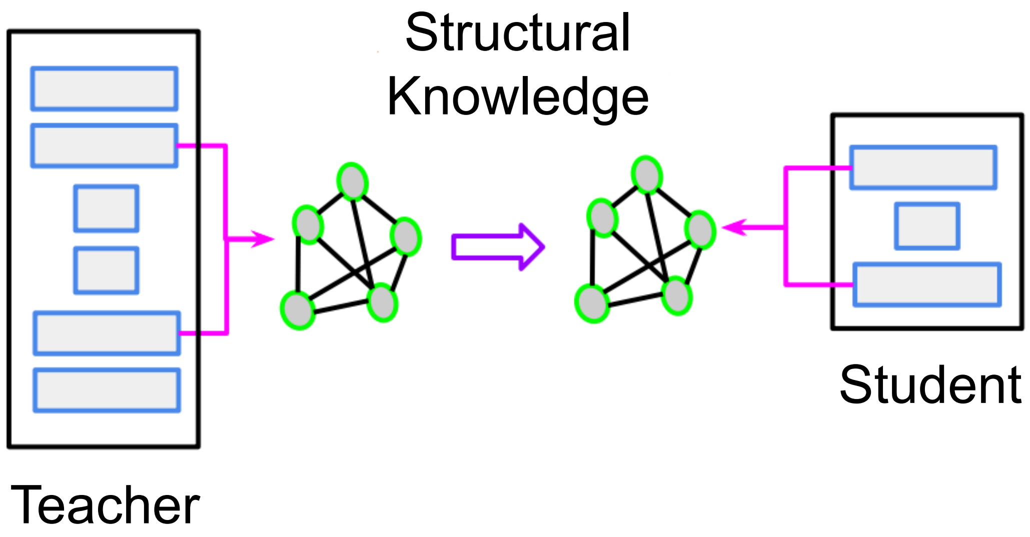
1) Logits
2) Embeddings
3) Structures

A recent learning strategy

4) distilling an MLP to replace GNNs

Knowledge Distillation from Structures

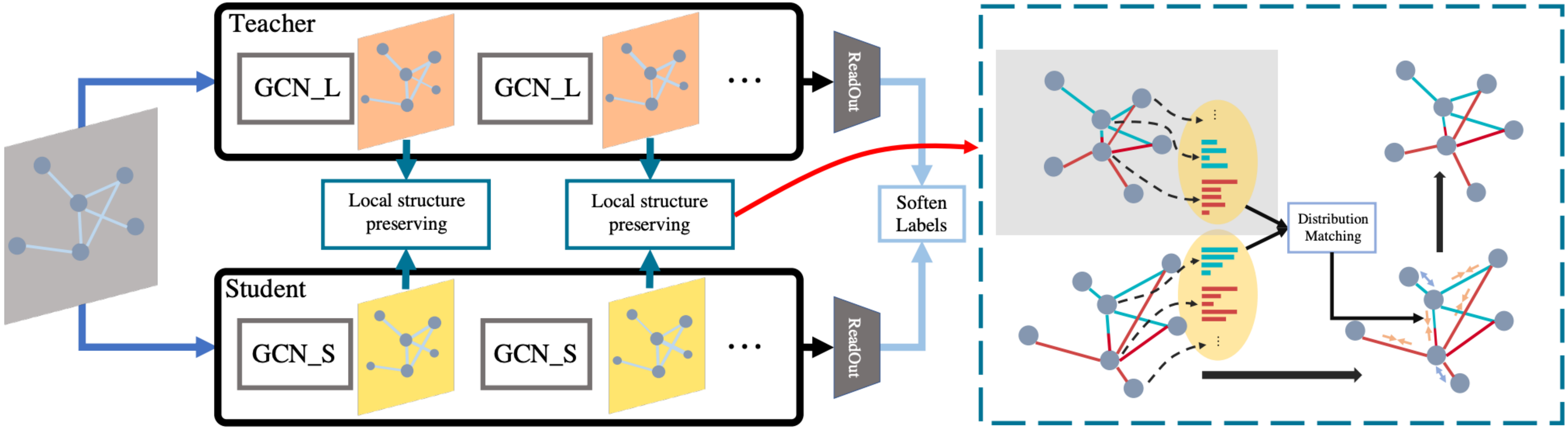
Illustration



The student should also learn graph structural and topology information from the teacher

Knowledge Distillation from Structures

Enable the student to capture the local structure: LSP



- 1st step: compute the distribution of the local structure for each node (denoted by node similarity to each other)
- 2nd step: match the distributions of the teacher with that of the student

Knowledge Distillation from Structures

Enable the student to capture the local structure: LSP

Model details: GAT

Model	Layers	Attention heads	Hidden features
Teacher	3	4,4,6	256,256,121
Student	5	2,2,2,2,2	68,68,68,68,121

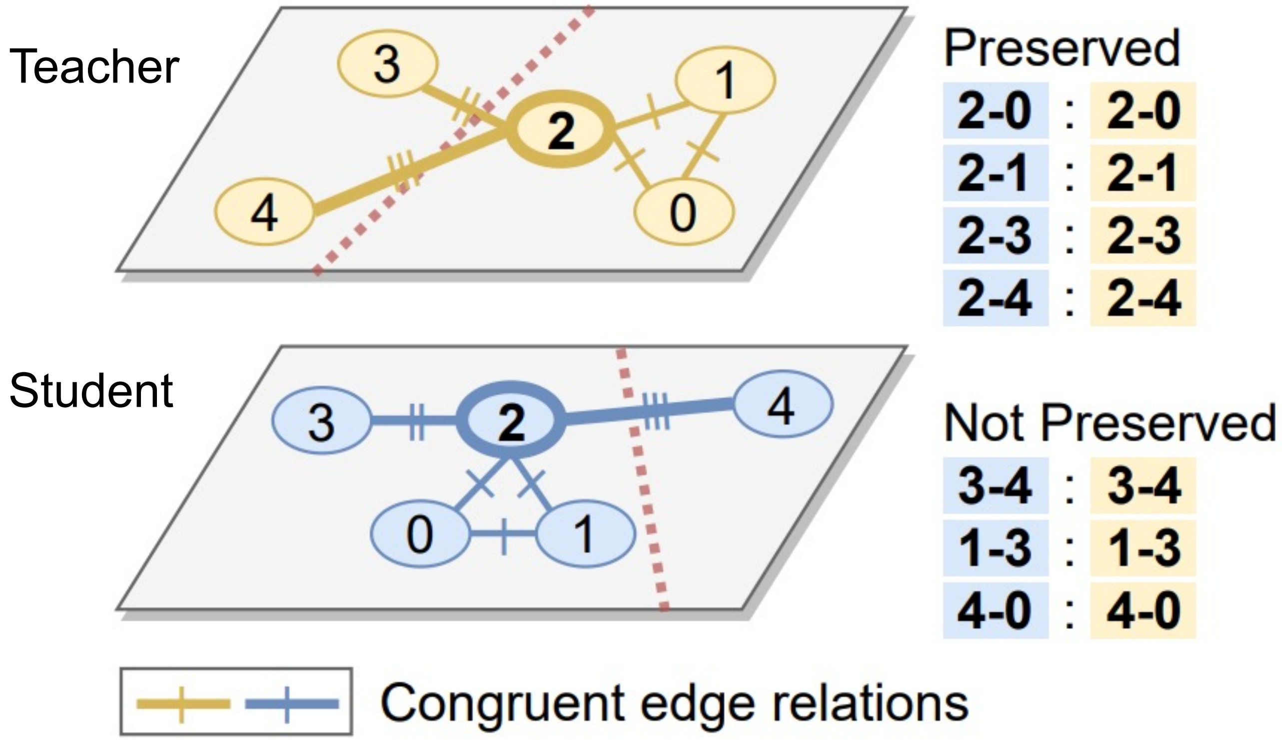
Node classification on PPI

Model	Params	RunTime	Training	F1 Score
Teacher	3.64M	48.5ms	1.7s/3.4G	97.6
Student_Full	0.16M	41.3ms	1.3s/1.2G	95.7
Student_KD [14]	-	-	-	-
Student_AT [47]	0.16M	41.3ms	1.9s/1.4G	95.4
Student_FitNet [31]	0.16M	41.3ms	2.4s/1.6G	95.6
Student_LSP (Ours)	0.16M	41.3ms	2.0s/1.5G	**96.1**

Reduced parameters; Comparable F1 to the teacher

Knowledge Distillation from Structures

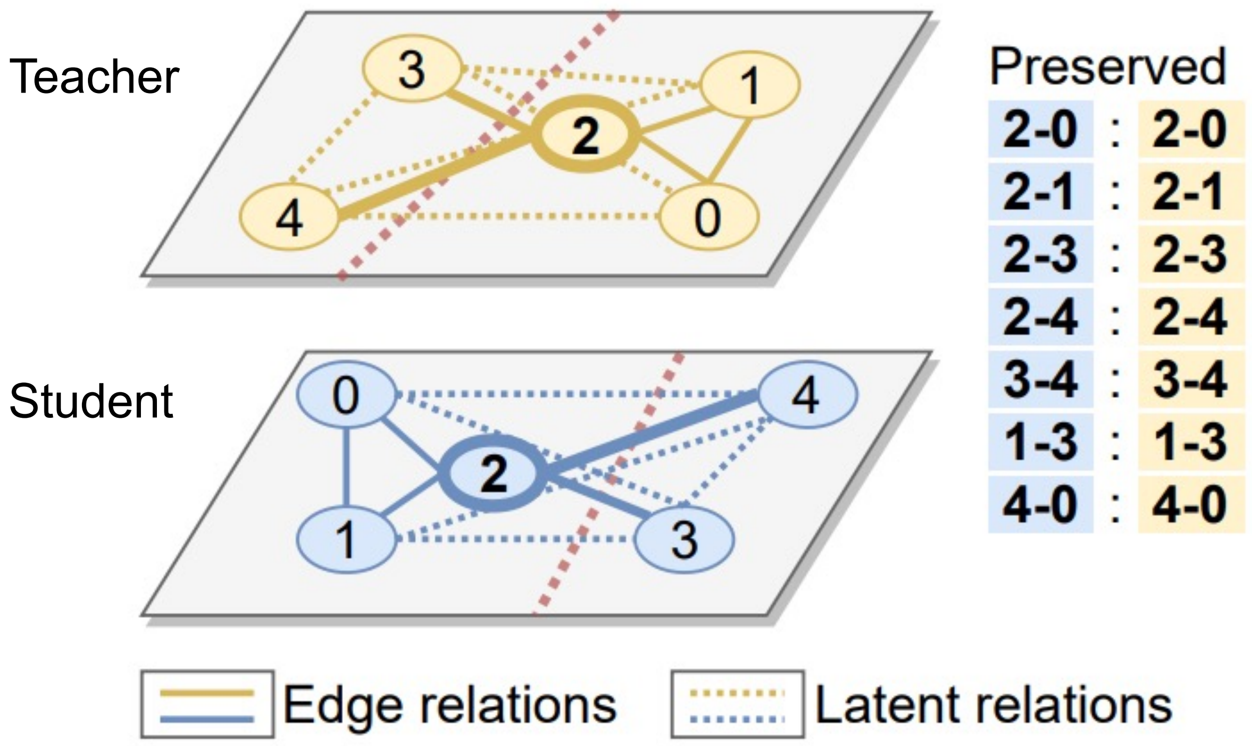
The strategy of LSP



Considering pairwise relationships, but not considering latent interactions among disconnected nodes.

Knowledge Distillation from Structures

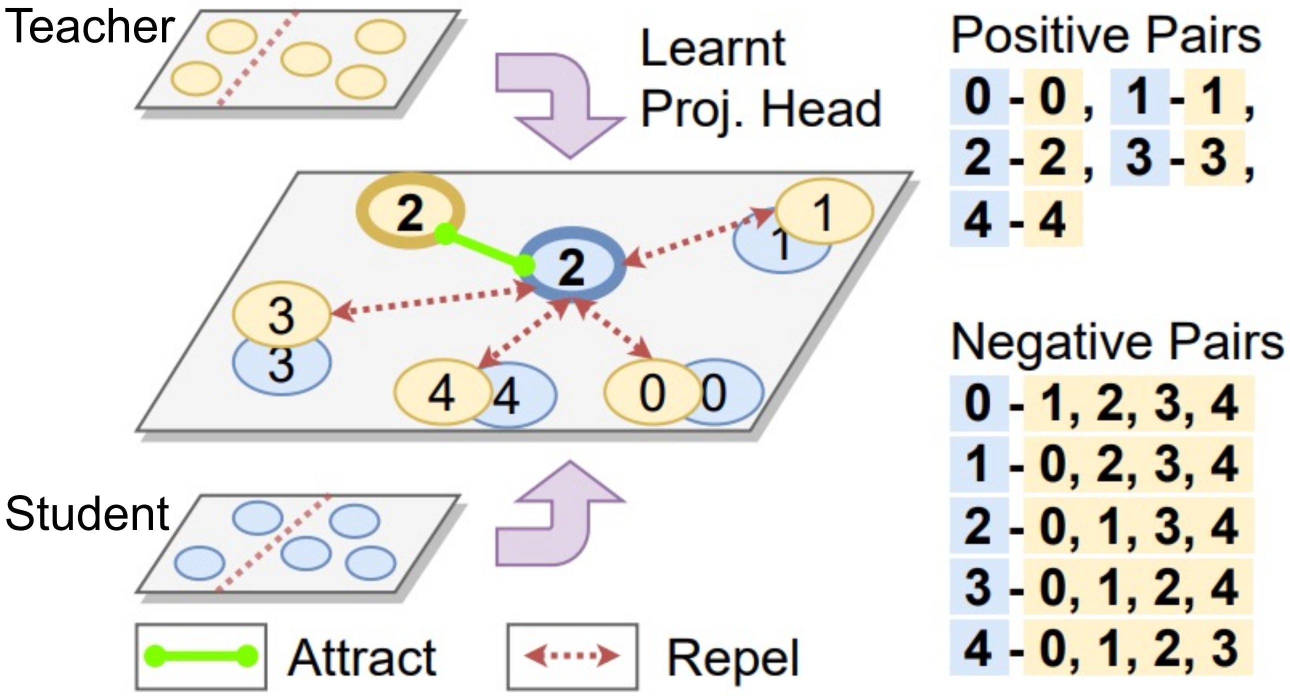
One step further: enhance the student with global structure



- Global Structure Preserving (GSP): consider all possible pairwise similarities among node features
- Challenging to optimize/scale up due to an explosion of possible pairs

Knowledge Distillation from Structures

Enhance the student with implicit global structure: G-CRD



- Use contrastive learning to capture implicit global topology
- Positive pair: same node embeddings of the teacher and the student
- Negative pair: random nodes

Knowledge Distillation from Structures

Enhance the student with implicit global structure: G-CRD

Molecular graph classification on MOLHIV

	Teacher (#Layer,#Param): Student (#Layer,#Param):	**GIN-E** (5L,3.3M) **GCN** (2L,15K)	**PNA** (5L,2.4M) **GCN** (2L,15K)	**GIN-E** (5L,3.3M) **GCN** (2L,40K)	**PNA** (5L,2.4M) **GCN** (2L,40K)	**PNA** (5L,2.4M) **GIN** (2L,10K)
Sup.	Supervised Teacher	77.69 ±1.61	77.48 ±1.71	77.69 ±1.61	77.48 ±1.71	77.48 ±1.71
	Supervised Student	73.02 ±1.46	73.02 ±1.46	73.65 ±1.50	73.65 ±1.50	73.03 ±2.02
Distillation	KD [30]	<u>74.08</u> ±1.03	<u>74.13</u> ±1.72	<u>75.25</u> ±1.71	74.45 ±1.27	73.42 ±2.14
	FitNet [47]	73.62 ±1.05 (↓)	73.65 ±1.25 (↓)	74.52 ±1.33 (↓)	74.39 ±1.46 (↓)	72.88 ±0.89 (↓)
	AT [48]	73.85 ±0.85 (↓)	73.64 ±1.50 (↓)	74.94 ±0.97 (↓)	73.89 ±1.92 (↓)	73.87 ±2.28 (↑)
	LSP [32]	73.58 ±1.29 (↓)	73.24 ±1.67 (↓)	75.04 ±1.20 (↓)	74.43 ±1.58 (↓)	70.74 ±1.82 (↓)
	GSP	72.83 ±1.30 (↓)	73.74 ±0.93 (↓)	75.12 ±1.27 (↓)	<u>75.09</u> ±1.48 (↑)	69.68 ±2.88 (↓)
	G-CRD (Ours)	**74.34** ±1.44 (↑)	**75.11** ±0.73 (↑)	**75.53** ±1.64 (↑)	**75.89** ±0.80 (↑)	**75.77** ±2.02 (↑)

- LSP performs poorly: preserving local structure is not sufficient
- GSP performs poorly: preserving every possible pair of nodes contributes less
- G-CRD performs well: preserving the implicit global structure is beneficial

Covered Topics for Knowledge from Models

How to obtain knowledge from

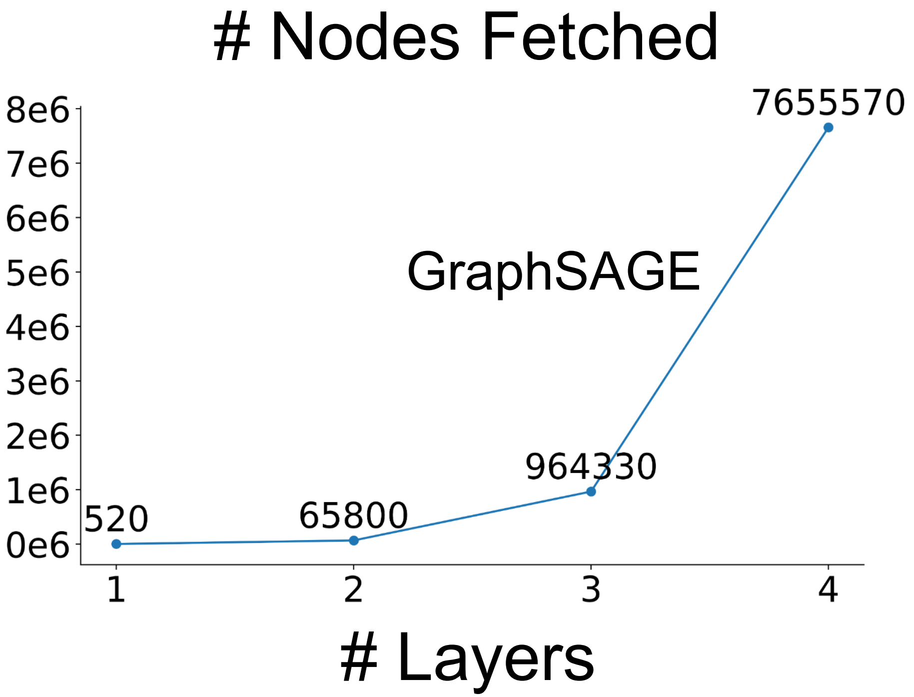
1) Logits
2) Embeddings
3) Structures

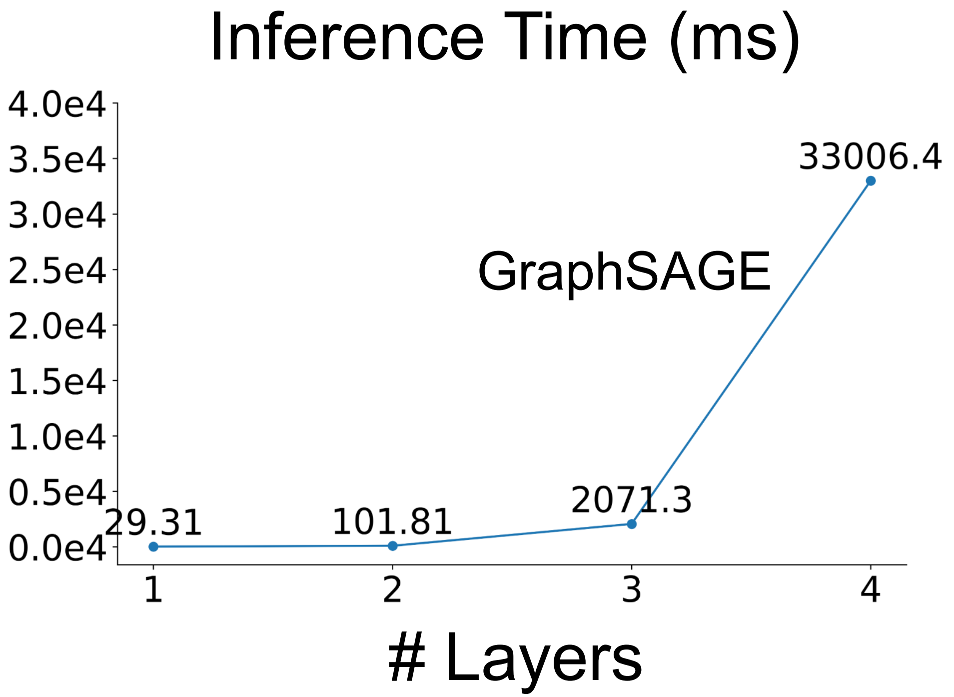
A recent learning strategy

4) distilling an MLP to replace GNNs

Distilling an MLP to Replace GNNs

The majority of KD on Graphs focus on GNN to GNN

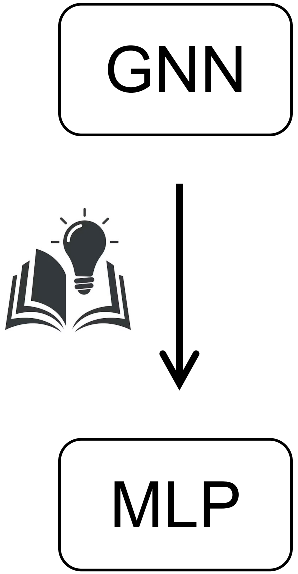


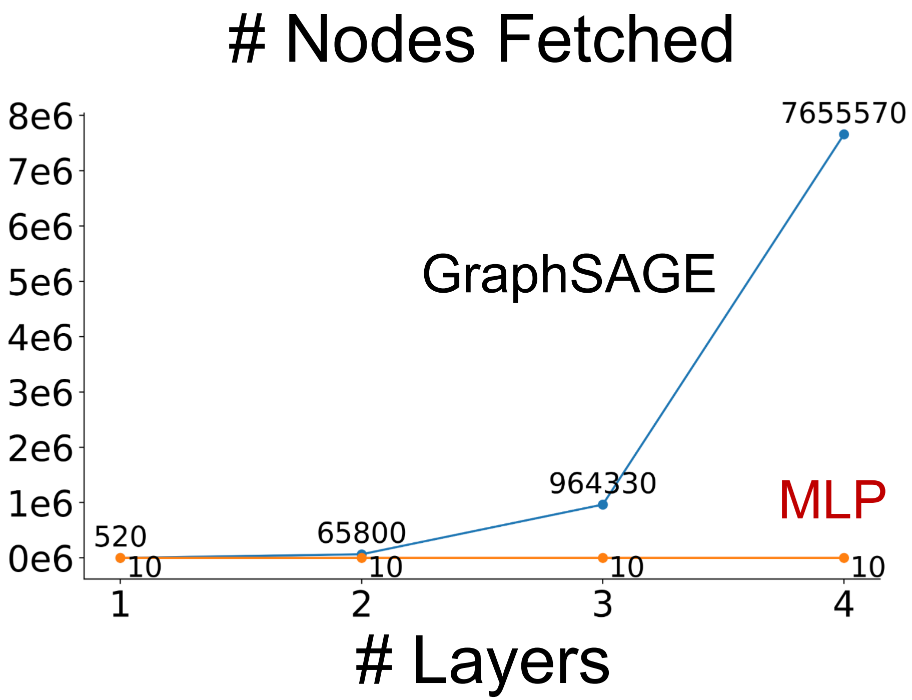


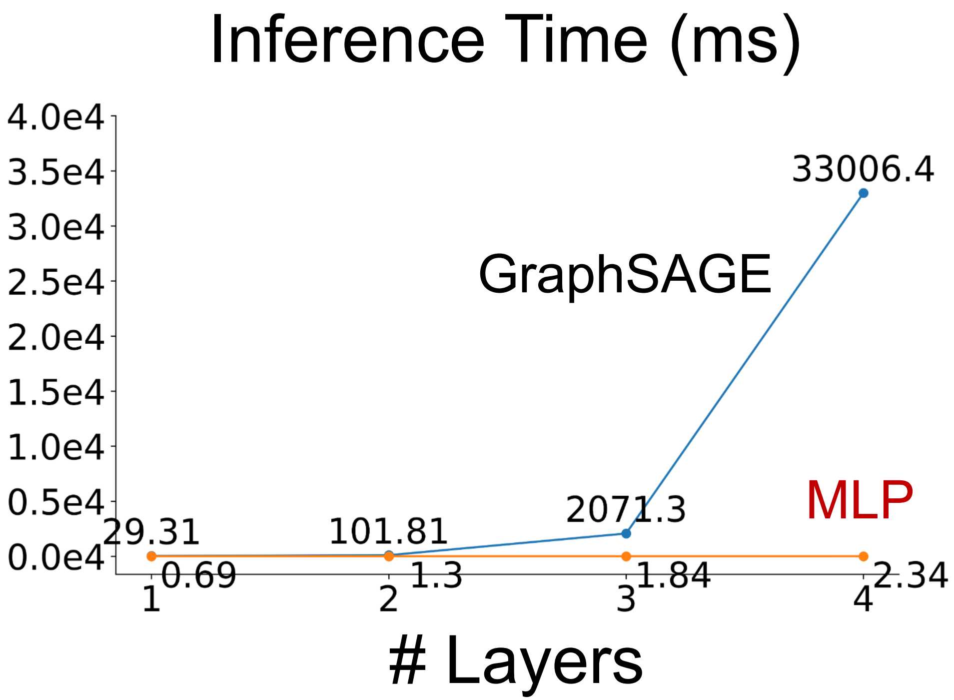
Inherent limitation: Dependence on message passing architecture, which is time-consuming and computation-intensive, making the model inapplicable to time-sensitive situations

Distilling an MLP to Replace GNNs

To avoid the troublesome message passing



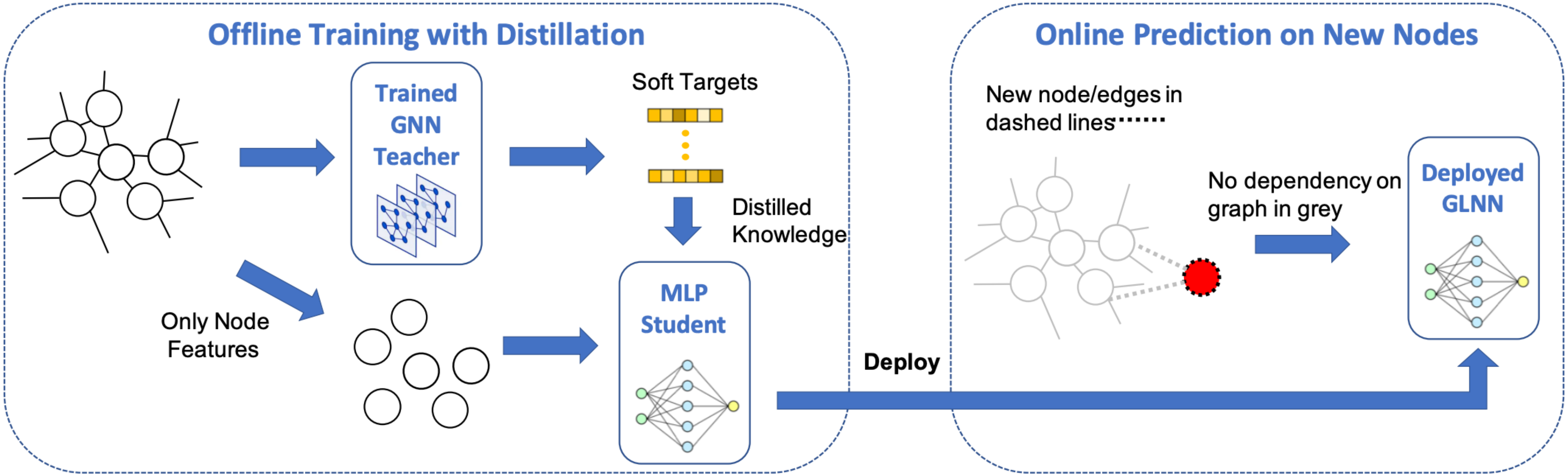




Distilling knowledge to an MLP (Multi-layer Perceptron), which is simple and does not require message passing

Distilling an MLP to Replace GNNs

Distilling knowledge to an MLP: GLNN



Distilling an MLP to Replace GNNs

Distilling knowledge to an MLP: GLNN

Datasets	SAGE	MLP	GLNN	Δ_{MLP}	Δ_{GNN}
Cora	80.52 ± 1.77	59.22 ± 1.31	$\mathbf{80.54 \pm 1.35}$	21.32 (36.00%)	0.02 (0.02%)
Citeseer	70.33 ± 1.97	59.61 ± 2.88	$\mathbf{71.77 \pm 2.01}$	12.16 (20.40%)	1.44 (2.05%)
Pubmed	75.39 ± 2.09	67.55 ± 2.31	$\mathbf{75.42 \pm 2.31}$	7.87 (11.65%)	0.03 (0.04%)
A-computer	82.97 ± 2.16	67.80 ± 1.06	$\mathbf{83.03 \pm 1.87}$	15.23 (22.46%)	0.06 (0.07%)
A-photo	90.90 ± 0.84	78.77 ± 1.74	$\mathbf{92.11 \pm 1.08}$	13.34 (16.94%)	1.21 (1.33%)
Arxiv	$\mathbf{70.92 \pm 0.17}$	56.05 ± 0.46	63.46 ± 0.45	7.41 (13.24%)	-7.46 (-10.52%)
Products	$\mathbf{78.61 \pm 0.49}$	62.47 ± 0.10	68.86 ± 0.46	6.39 (10.23%)	-9.75 (-12.4%)

GLNN performs well on small datasets,
but performs poorly on large datasets

Graph-less Neural Networks: Teaching Old MLPs New Tricks via Distillation

Distilling an MLP to Replace GNNs

Problems of simply distilling knowledge to an MLP

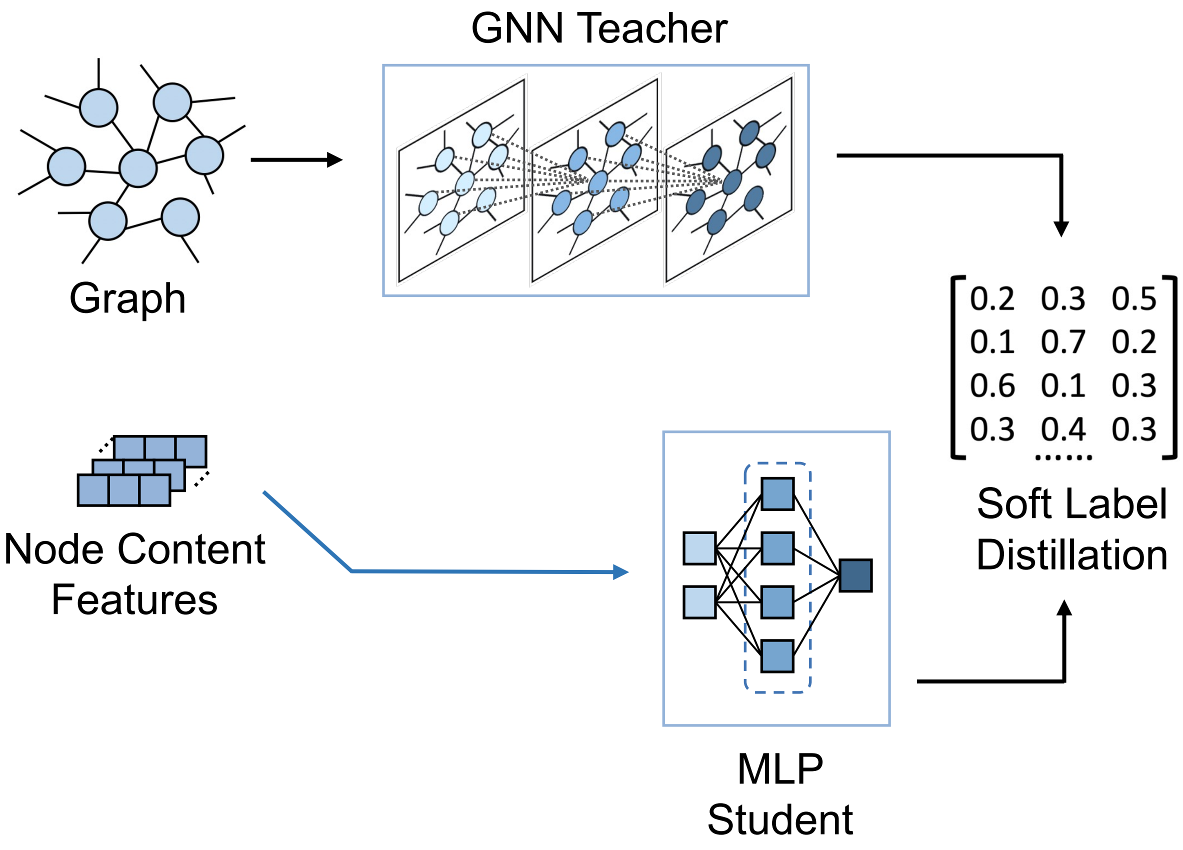
- The misalignment between content feature and label spaces
- The strict hard matching to teacher's output
- The sensitivity to node feature noises

Therefore, a question can be asked:

Can we learn MLPs that are graph structure-aware in both the feature and representation spaces, insensitive to node feature noises, and have superior performance as well as fast inference speed?

Distilling an MLP to Replace GNNs

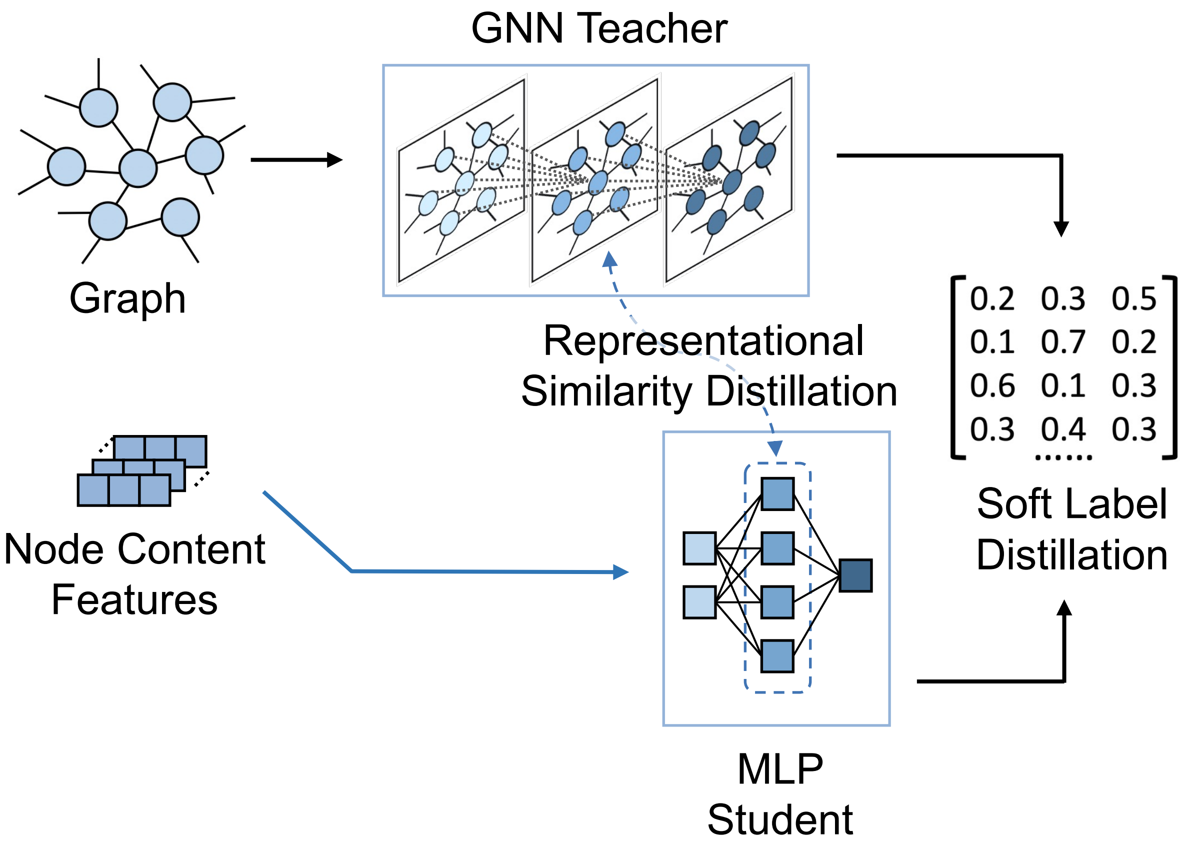
Learn a Noise-robust Structure-aware MLPs On Graphs: NOSMOG



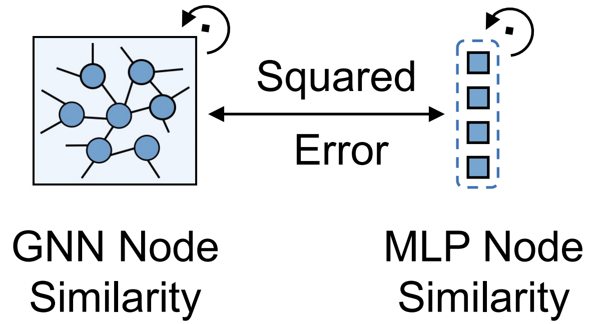
- Distill Knowledge of Logits

Distilling an MLP to Replace GNNs

Learn a Noise-robust Structure-aware MLPs On Graphs: NOSMOG

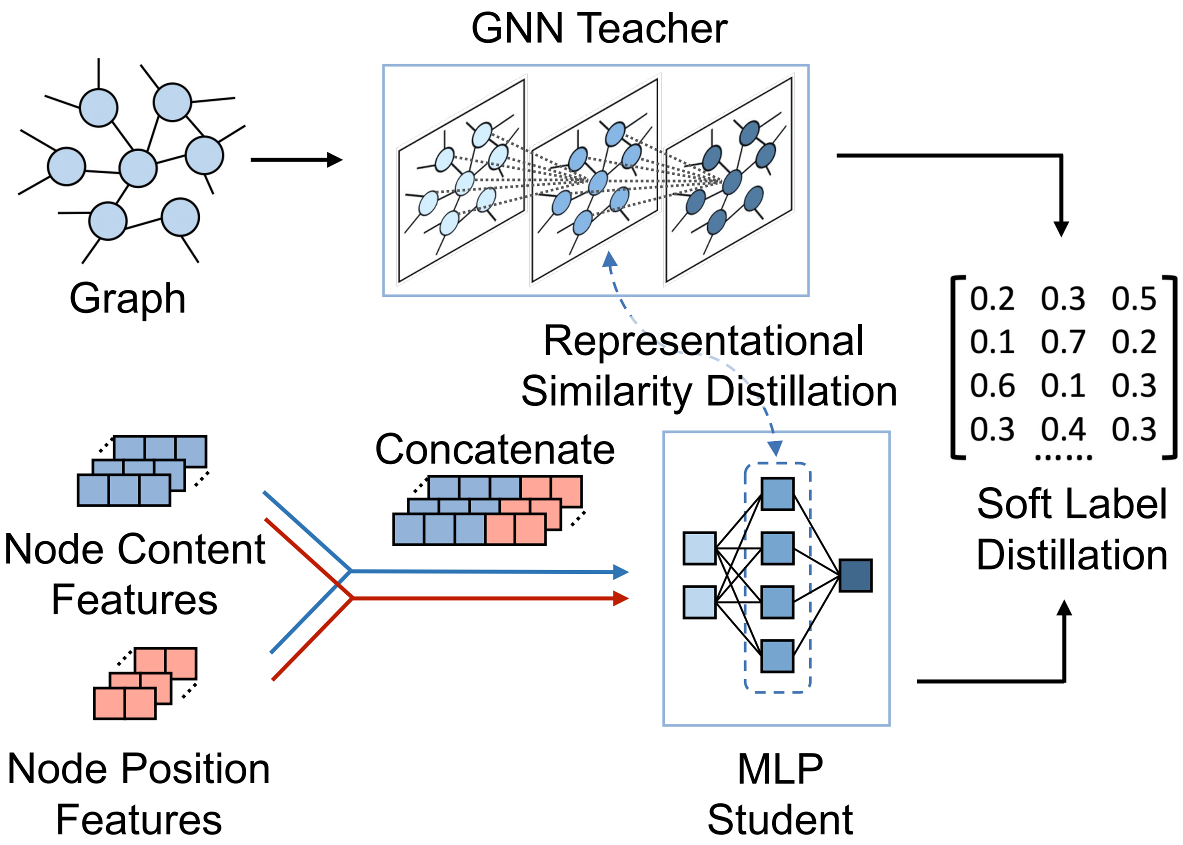


- Distill Knowledge of Logits
- Distill Knowledge of Learned Embedding and Structure



Distilling an MLP to Replace GNNs

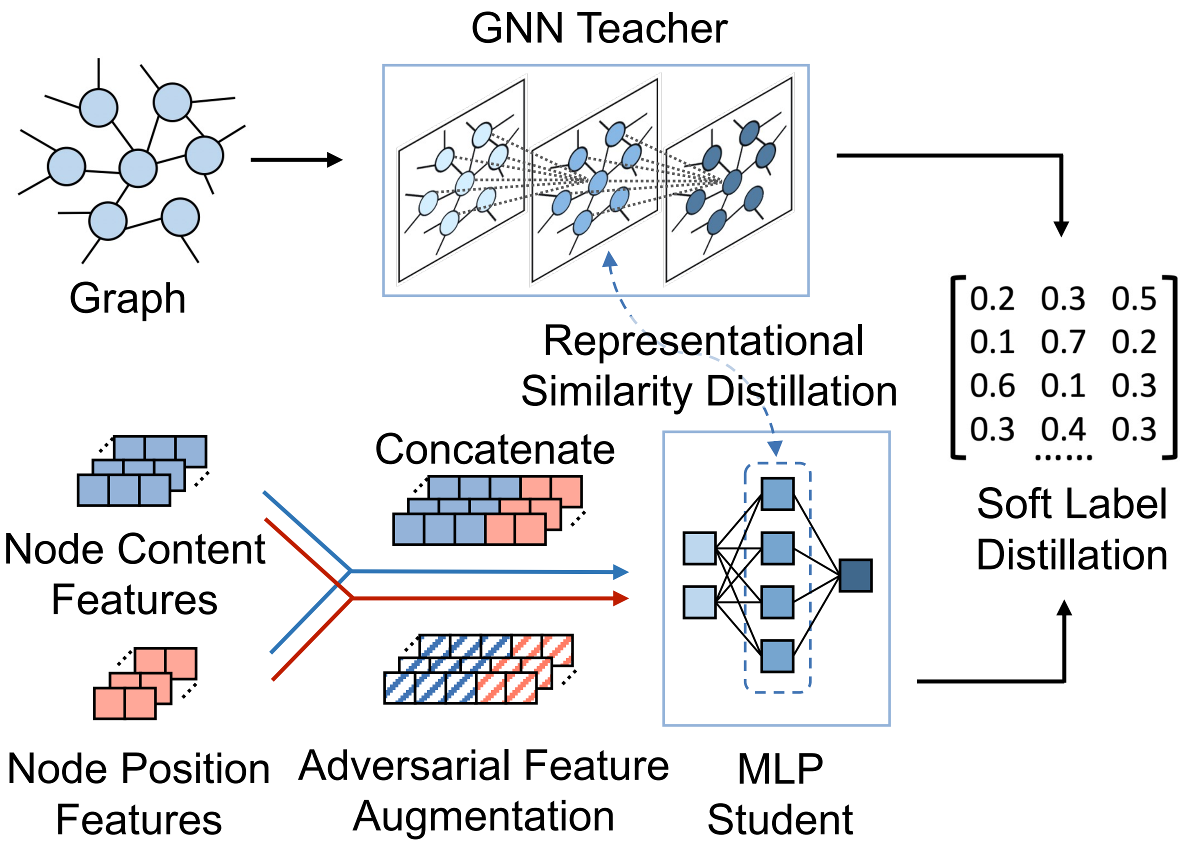
Learn a Noise-robust Structure-aware MLPs On Graphs: NOSMOG



- Distill Knowledge of Logits
- Distill Knowledge of Learned Embedding and Structure
- Learning Knowledge of Node Positions from Data

Distilling an MLP to Replace GNNs

Learn a Noise-robust Structure-aware MLPs On Graphs: NOSMOG



- Distill Knowledge of Logits
- Distill Knowledge of Learned Embedding and Structure
- Learning Knowledge of Node Positions from Data
- Enhance Knowledge Acquisition with Augmentation

Distilling an MLP to Replace GNNs

Learn a Noise-robust Structure-aware MLPs On Graphs: NOSMOG

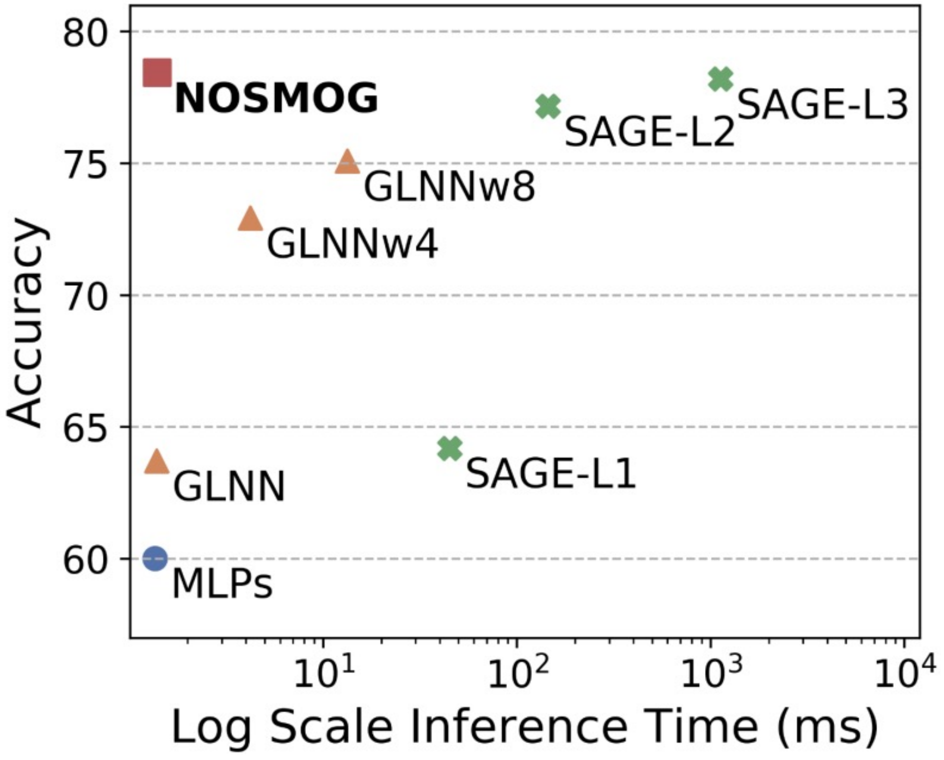
Datasets	SAGE	MLP	GLNN	NOSMOG	Δ_{GNN}	Δ_{MLP}	Δ_{GLNN}
Cora	80.64 ± 1.57	59.18 ± 1.60	80.26 ± 1.66	**83.04 ± 1.26**	$\uparrow 2.98\%$	$\uparrow 40.32\%$	$\uparrow 3.46\%$
Citeseer	70.49 ± 1.53	58.50 ± 1.86	71.22 ± 1.50	**73.78 ± 1.54**	$\uparrow 4.67\%$	$\uparrow 26.12\%$	$\uparrow 3.59\%$
Pubmed	75.56 ± 2.06	68.39 ± 3.09	75.59 ± 2.46	**77.34 ± 2.36**	$\uparrow 2.36\%$	$\uparrow 13.09\%$	$\uparrow 2.32\%$
A-computer	82.82 ± 1.37	67.62 ± 2.21	82.71 ± 1.18	**84.04 ± 1.01**	$\uparrow 1.47\%$	$\uparrow 24.28\%$	$\uparrow 1.61\%$
A-photo	90.85 ± 0.87	77.29 ± 1.79	91.95 ± 1.04	**93.36 ± 0.69**	$\uparrow 2.76\%$	$\uparrow 20.79\%$	$\uparrow 1.53\%$
Arxiv	70.73 ± 0.35	55.67 ± 0.24	63.75 ± 0.48	**71.65 ± 0.29**	$\uparrow 1.30\%$	$\uparrow 28.70\%$	$\uparrow 12.39\%$
Products	77.17 ± 0.32	60.02 ± 0.10	63.71 ± 0.31	**78.45 ± 0.38**	$\uparrow 1.66\%$	$\uparrow 30.71\%$	$\uparrow 23.14\%$
					+2.46%	+26.29%	+6.86%

NOSMOG achieves the best performance

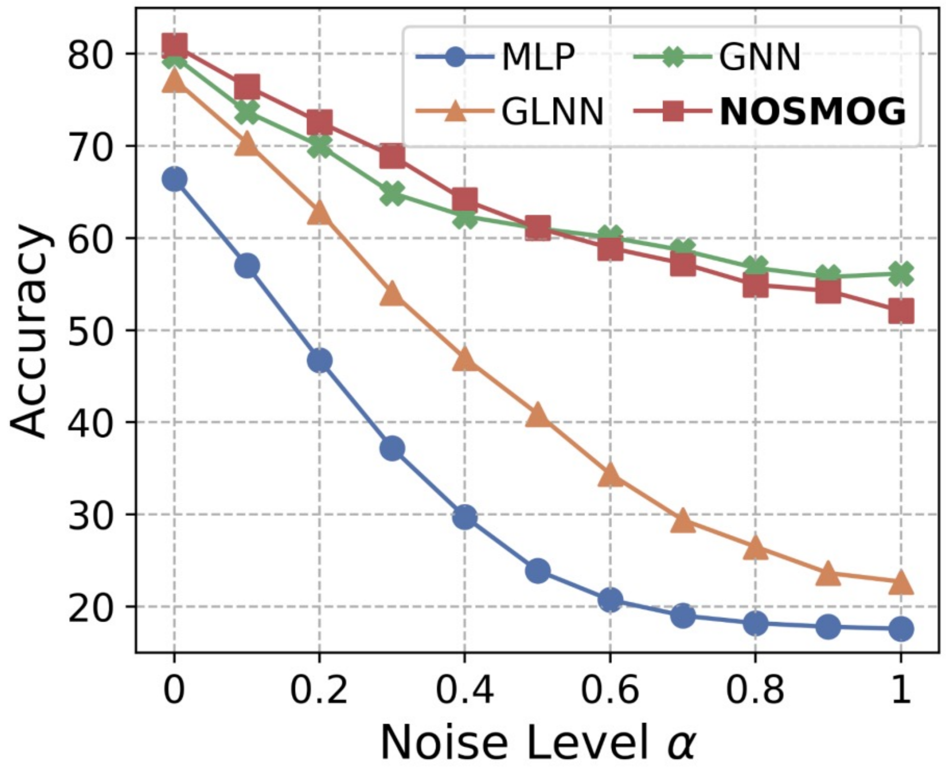
Distilling an MLP to Replace GNNs

Learn a Noise-robust Structure-aware MLPs On Graphs: NOSMOG

NOSMOG is 833x faster than GNNs



NOSMOG is as robust as GNNs



Learning MLPs on Graphs: A Unified View of Effectiveness, Robustness, and Efficiency

Graph Learning Enhanced by Knowledge from Models

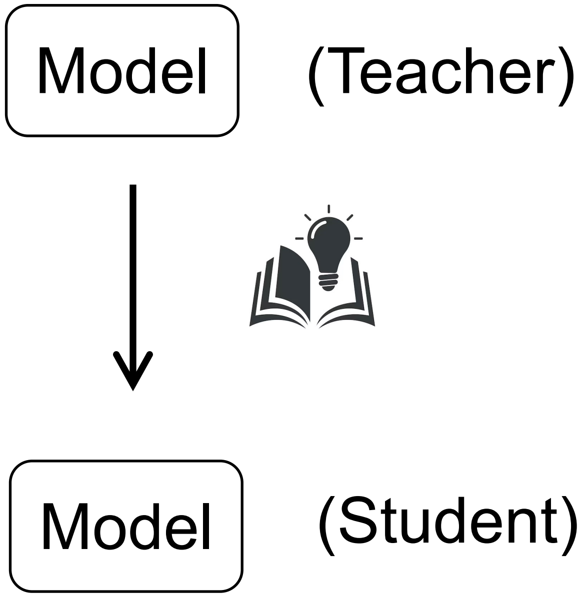
Chapter summary

How to obtain knowledge from

1) Logits
2) Embeddings
3) Structures

A recent learning strategy

4) distilling an MLP to replace GNNs



Tutorial Outline

- Preliminaries and Foundations
- Graph Learning Enhanced by Knowledge from Data
- Graph Learning Enhanced by Knowledge from Models
- Graph Learning Enhanced by Knowledge from Humans and Domains
- Graph Learning Enhanced by Knowledge from External Sources
- Knowledge-enhanced Graph Learning for Real-world Applications
- Summary and Future Directions

Topics for Knowledge from Humans and Domains

Graph learning enhanced by human feedback

- Graph active learning with human knowledge

Graph learning enhanced by domain knowledge

- Chemistry domain knowledge for molecular property prediction

Topics for Knowledge from Humans and Domains

Graph learning enhanced by human feedback

- Graph active learning with human knowledge

Graph learning enhanced by domain knowledge

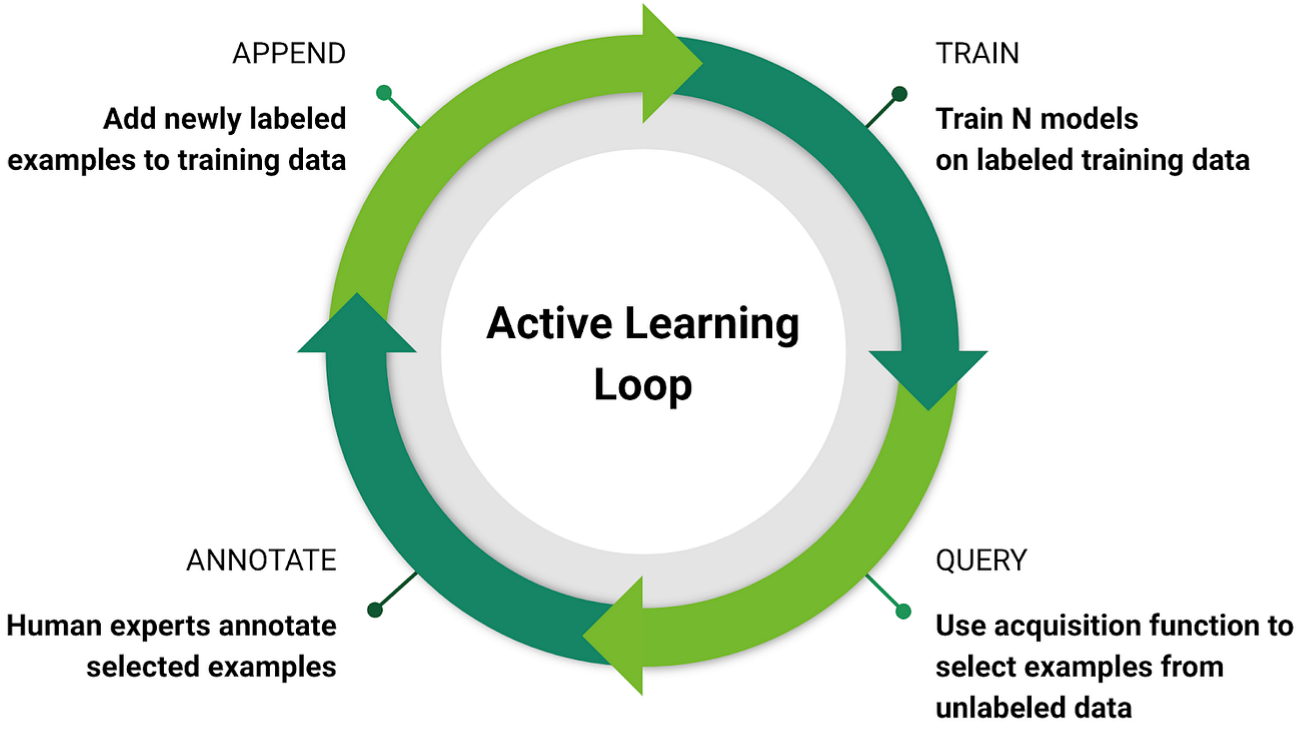
- Chemistry domain knowledge for molecular property prediction

Graph Active Learning

What is graph active learning?

Chooses graph data examples to label from a large pool of unlabeled data repeatedly:

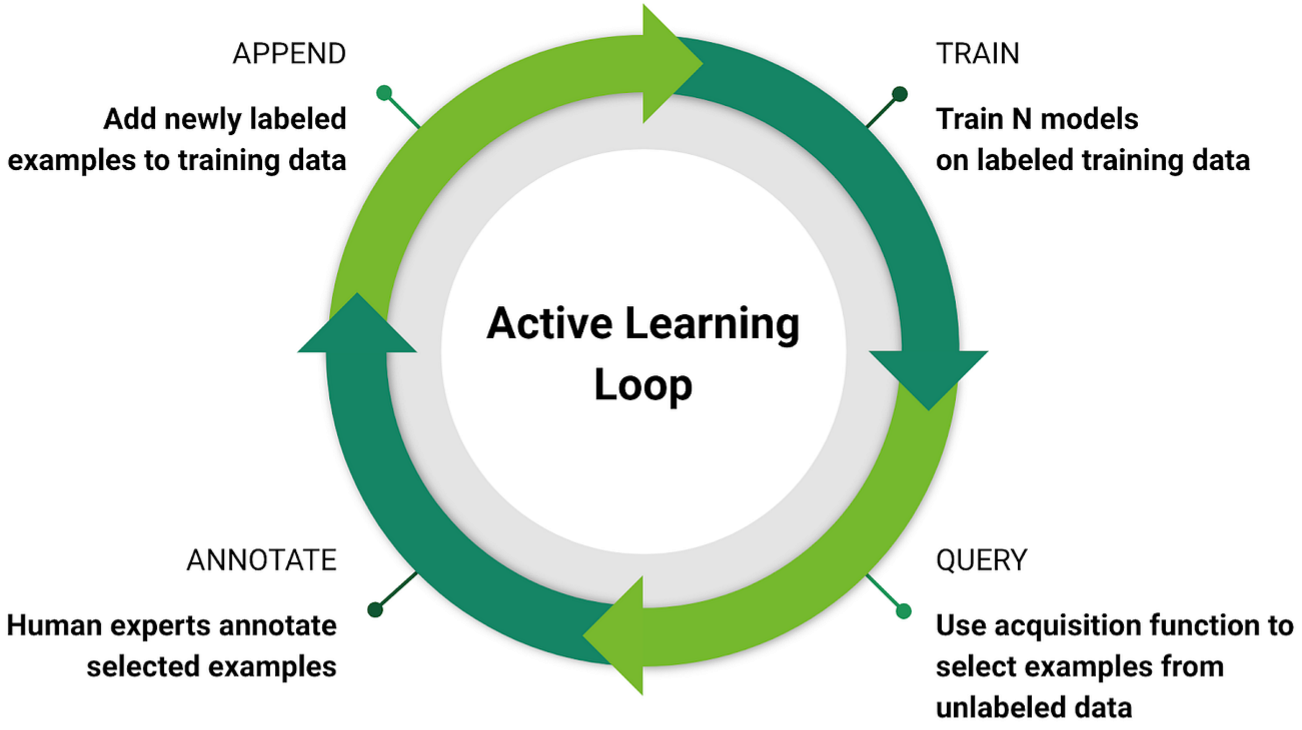
- training a model on the small pool of labeled graph data
- selecting graph data examples to label based on different query heuristics



Graph Active Learning

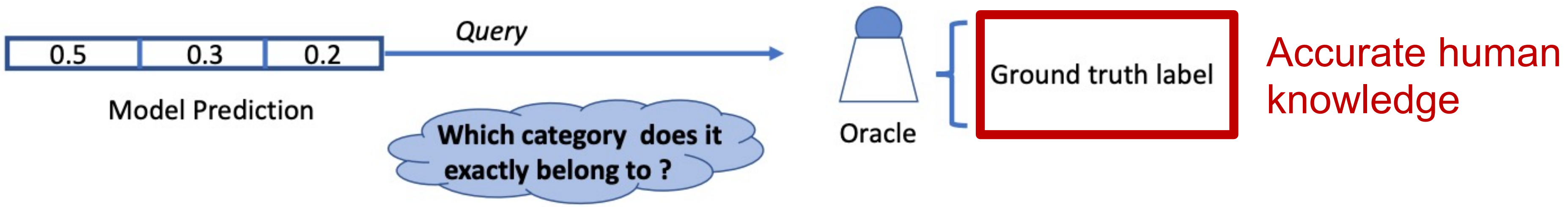
Why do we need graph active learning?

- Incorporating human knowledge
- Reduce the annotation cost by focusing on the most relevant graph data examples



Graph Active Learning

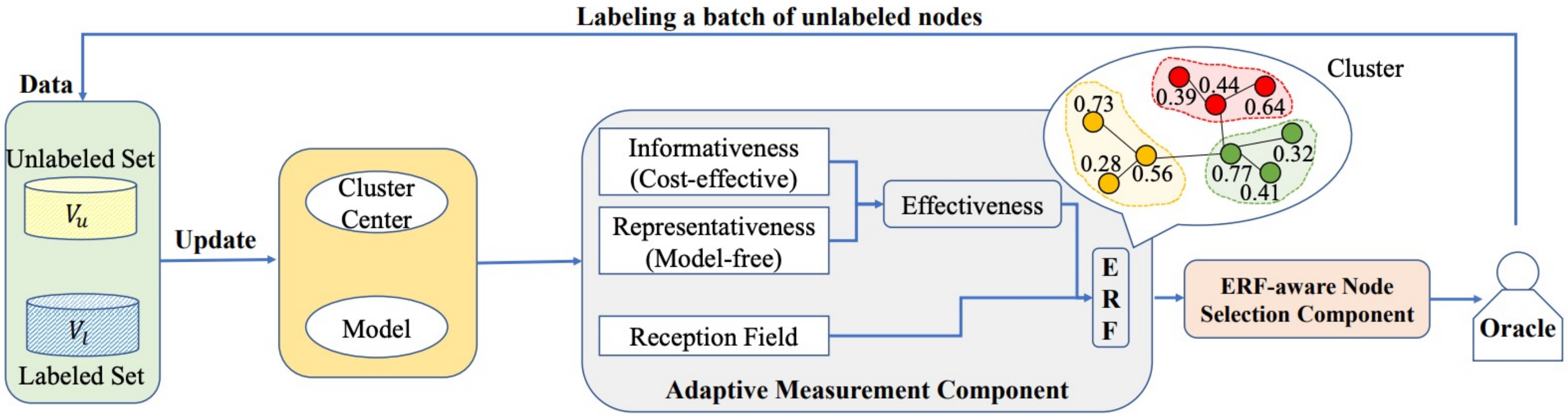
Incorporating accurate human knowledge: ALG



In this setting, ground truth labels that are one-hot encoding serve as accurate human knowledge

Graph Active Learning

Incorporating accurate human knowledge: ALG

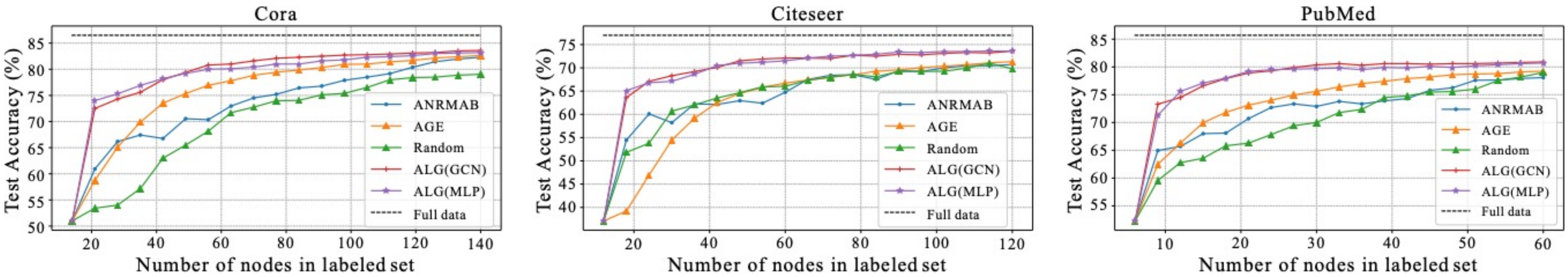


- An adaptive measurement component that computes the node importance, and a selection component to choose a set of nodes to label given their importance
- Measurement component: K-means-based. Center nodes in clusters are more representative
- Selection component: the model should select nodes from each cluster for fairness

ALG: Fast and Accurate Active Learning Framework for Graph Convolutional Networks

Graph Active Learning

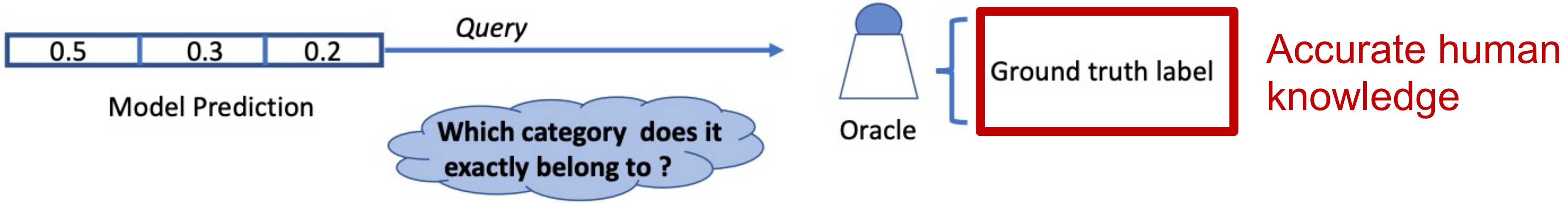
Incorporating accurate human knowledge: ALG



- When labeling more nodes, ALG quickly boosts its accuracy at the beginning and consistently outperforms the baselines
- ALG only needs to label about 5% of all the nodes in Cora to achieve the accuracy of 83.5%, which is comparable to the performance trained on the full dataset (a gap <3%)

Graph Active Learning

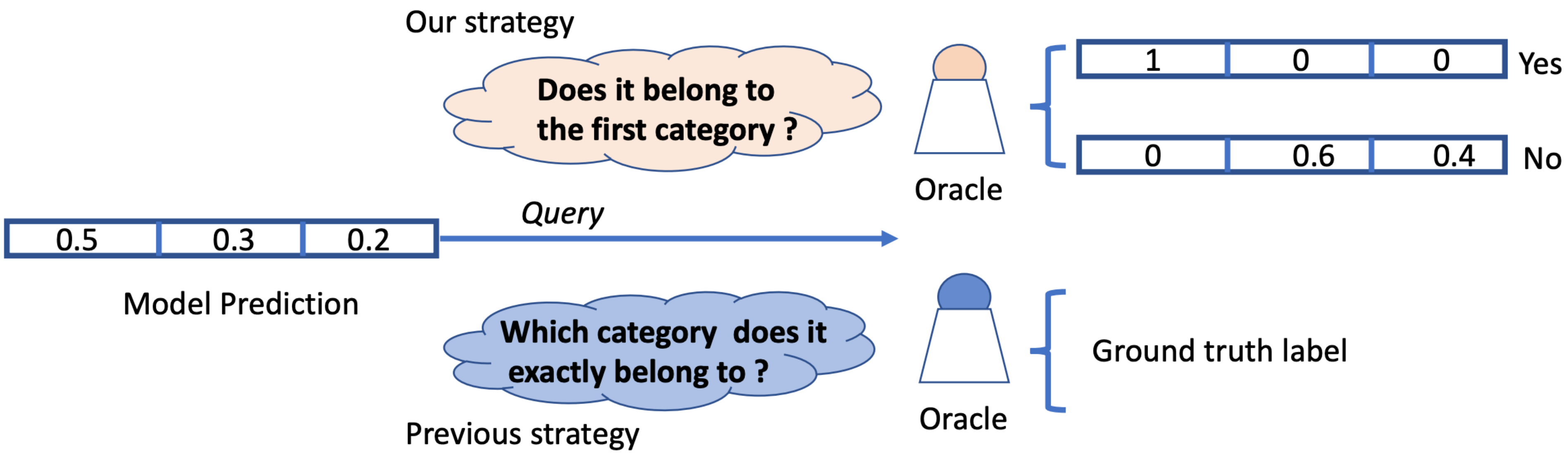
Problem of using accurate human knowledge



Exact labeling task (for ground truth labels) is costly, especially when the categorization task is specialized and strongly depend on the oracle expertise

Graph Active Learning

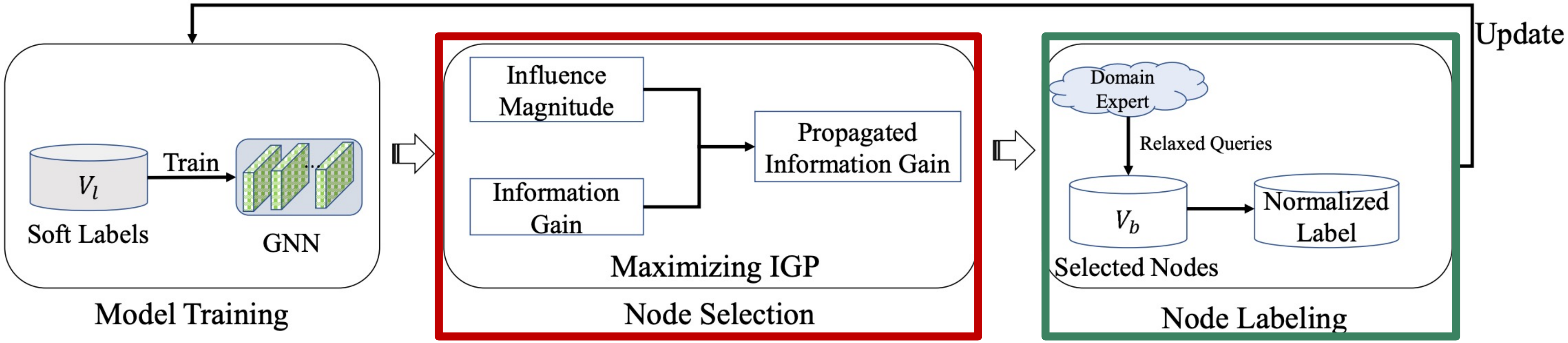
Human knowledge as soft label: IGP



- A domain expert only judges the correctness of the predicted labels (a binary question) rather than identifying the exact class (a multi-class question)
- If the model prediction is incorrect, the node is annotated with the soft label by re-normalizing the model outputs over the remaining classes

Graph Active Learning

Human knowledge as soft label: IGP



IGP measures the expected information gain of labeling each node and selects a batch of nodes that can maximize the information gain propagation on the graph

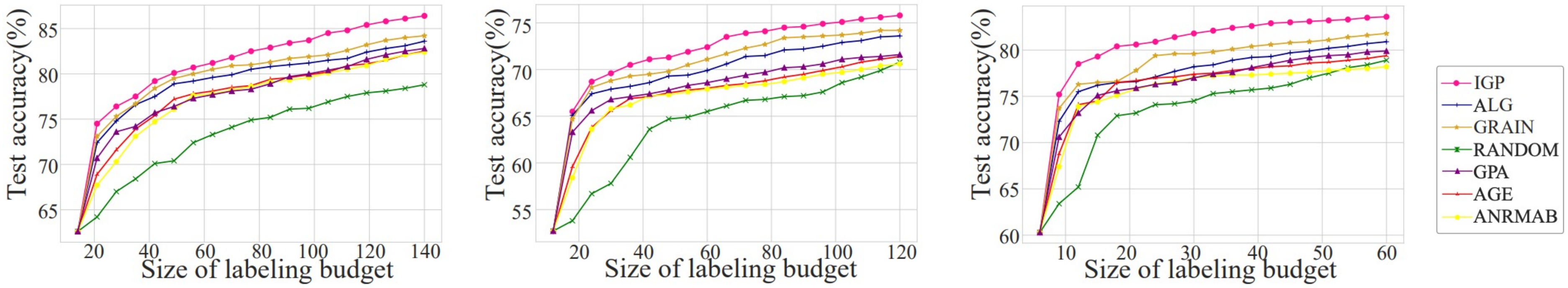
The oracle only needs to judge the correctness of the generated label

Graph Active Learning

Human knowledge as soft label: IGP

Method	Cora	Citeseer	PubMed	Reddit	ogbn-arxiv
Random	78.8(±0.8)	70.8(±0.9)	78.9(±0.6)	91.1(±0.5)	68.2(±0.4)
AGE	82.5(±0.6)	71.4(±0.6)	79.4(±0.4)	91.6(±0.3)	68.9(±0.3)
ANRMAB	82.4(±0.5)	70.6(±0.6)	78.2(±0.3)	91.5(±0.3)	68.7(±0.2)
GPA	82.8(±0.4)	71.6(±0.4)	79.9(±0.5)	91.8(±0.2)	69.2(±0.3)
SEAL	83.2(±0.5)	72.1(±0.4)	80.3(±0.4)	92.1(±0.3)	69.5(±0.1)
ALG	83.6(±0.6)	73.6(±0.5)	80.9(±0.3)	92.4(±0.3)	70.1(±0.2)
GRAIN	84.2(±0.3)	74.2(±0.3)	81.8(±0.2)	92.5(±0.1)	70.3(±0.2)
IGP	**86.4(±0.6)**	**75.8(±0.3)**	**83.6(±0.5)**	**93.4(±0.2)**	**70.9(±0.3)**

Better performance on node classification



(a) Cora (b) Citeseer (c) PubMed

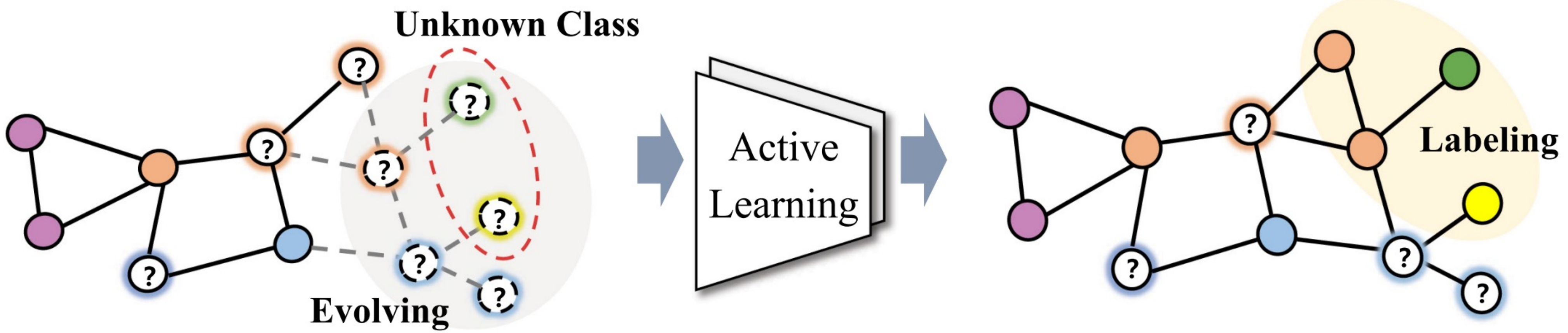
Different labeling budgets

Information Gain Propagation: a New Way to Graph Active Learning with Soft Labels

Graph Active Learning

Open world scenario: OWGAL

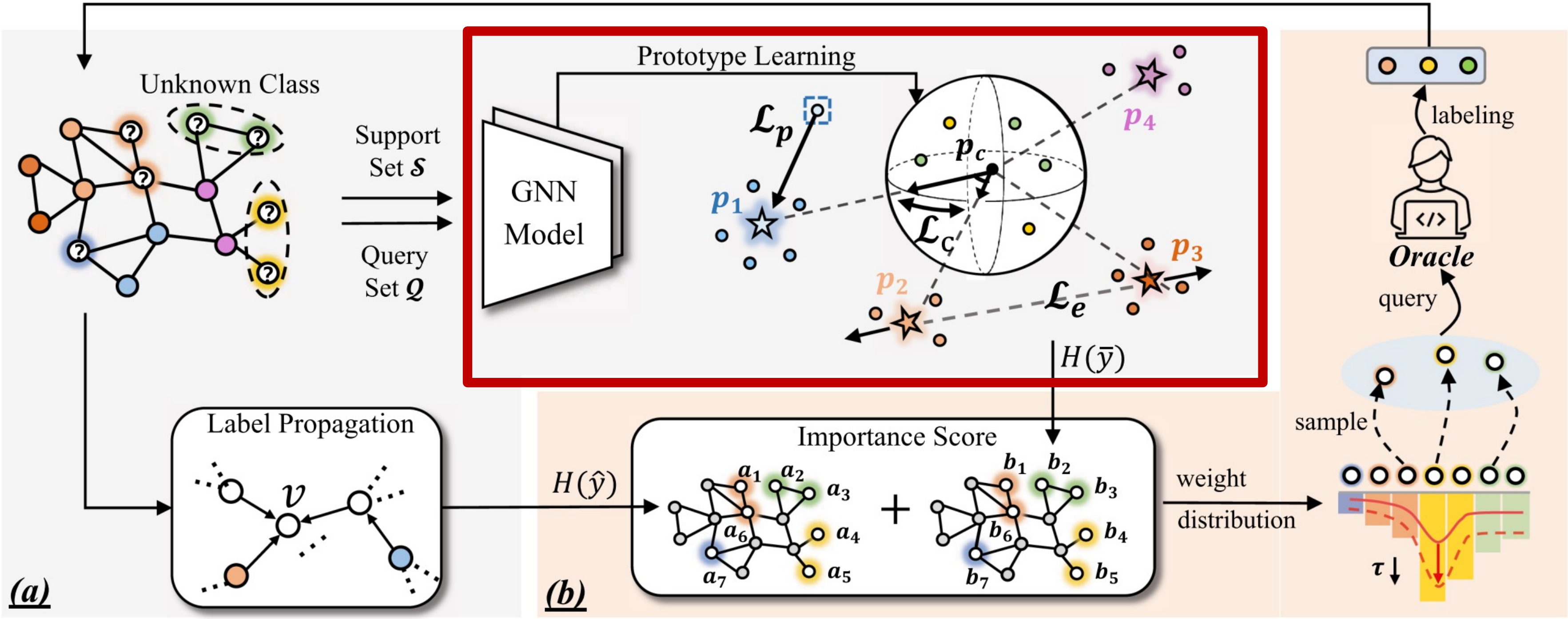
- Existing GAL methods mainly focus on a "closed-world" setting, where all nodes belong to a fixed known group of classes
- In the open world scenario, a graph develops over time as new nodes, edges, and classes are introduced, and thus GAL is required to make a fair selection among all nodes for labeling, despite its class has been seen or not



Open-world graph active learning (OWGAL) should not only select the most informative nodes, but uncover nodes of latent new classes

Graph Active Learning

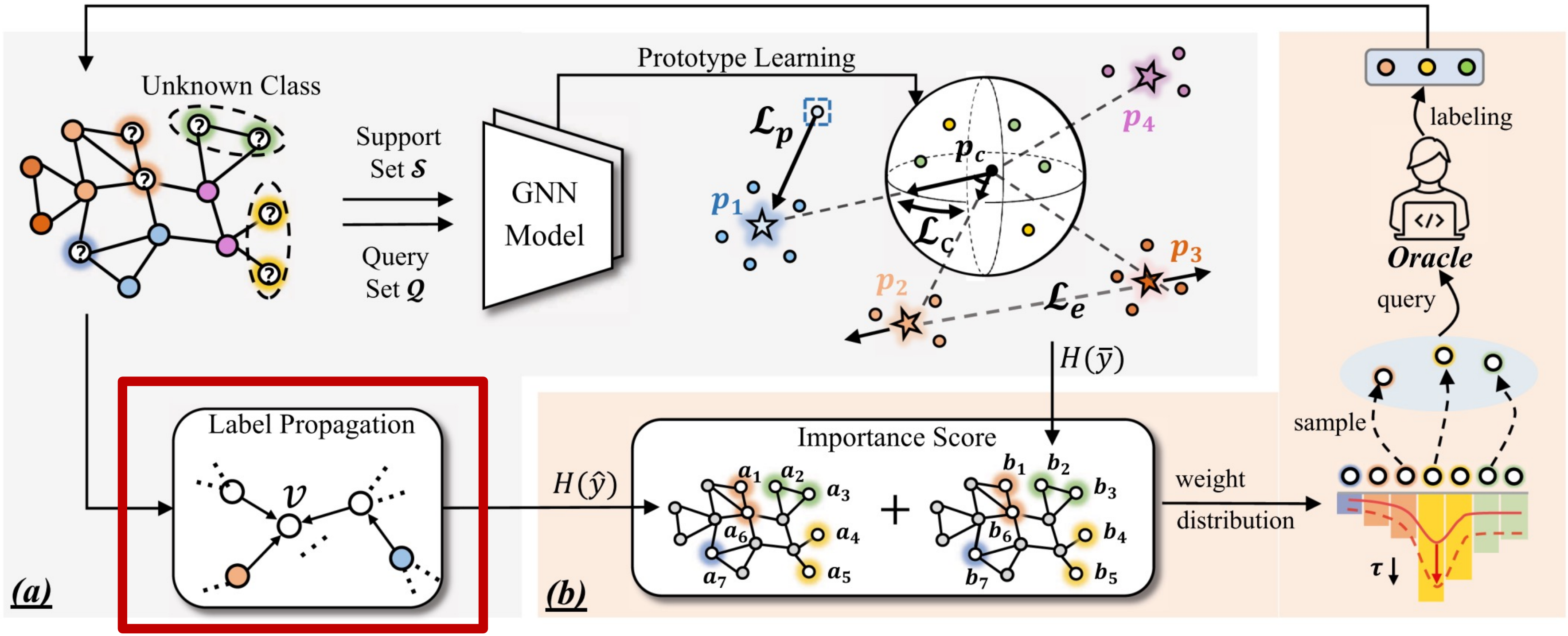
Open world scenario: OWGAL



To make known and unknown classes more identifiable, learn more compact representations for nodes in the same class, and push known classes as far away from each other as possible

Graph Active Learning

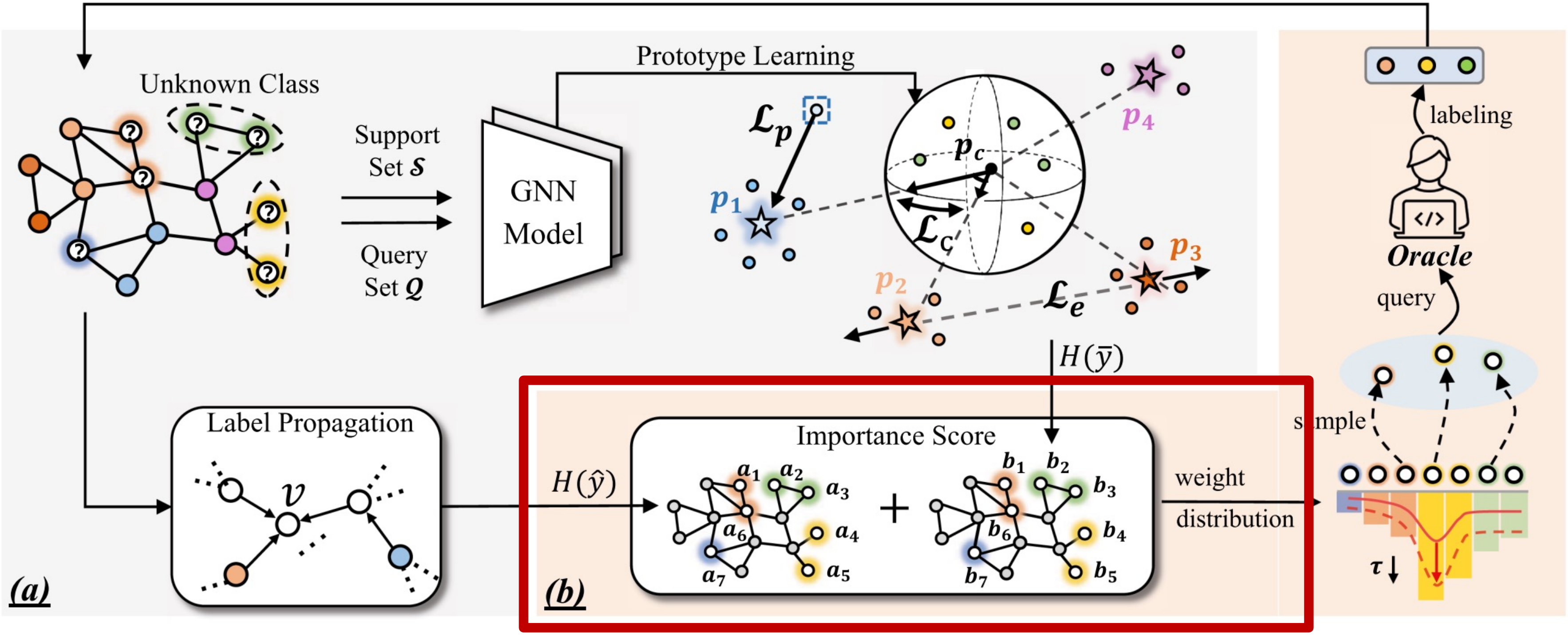
Open world scenario: OWGAL



Label propagation: motivated by the homophily assumption that adjacent nodes tend to have the same label, node v is more likely to belong to a novel class if no labeled node reach it within K steps of random walks on graph

Graph Active Learning

Open world scenario: OWGAL



Unify the discovery of unknown classes and important nodes, as it weighs unseen-class nodes as much as the nodes near the decision boundary, both of which have high uncertainty scores and should be sampled for labeling

Graph Active Learning

Open world scenario: OWGAL

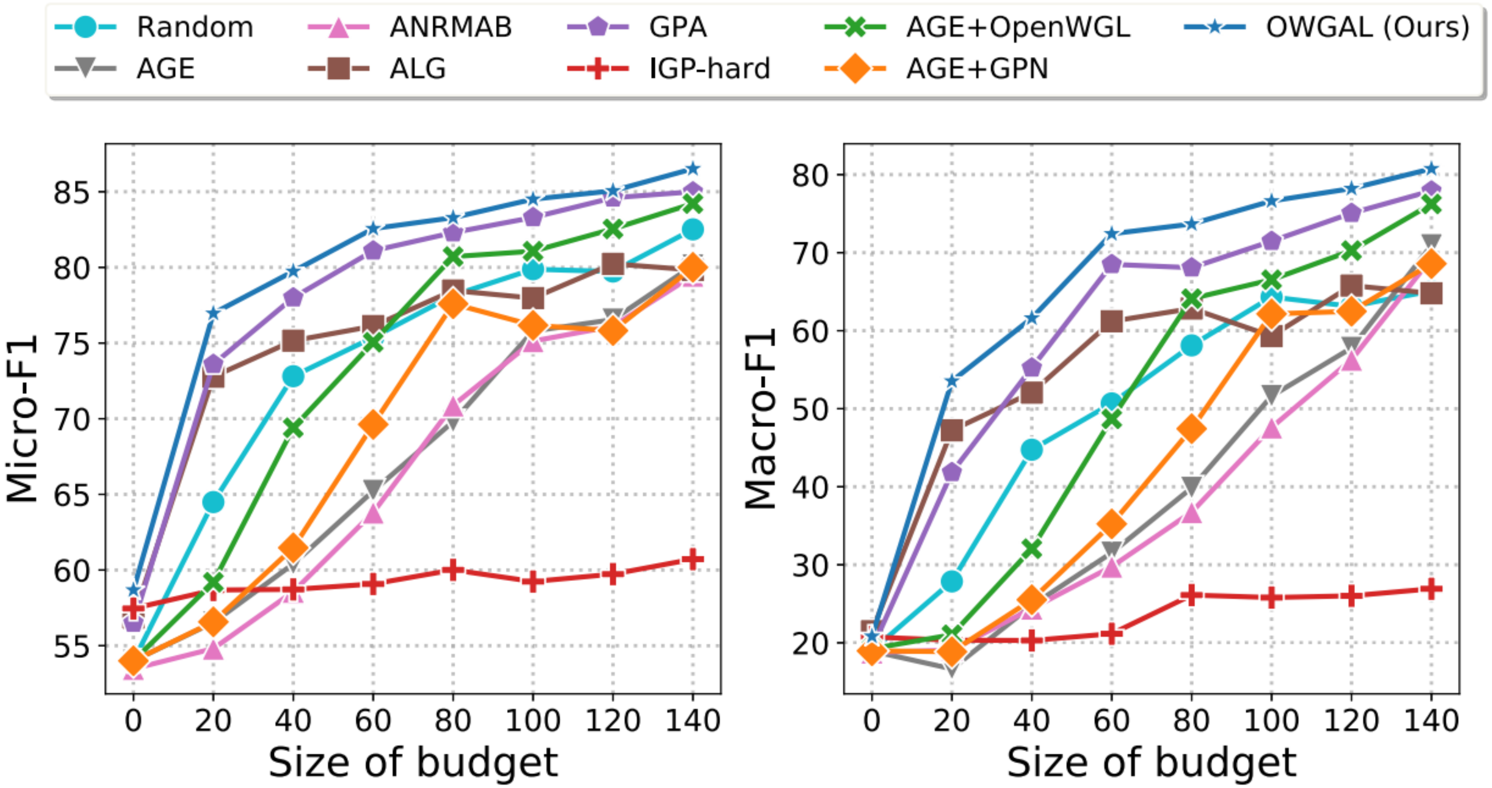
Baselines	Amazon-Computer		Amazon-Photo		Coauthor-CS		Coauthor-Physics	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
Random	82.51 ± 1.77%	74.16 ± 2.90%	89.88 ± 1.55%	86.65 ± 2.94%	88.25 ± 0.34%	80.18 ± 1.03%	91.43 ± 1.01%	87.50 ± 1.83%
AGE	80.12 ± 2.24%	70.94 ± 3.08%	90.35 ± 0.87%	89.17 ± 1.32%	89.08 ± 1.69%	82.69 ± 2.05%	91.76 ± 2.30%	88.08 ± 3.99%
ANRMAB	79.57 ± 1.53%	69.28 ± 2.77%	89.63 ± 0.53%	88.55 ± 1.07%	89.87 ± 1.32%	82.33 ± 1.55%	90.96 ± 2.02%	87.68 ± 1.82%
GPA	82.92 ± 1.45%	75.31 ± 3.71%	91.59 ± 0.41%	90.75 ± 0.45%	88.89 ± 1.51%	85.78 ± 1.82%	91.08 ± 1.49%	87.86 ± 2.22%
ALG	79.83 ± 1.85%	64.79 ± 5.91%	80.61 ± 1.30%	83.58 ± 2.64%	85.97 ± 0.09%	72.01 ± 0.19%	90.86 ± 0.28%	87.96 ± 0.74%
IGP-hard	65.72 ± 2.37%	31.92 ± 1.55%	81.47 ± 1.61%	80.23 ± 1.71%	82.34 ± 0.23%	59.51 ± 0.43%	85.92 ± 1.42%	65.28 ± 3.66%
AGE+OpenWGL	84.21 ± 0.27%	76.24 ± 1.05%	91.33 ± 1.14%	90.00 ± 1.36%	88.03 ± 0.19%	76.26 ± 0.51%	92.79 ± 0.34%	89.74 ± 0.54%
AGE+GPN	80.01 ± 2.22%	68.57 ± 4.10%	89.93 ± 0.52%	88.98 ± 0.62%	90.16 ± 0.09%	87.03 ± 0.25%	92.92 ± 0.19%	90.11 ± 0.23%
OWGAL (Ours)	**86.39 ± 1.20%**	**80.79 ± 2.80%**	**93.09 ± 0.61%**	**91.71 ± 0.79%**	**91.42 ± 0.4%**	**88.77 ± 0.5%**	**93.11 ± 0.28%**	**90.36 ± 0.62%**

Baselines	Arxiv		Reddit		Products	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
Random	62.53 ± 0.74%	33.79 ± 1.12%	85.54 ± 0.94%	70.47 ± 2.25%	63.61 ± 0.68%	22.89 ± 0.59%
AGE	61.44 ± 0.38%	31.91 ± 1.44%	78.40 ± 0.43%	55.39 ± 1.09%	62.48 ± 1.04%	23.29 ± 0.69%
ANRMAB	61.85 ± 0.54%	30.67 ± 1.83%	77.40 ± 0.58%	56.12 ± 1.36%	62.55 ± 0.93%	23.07 ± 0.83%
GPA	58.59 ± 0.34%	31.34 ± 0.98%	85.00 ± 1.52%	69.65 ± 5.96%	58.35 ± 1.06%	21.22 ± 0.25%
ALG	59.71 ± 0.50%	22.36 ± 1.28%	86.14 ± 0.67%	68.55 ± 1.27%	63.59 ± 0.88%	21.63 ± 0.67%
IGP-hard	56.35 ± 3.58%	22.92 ± 2.74%	77.37 ± 0.58%	49.21 ± 2.37%	59.97 ± 0.97%	21.31 ± 0.77%
AGE+OpenWGL	62.27 ± 0.12%	33.85 ± 0.39%	82.14 ± 1.04%	67.35 ± 3.09%	58.89 ± 0.85%	22.31 ± 0.85%
AGE+GPN	61.34 ± 0.38%	31.14 ± 0.82%	74.73 ± 1.37%	48.62 ± 2.91%	56.72 ± 0.73%	21.13 ± 0.94%
OWGAL (Ours)	**63.32 ± 0.49%**	**38.23 ± 0.62%**	**88.41 ± 0.71%**	**80.12 ± 1.35%**	**66.46 ± 0.79%**	**27.15 ± 0.63%**

Better performance on node classification

Graph Active Learning

Open world scenario: OWGAL



OWGAL achieves the best performance across different budgets

Topics for Knowledge from Humans and Domains

Graph learning enhanced by human feedback

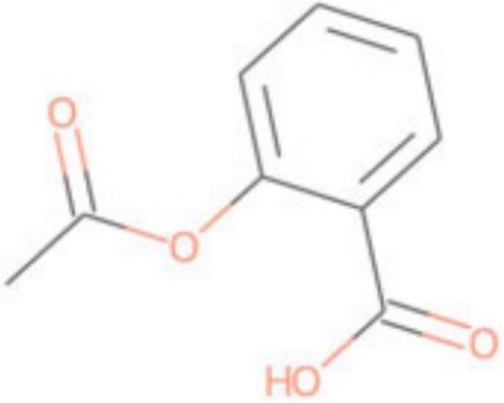
- Graph active learning with human knowledge

Graph learning enhanced by domain knowledge

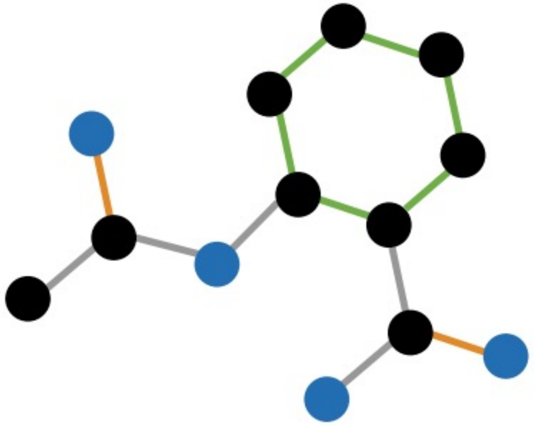
- Chemistry domain knowledge for molecular property prediction

Molecular Graphs

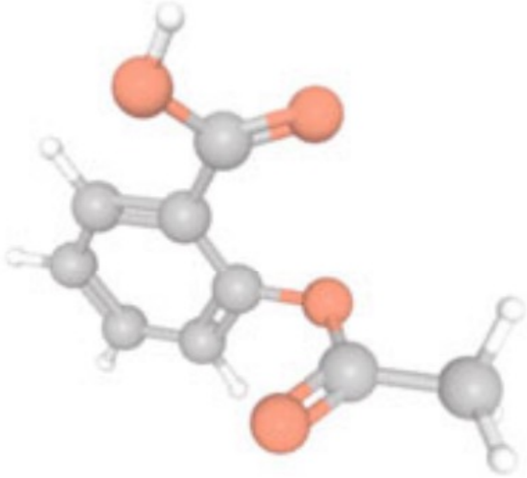
Molecules can be represented as a graph



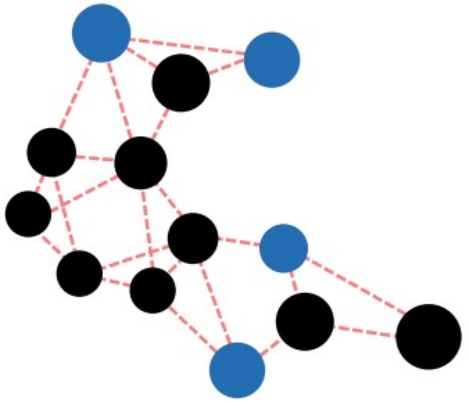
Aspirin in 2D



2D topological graph



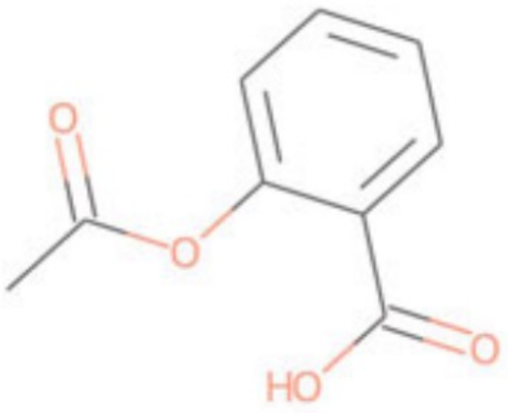
Aspirin in 3D



3D geometric graph

Domain Knowledge for Molecular Property Prediction

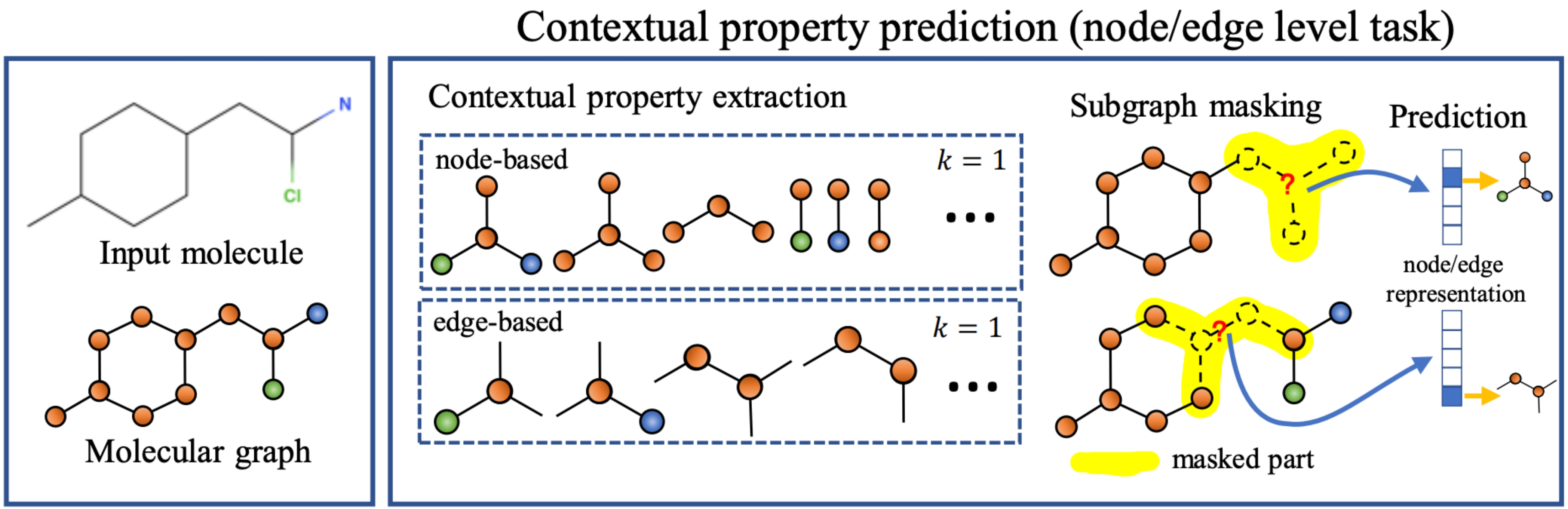
Molecular Property Prediction



$\xrightarrow{\text{model}}$ Property $\begin{cases} \text{Toxicity} \\ \text{Solubility} \\ \text{Activity} \\ \ldots \end{cases}$

Domain Knowledge for Molecular Property Prediction

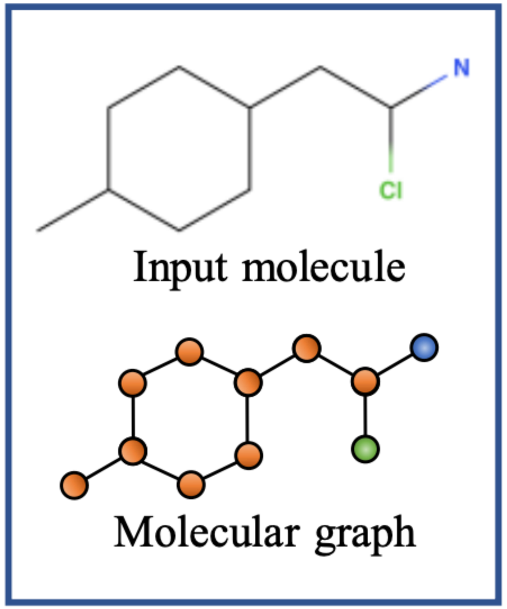
Encode molecules in a self-supervised way: GROVER



(Task 1) Contextual property prediction: predict the context-aware properties of the target node/edge within some local subgraph

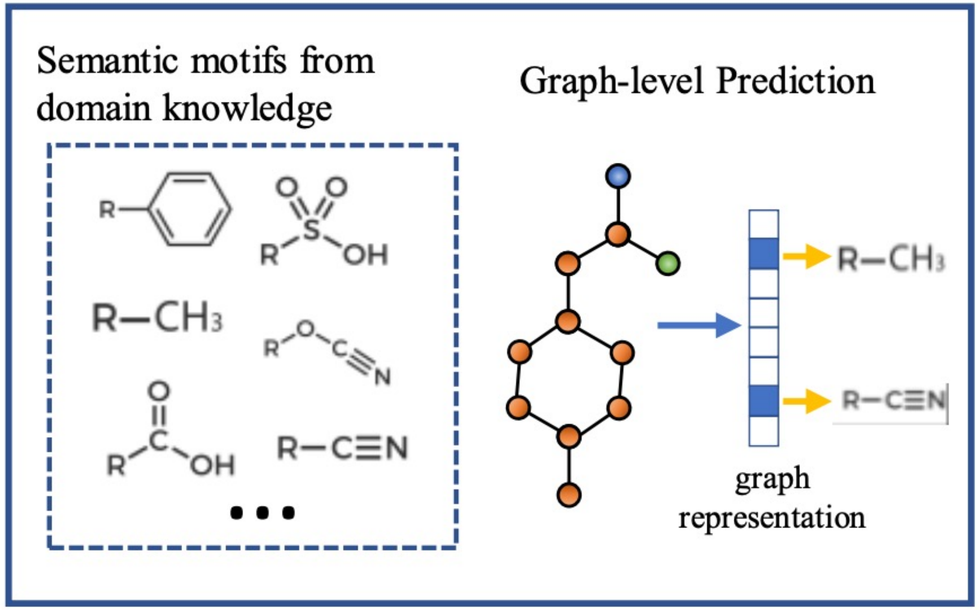
Domain Knowledge for Molecular Property Prediction

Encode molecules in a self-supervised way: GROVER



...

Graph-level motif prediction



(Task 2) Motif prediction: functional groups, as one important class of motifs in molecules, encode the rich domain knowledge of molecules

Domain Knowledge for Molecular Property Prediction

Encode molecules in a self-supervised way: GROVER

Classification (Higher is better)						
Dataset # Molecules	BBBP 2039	SIDER 1427	ClinTox 1478	BACE 1513	Tox21 7831	ToxCast 8575
TF_Robust [40]	$0.860_{(0.087)}$	$0.607_{(0.033)}$	$0.765_{(0.085)}$	$0.824_{(0.022)}$	$0.698_{(0.012)}$	$0.585_{(0.031)}$
GraphConv [24]	$0.877_{(0.036)}$	$0.593_{(0.035)}$	$0.845_{(0.051)}$	$0.854_{(0.011)}$	$0.772_{(0.041)}$	$0.650_{(0.025)}$
Weave [23]	$0.837_{(0.065)}$	$0.543_{(0.034)}$	$0.823_{(0.023)}$	$0.791_{(0.008)}$	$0.741_{(0.044)}$	$0.678_{(0.024)}$
SchNet [45]	$0.847_{(0.024)}$	$0.545_{(0.038)}$	$0.717_{(0.042)}$	$0.750_{(0.033)}$	$0.767_{(0.025)}$	$0.679_{(0.021)}$
MPNN [13]	$0.913_{(0.041)}$	$0.595_{(0.030)}$	$0.879_{(0.054)}$	$0.815_{(0.044)}$	$0.808_{(0.024)}$	$0.691_{(0.013)}$
DMPNN [63]	$0.919_{(0.030)}$	$0.632_{(0.023)}$	$0.897_{(0.040)}$	$0.852_{(0.053)}$	$0.826_{(0.023)}$	$0.718_{(0.011)}$
MGCN [30]	$0.850_{(0.064)}$	$0.552_{(0.018)}$	$0.634_{(0.042)}$	$0.734_{(0.030)}$	$0.707_{(0.016)}$	$0.663_{(0.009)}$
AttentiveFP [61]	$0.908_{(0.050)}$	$0.605_{(0.060)}$	$0.933_{(0.020)}$	$0.863_{(0.015)}$	$0.807_{(0.020)}$	$0.579_{(0.001)}$
N-GRAM [29]	$0.912_{(0.013)}$	$0.632_{(0.005)}$	$0.855_{(0.037)}$	$0.876_{(0.035)}$	$0.769_{(0.027)}$	-[4]
HU. et.al[18]	$0.915_{(0.040)}$	$0.614_{(0.006)}$	$0.762_{(0.058)}$	$0.851_{(0.027)}$	$0.811_{(0.015)}$	$0.714_{(0.019)}$
$\text{GROVER}_{\text{base}}$	$0.936_{(0.008)}$	$0.656_{(0.006)}$	$0.925_{(0.013)}$	$0.878_{(0.016)}$	$0.819_{(0.020)}$	$0.723_{(0.010)}$
$\text{GROVER}_{\text{large}}$	$\mathbf{0.940}_{(0.019)}$	$\mathbf{0.658}_{(0.023)}$	$\mathbf{0.944}_{(0.021)}$	$\mathbf{0.894}_{(0.028)}$	$\mathbf{0.831}_{(0.025)}$	$\mathbf{0.737}_{(0.010)}$

GROVER achieves the best performance on all datasets compared with existing molecular property predicting methods

Domain Knowledge for Molecular Property Prediction

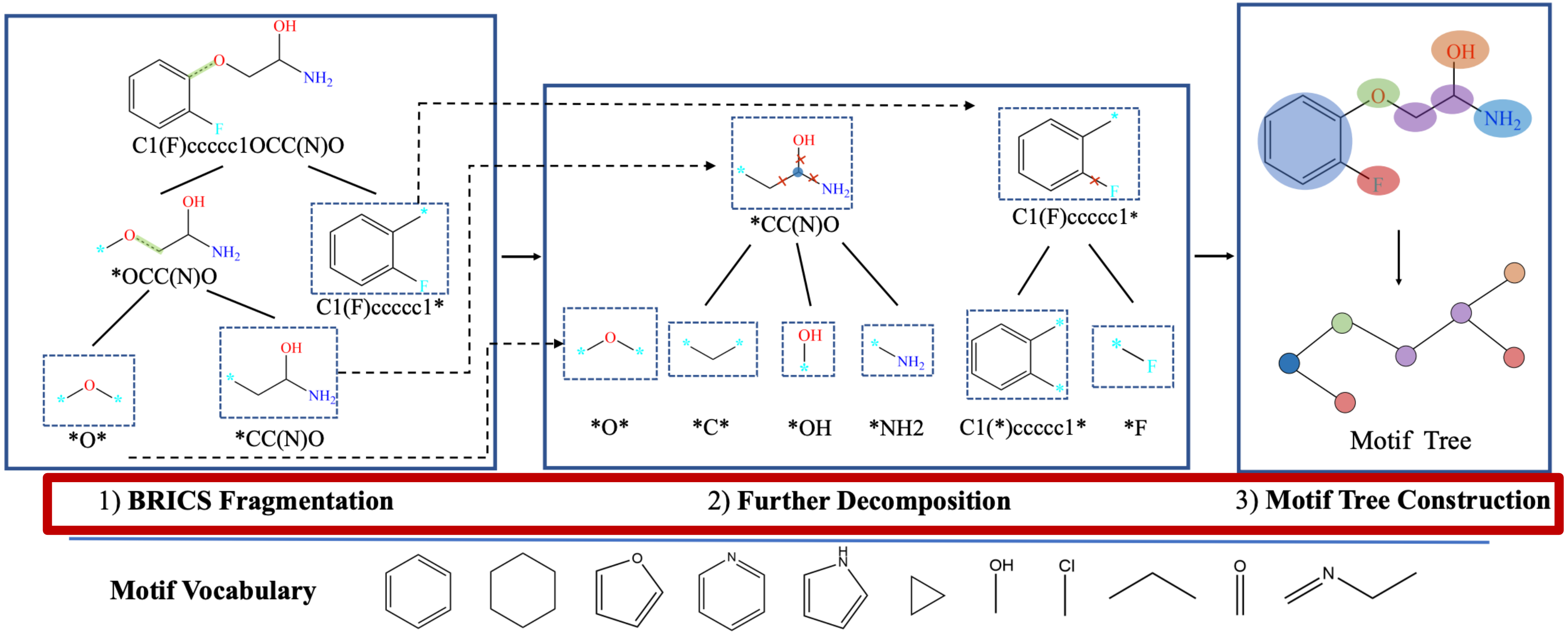
Encode molecules in a self-supervised way: GROVER

	GROVER	No Pretrain	Abs. Imp.
BBBP (2039)	**0.940**	0.911	+0.029
SIDER (1427)	**0.658**	0.624	+0.034
ClinTox (1478)	**0.944**	0.884	+0.060
BACE (1513)	**0.894**	0.858	+0.036
Tox21 (7831)	**0.831**	0.803	+0.028
ToxCast (8575)	**0.737**	0.721	+0.016
Average	**0.834**	0.803	+0.038

Self-supervised pre-training strategy can learn the implicit domain knowledge and enhance the prediction performance

Domain Knowledge for Molecular Property Prediction

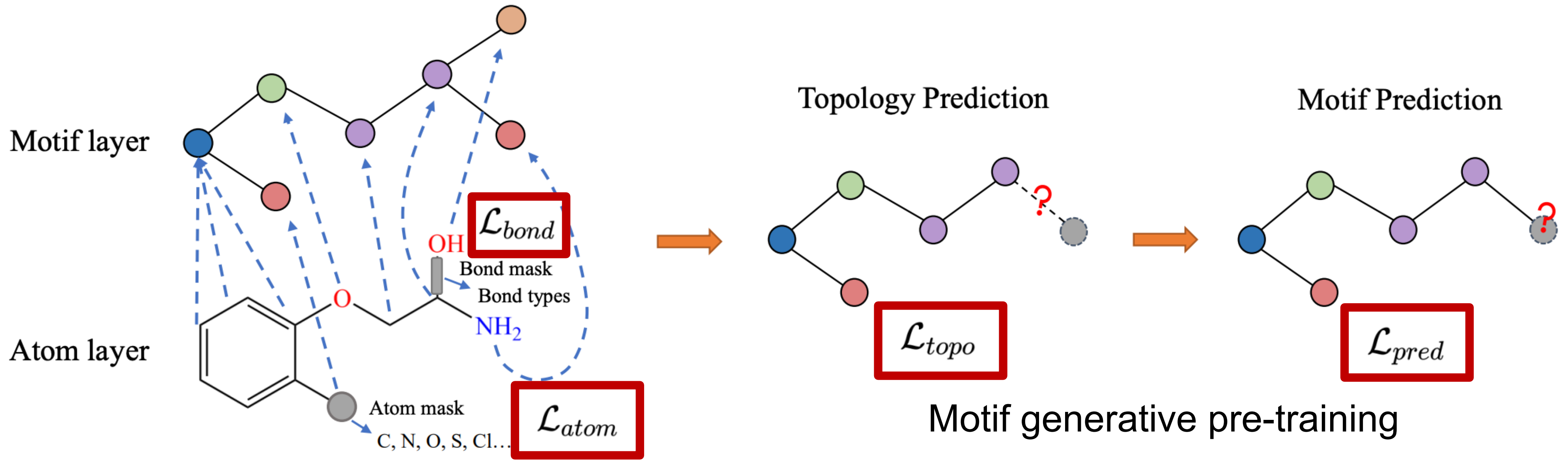
One step further, generating motifs for molecules: MGSSL



- Three steps: 1) a molecule graph is cleaved based on BRICS. 2) further decomposition to reduce the redundancy of motifs 3) construct motif trees from molecule graphs
- A motif vocabulary can be built via these 3 steps preprocessing

Domain Knowledge for Molecular Property Prediction

One step further, generating motifs for molecules: MGSSL



Multiple pre-training strategies: 1) reconstruct the masked atom and bond, 2) use the generated motif for motif generative pre-training

Domain Knowledge for Molecular Property Prediction

One step further, generating motifs for molecules: MGSSL

SSL methods	muv	clintox	sider	hiv	tox21	bace	toxcast	bbbp	Avg.
No pretrain	71.7±2.3	58.2±2.8	57.2±0.7	75.4±1.5	74.3±0.5	70.0±2.5	63.3±1.5	65.5±1.8	67.0
Infomax	75.1±2.8	73.0±3.2	58.2±0.5	76.5±1.6	75.2±0.3	75.6±1.0	62.8±0.6	68.1±1.3	70.6
Attribute masking	74.7±1.9	77.5±3.1	59.6±0.7	77.9±1.2	**77.2±0.4**	78.3±1.1	63.3±0.8	65.6±0.9	71.8
GCC	74.1±1.4	73.2±2.6	58.0±0.9	75.5±0.8	76.6±0.5	75.0±1.5	63.5±0.4	66.9±0.7	70.4
GPT-GNN	75.0±2.5	74.9±2.7	59.3±0.8	77.0±1.7	76.1±0.4	78.5±0.9	63.1±0.5	67.5±1.3	71.4
Grover	75.8±1.7	76.9±1.9	60.7±0.5	77.8±1.4	76.3±0.6	79.5±1.1	63.4±0.6	68.0±1.5	72.3
MGSSL (DFS)	78.1±1.8	79.7±2.2	60.5±0.7	**79.5±1.1**	76.4±0.4	**79.7±0.8**	63.8±0.3	**70.5±1.1**	73.5
MGSSL (BFS)	**78.7±1.5**	**80.7±2.1**	**61.8±0.8**	78.8±1.2	76.5±0.3	79.1±0.9	**64.1±0.7**	69.7±0.9	**73.7**

MGSSL achieves the best performance, demonstrating the effectiveness learning from generated motifs

Topics for Knowledge from Humans and Domains

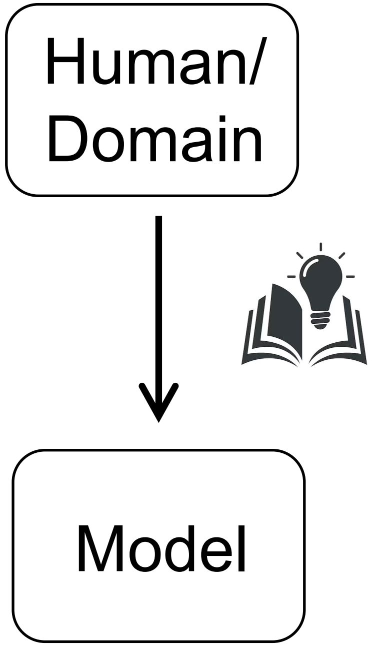
Chapter summary

Graph learning enhanced by human feedback

- Graph active learning with human knowledge

Graph learning enhanced by domain knowledge

- Chemistry domain knowledge for molecular property prediction



Q & A

Tutorial Website:

yijuntian.com/tutorial

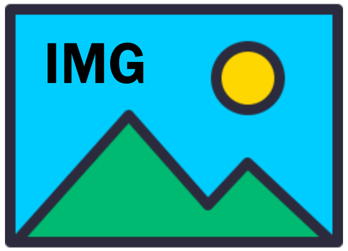
Tutorial Outline

- Preliminaries and Foundations
- Graph Learning Enhanced by Knowledge from Data
- Graph Learning Enhanced by Knowledge from Models
- Graph Learning Enhanced by Knowledge from Humans and Domains
- Graph Learning Enhanced by Knowledge from External Sources
- Knowledge-enhanced Graph Learning for Real-world Applications
- Summary and Future Directions

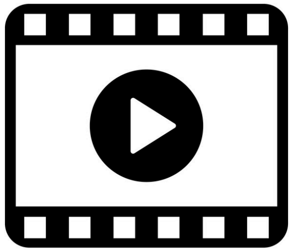
Knowledge from External Sources

Text

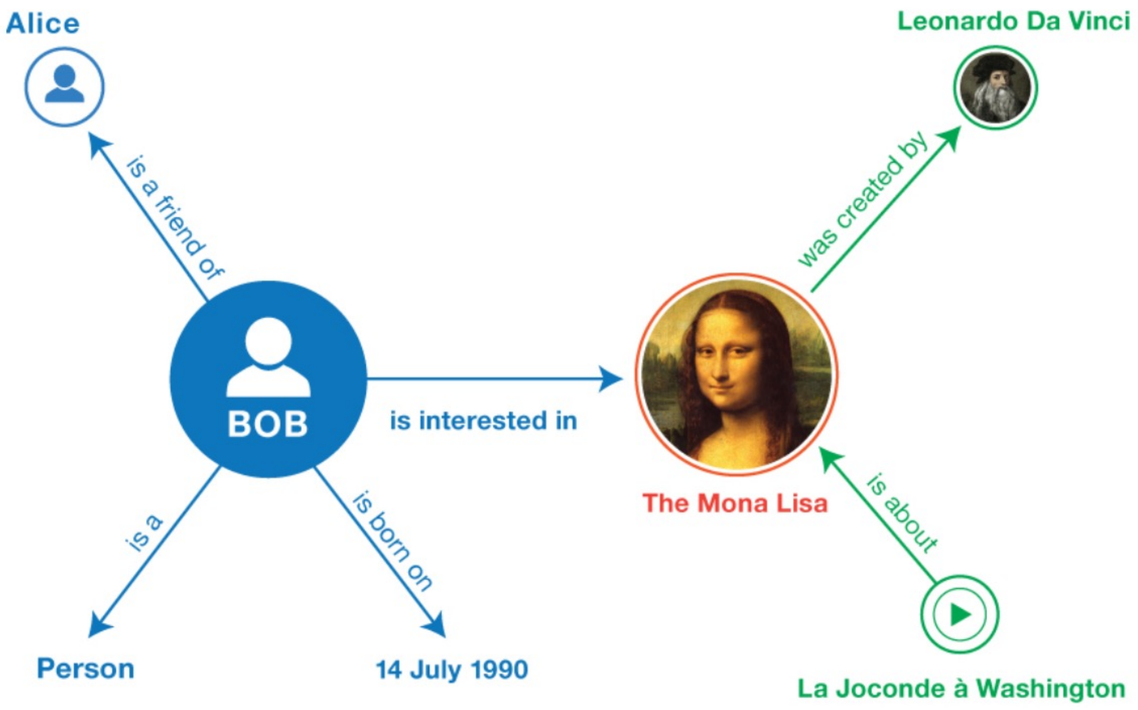
Wikipedia



Image



Video



Knowledge triplets

Topics for Knowledge from External Sources

Graph learning on text-rich graphs

- Graph learning on textual-node graphs
- Graph learning on textual-edge graphs

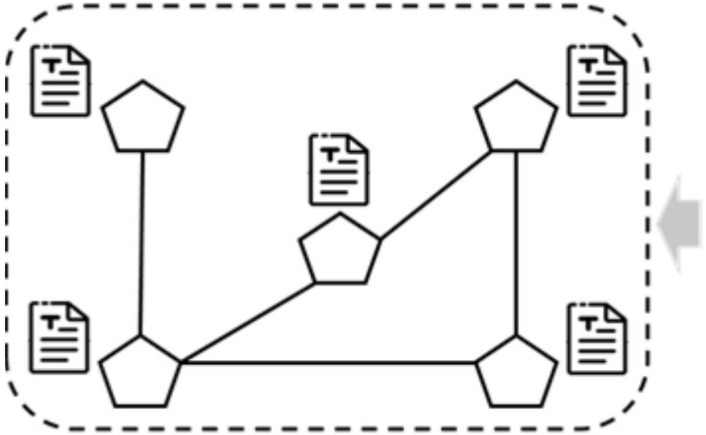
Graph learning on knowledge graphs

- Knowledge Graph Embedding
- Advancing KG tasks with text data

Graph Learning on Textual-node Graphs

What are textual-node graphs?

- A text-rich network contains a node set, an edge set, and a text set
- Each node is associated with some textual information

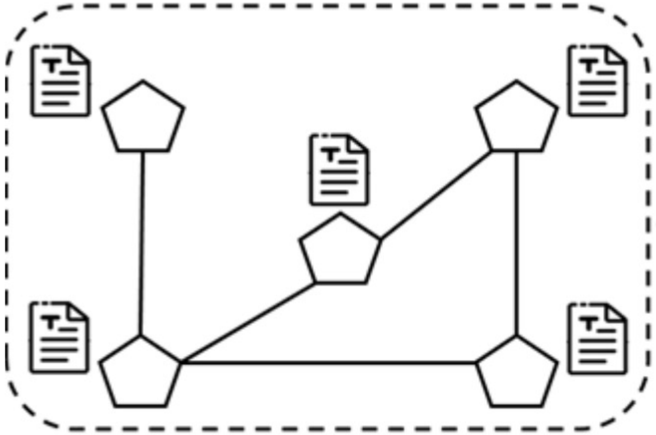

Text-rich network

ID	Document Text
1	Privacy preservation in wireless sensor…
2	Tabu search for the Steiner problem…
3	Feature interaction: a critical review…
4	A study of malware in P2P networks…
5	Understanding the paradoxical effects…

Graph Learning on Textual-node Graphs

Why do we need text-rich graphs?

- Text-rich graphs provide additional context and explanation to graphs
- Text allows for modeling the latent correlation between nodes



Text-rich network

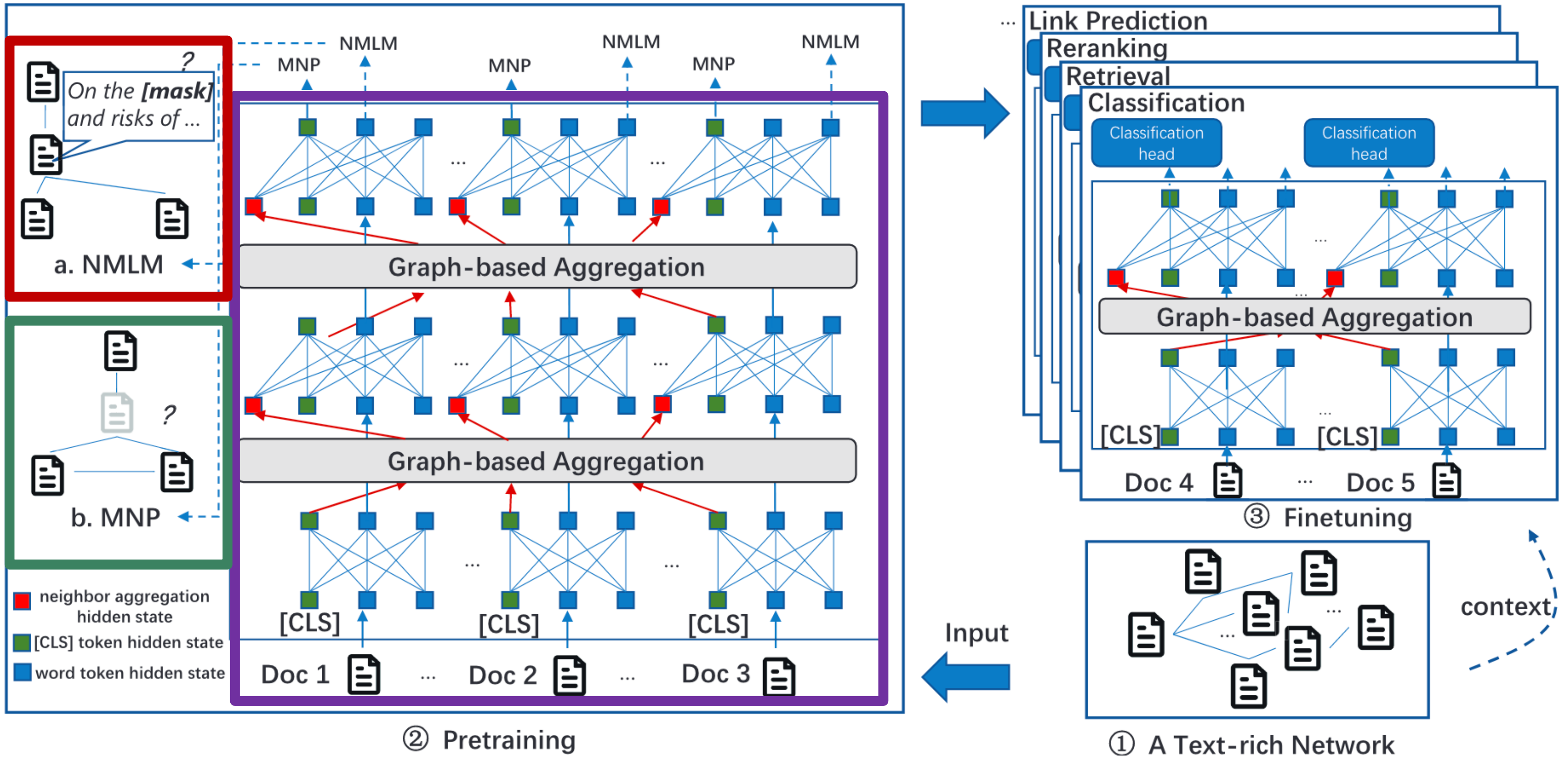
ID	Document Text
1	Privacy preservation in wireless sensor…
2	Tabu search for the Steiner problem…
3	Feature interaction: a critical review…
4	A study of malware in P2P networks…
5	Understanding the paradoxical effects…

Graph Learning on Textual-node Graphs

Pretraining-finetuning on textual-node graphs: Patton

Network-contextualized Masked Language Modeling: mask several tokens in the text sequence and utilizes the surrounding unmasked tokens to predict them

Masked Node Prediction: Predict the masked nodes based on the adjacent network structure



Graphformer

PATTON: Language Model Pretraining on Text-Rich Networks

Graph Learning on Textual-node Graphs

Pretraining-finetuning on text-rich graphs: Patton

Off-the-shelf PLMs

Continuous pretraining method

Method	Mathematics Macro-F1	Mathematics Micro-F1	Geology Macro-F1	Geology Micro-F1	Economics Macro-F1	Economics Micro-F1	Clothes Macro-F1	Clothes Micro-F1	Sports Macro-F1	Sports Micro-F1
BERT	18.14$_{0.07}$	22.04$_{0.32}$	21.97$_{0.87}$	29.63$_{0.36}$	14.17$_{0.08}$	19.77$_{0.12}$	45.10$_{1.47}$	68.54$_{2.25}$	31.88$_{0.23}$	34.58$_{0.56}$
GraphFormers	18.69$_{0.52}$	23.24$_{0.46}$	22.64$_{0.92}$	31.02$_{1.16}$	13.68$_{1.03}$	19.00$_{1.44}$	46.27$_{1.92}$	68.97$_{2.46}$	43.77$_{0.63}$	50.47$_{0.78}$
SciBERT	23.50$_{0.64}$	23.10$_{2.23}$	29.49$_{1.25}$	37.82$_{1.89}$	15.91$_{0.48}$	21.32$_{0.66}$	-	-	-	-
SPECTER	23.37$_{0.07}$	29.83$_{0.96}$	30.40$_{0.48}$	38.54$_{0.77}$	16.16$_{0.17}$	19.84$_{0.47}$	-	-	-	-
SimCSE (unsup)	20.12$_{0.08}$	26.11$_{0.39}$	38.78$_{0.19}$	38.55$_{0.17}$	14.54$_{0.26}$	19.07$_{0.43}$	42.70$_{2.32}$	58.72$_{0.34}$	41.91$_{0.85}$	59.19$_{0.55}$
SimCSE (sup)	20.39$_{0.07}$	25.56$_{0.00}$	25.66$_{0.28}$	33.89$_{0.40}$	15.03$_{0.53}$	18.64$_{1.32}$	52.82$_{0.87}$	75.54$_{0.98}$	46.69$_{0.10}$	59.19$_{0.55}$
LinkBERT	15.78$_{0.91}$	19.75$_{1.19}$	24.08$_{0.58}$	31.32$_{0.04}$	12.71$_{0.12}$	16.39$_{0.22}$	44.94$_{2.52}$	65.33$_{4.34}$	35.60$_{0.33}$	38.30$_{0.09}$
BERT.MLM	23.44$_{0.39}$	31.75$_{0.58}$	36.31$_{0.36}$	48.04$_{0.69}$	16.60$_{0.21}$	22.71$_{1.16}$	46.98$_{0.84}$	68.00$_{0.84}$	62.21$_{0.13}$	75.43$_{0.74}$
SciBERT.MLM	23.34$_{0.42}$	30.11$_{0.97}$	36.94$_{0.28}$	46.54$_{0.40}$	16.28$_{0.38}$	21.41$_{0.81}$	-	-	-	-
SimCSE.in-domain	25.15$_{0.09}$	29.85$_{0.20}$	38.91$_{0.08}$	48.93$_{0.14}$	18.08$_{0.22}$	23.79$_{0.44}$	57.03$_{0.20}$	80.16$_{0.31}$	65.57$_{0.35}$	75.22$_{0.18}$
PATTON	**27.58**$_{0.03}$	**32.82**$_{0.01}$	39.35$_{0.06}$	48.19$_{0.15}$	19.32$_{0.05}$	25.12$_{0.05}$	**60.14**$_{0.28}$	**84.88**$_{0.09}$	**67.57**$_{0.08}$	**78.60**$_{0.15}$
SciPATTON	27.35$_{0.04}$	31.70$_{0.01}$	**39.65**$_{0.10}$	**48.93**$_{0.06}$	**19.91**$_{0.08}$	**25.68**$_{0.32}$	-	-	-	-
w/o NMLM	25.91$_{0.45}$	27.79$_{2.07}$	38.78$_{0.19}$	48.48$_{0.17}$	18.86$_{0.23}$	24.25$_{0.26}$	56.68$_{0.24}$	80.27$_{0.17}$	65.83$_{0.28}$	76.24$_{0.54}$
w/o MNP	24.79$_{0.65}$	29.44$_{1.50}$	38.00$_{0.73}$	47.82$_{1.06}$	18.69$_{0.59}$	25.63$_{1.44}$	47.35$_{1.20}$	68.50$_{2.60}$	64.23$_{1.53}$	76.03$_{1.67}$

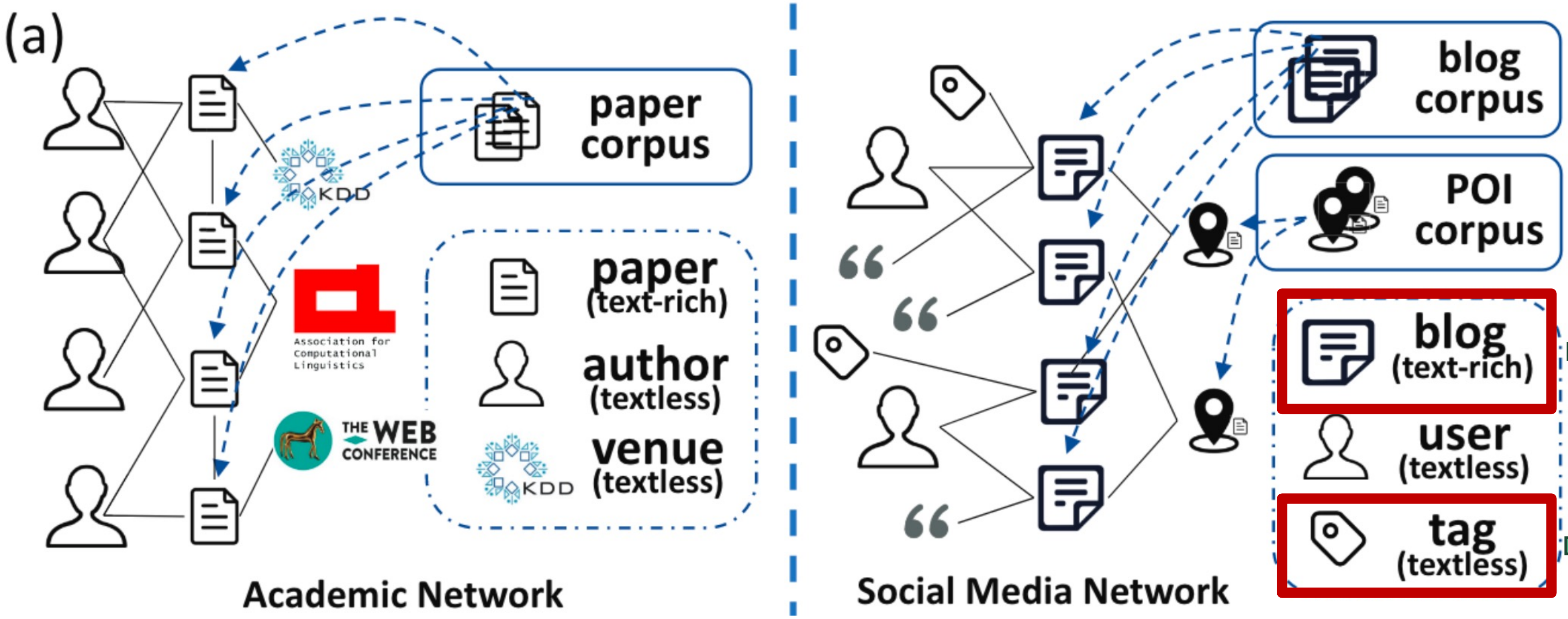
Consistently better

PATTON: Language Model Pretraining on Text-Rich Networks

Graph Learning on Textual-node Graphs

Heterogeneous text-rich graphs

A heterogeneous network includes the sets of nodes, edges, node types, and edge types.



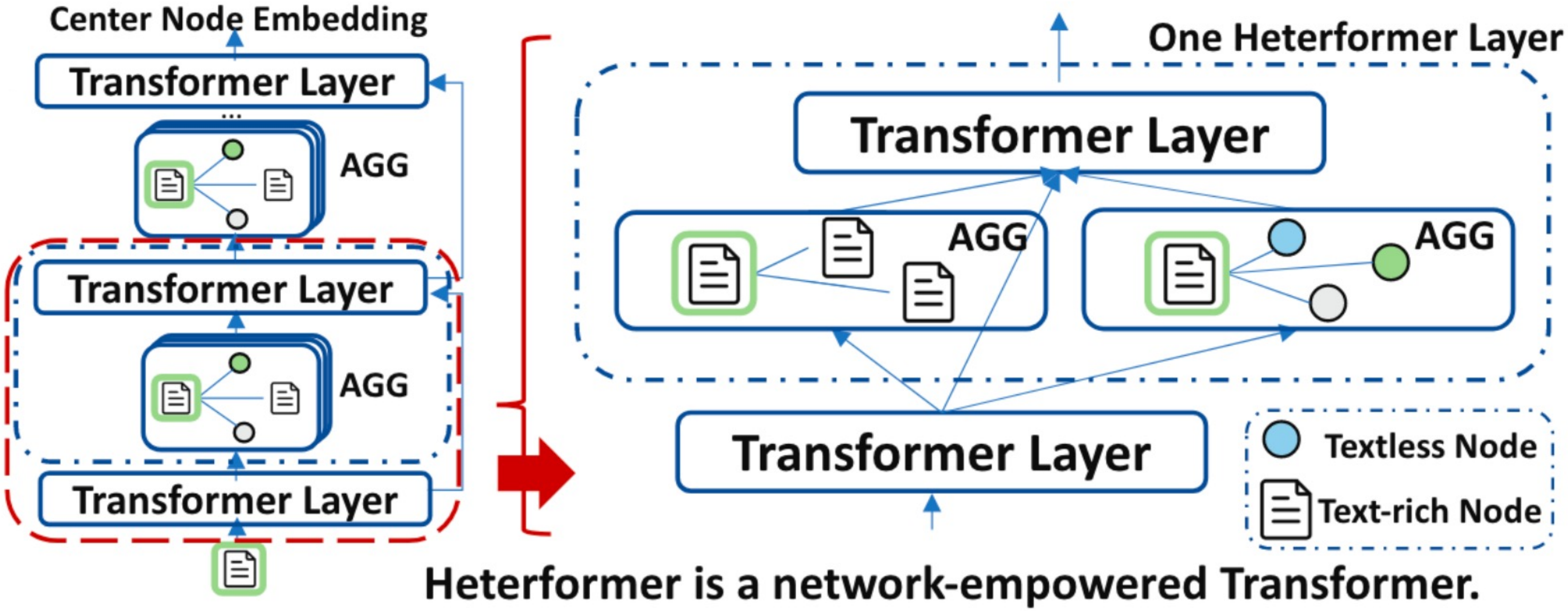
Text-rich if it is associated with semantically rich text information

Textless if it is not associated with semantically rich text information

Graph Learning on Textual-node Graphs

Learn on heterogeneous text-rich networks: Heterformer

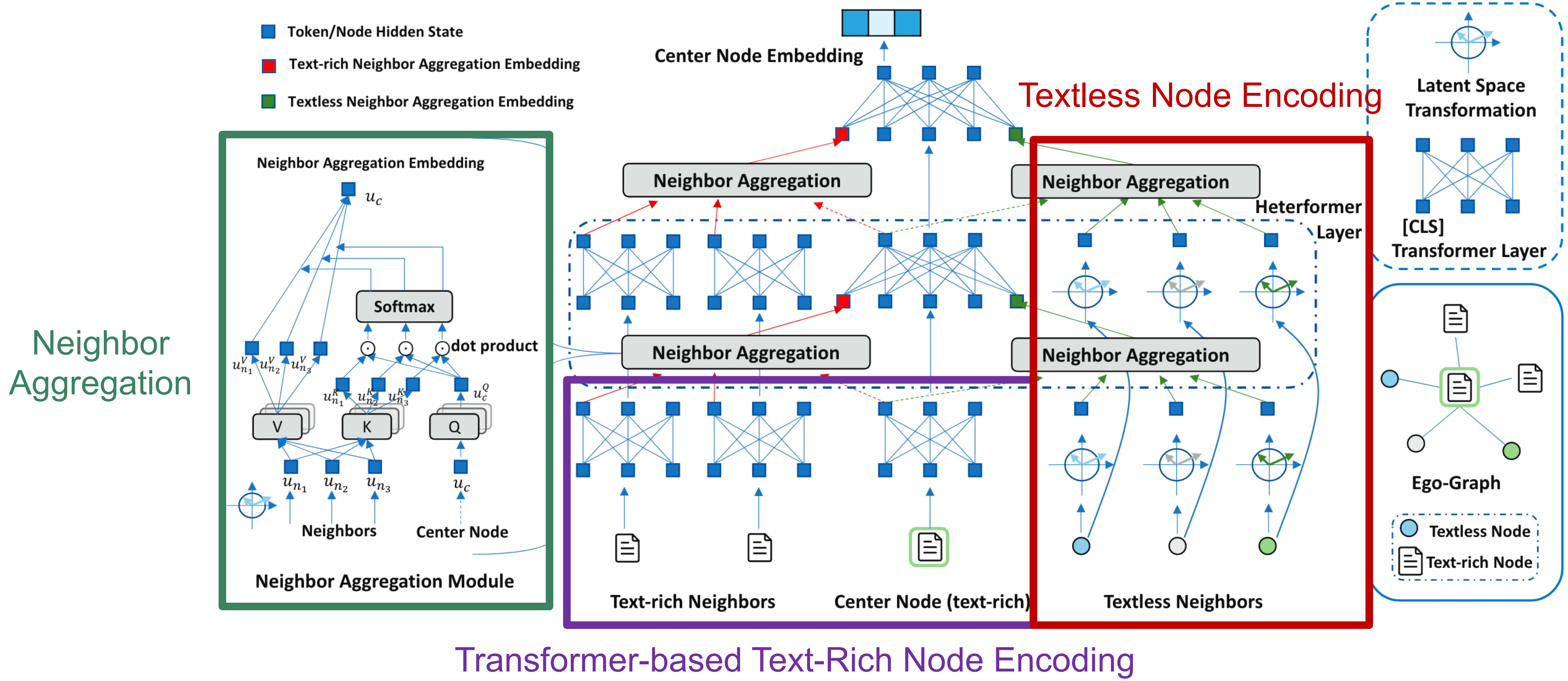
Heterformer is a network-empowered Transformer



Heterformer is a network-empowered Transformer.

Graph Learning on Textual-node Graphs

Learn on heterogeneous text-rich networks: Heterformer



Heterformer: Transformer-based Deep Node Representation Learning on Heterogeneous Text-Rich Networks

Graph Learning on Textual-node Graphs

Learn on heterogeneous text-rich networks: Heterformer

	Method	DBLP PREC	DBLP MRR	DBLP NDCG	Twitter PREC	Twitter MRR	Twitter NDCG	Goodreads PREC	Goodreads MRR	Goodreads NDCG
	MeanSAGE	0.7019	0.7964	0.8437	0.6489	0.7450	0.7991	0.6302	0.7409	0.8001
	BERT	0.7569	0.8340	0.8726	0.7179	0.7833	0.8265	0.5571	0.6668	0.7395
Homo GNN	BERT+MeanSAGE	0.8131	0.8779	0.9070	0.7201	0.7845	0.8275	0.7301	0.8167	0.8594
Homo GNN	BERT+MAXSAGE	0.8193	0.8825	0.9105	0.7198	0.7845	0.8276	0.7280	0.8164	0.8593
Homo GNN	BERT+GAT	0.8119	0.8771	0.9063	0.7231	0.7873	0.8300	0.7333	0.8170	0.8593
Homo GNN	GraphFormers	0.8324	0.8916	0.9175	0.7258	0.7891	0.8312	0.7444	0.8260	0.8665
Hetero GNN	BERT+RGCN	0.7979	0.8633	0.8945	0.7111	0.7764	0.8209	0.7488	0.8303	0.8699
Hetero GNN	BERT+HAN	0.8136	0.8782	0.9072	0.7237	0.7880	0.8306	0.7329	0.8174	0.8597
Hetero GNN	BERT+HGT	0.8170	0.8814	0.9098	0.7153	0.7800	0.8237	0.7224	0.8112	0.8552
Hetero GNN	BERT+SHGN	0.8149	0.8785	0.9074	0.7218	0.7866	0.8295	0.7362	0.8195	0.8613
Hetero GNN	GraphFormers++	0.8233	0.8856	0.9130	0.7159	0.7799	0.8236	0.7536	0.8328	0.8717
	Heterformer	**0.8474***	**0.9019***	**0.9255***	**0.7272***	**0.7908***	**0.8328***	**0.7633***	**0.8400***	**0.8773***

Heterformer captures the heterogeneous structure information and the rich contextualized textual information hidden inside the networks

Heterformer: Transformer-based Deep Node Representation Learning on Heterogeneous Text-Rich Networks

Graph Learning on Textual-edge Graphs

What are textual-edge graphs?

A text-rich network contains a node set, an edge set, and a text set.

Each edge is associated with some textual information

- When a person replies to another on social media, there will be a directed edge between them accompanied by the response texts
- When a user comments on an item, the user’s review will be naturally associated with the user-item edge

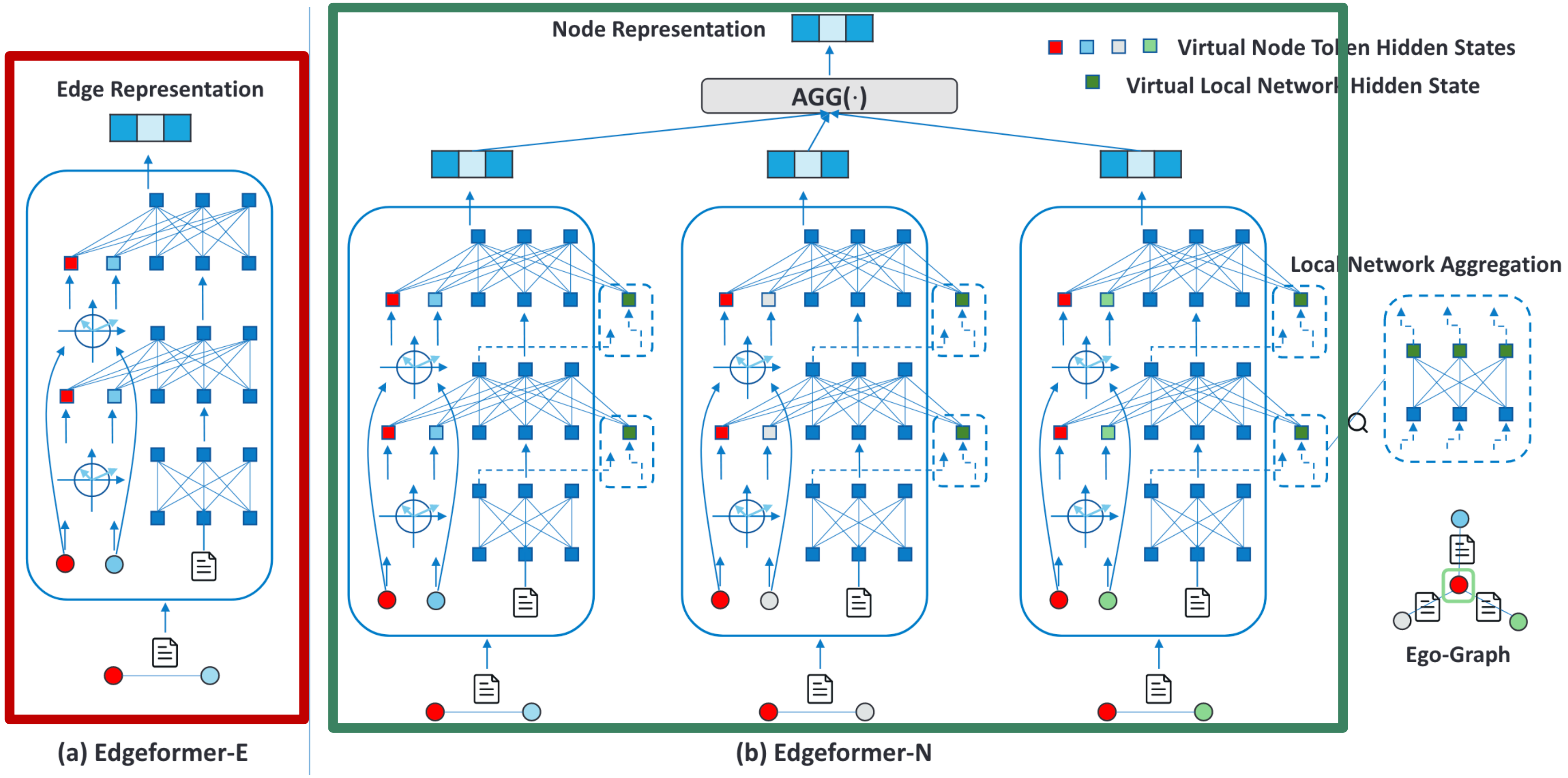
Why do we need textual-edge graphs?

- Textual-edge is ubiquitous in the real world
- Textual-edge provides rich information to describe the relationship between nodes

Graph Learning on Textual-edge Graphs

Learn on textual-edge graphs: Edgeformer

Network-aware Edge Representation Learning



Text-aware Node Representation Learning

Edgeformers: Graph-Empowered Transformers for Representation Learning on Textual-Edge Networks

Graph Learning on Textual-edge Graphs

Learn on textual-edge graphs: Edgeformer

Edge Classification

Model	Amazon-Movie Macro-F1	Amazon-Movie Micro-F1	Amazon-Apps Macro-F1	Amazon-Apps Micro-F1	Goodreads-Crime Macro-F1	Goodreads-Crime Micro-F1	Goodreads-Children Macro-F1	Goodreads-Children Micro-F1
TF-IDF	50.01	64.22	48.30	62.88	43.07	51.72	39.42	49.90
TF-IDF+nodes	53.59	66.34	50.56	65.08	49.35	57.50	47.32	56.78
BERT	61.38	71.36	59.11	69.27	56.41	61.29	51.57	57.72
BERT+nodes	*63.00*	*72.45*	*59.72*	*70.82*	*58.64*	*65.02*	*54.42*	*60.46*
Edgeformer-E	**64.18**	**73.59**	**60.67**	**71.28**	**61.03**	**65.86**	**57.45**	**61.71**

Link Prediction

Model	Amazon-Movie MRR	Amazon-Movie NDCG	Amazon-Apps MRR	Amazon-Apps NDCG	Goodreads-Crime MRR	Goodreads-Crime NDCG	Goodreads-Children MRR	Goodreads-Children NDCG	StackOverflow MRR	StackOverflow NDCG
MF	0.2032	0.3546	0.1482	0.3052	0.1923	0.3443	0.1137	0.2716	0.1040	0.2642
MeanSAGE	0.2138	0.3657	0.1766	0.3343	0.1832	0.3368	0.1066	0.2647	0.1174	0.2768
MaxSAGE	0.2178	0.3694	0.1674	0.3258	0.1846	0.3387	0.1066	0.2647	0.1173	0.2769
GIN	0.2140	0.3648	0.1797	0.3362	0.1846	0.3374	0.1128	0.2700	0.1189	0.2778
CensNet	0.2048	0.3568	0.1894	0.3457	0.1880	0.3398	0.1157	0.2726	0.1235	0.2806
NENN	0.2565	0.4032	0.1996	0.3552	0.2173	0.3670	0.1297	0.2854	0.1257	0.2854
BERT	0.2391	0.3864	0.1790	0.3350	0.1986	0.3498	0.1274	0.2836	0.1666	0.3252
BERT+MaxSAGE	0.2780	0.4224	0.2055	0.3602	0.2193	0.3694	0.1312	0.2872	0.1681	0.3264
BERT+MeanSAGE	0.2491	0.3972	0.1983	0.3540	0.1952	0.3477	0.1223	0.2791	0.1678	0.3264
BERT+GIN	0.2573	0.4037	0.2000	0.3552	0.2007	0.3522	0.1238	0.2801	*0.1708*	*0.3279*
GraphFormers	0.2756	0.4198	0.2066	0.3607	0.2176	0.3684	0.1323	0.2887	0.1693	0.3278
BERT+CensNet	0.1919	0.3462	0.1544	0.3132	0.1437	0.3000	0.0847	0.2436	0.1173	0.2789
BERT+NENN	*0.2821*	*0.4256*	*0.2127*	*0.3666*	*0.2262*	*0.3756*	*0.1365*	*0.2925*	0.1619	0.3215
Edgeformer-N	**0.2919**	**0.4344**	**0.2239**	**0.3771**	**0.2395**	**0.3875**	**0.1446**	**0.3000**	**0.1754**	**0.3339**
+ Δ %	3.5%	2.1%	5.3%	2.9%	5.9%	3.2%	5.9%	2.6%	2.7%	1.8%

Significant Improvement

Edgeformers: Graph-Empowered Transformers for Representation Learning on Textual-Edge Networks

Topics for Knowledge from External Sources

Graph learning on text-rich graphs

- Graph learning on textual-node graphs
- Graph learning on textual-edge graphs

Graph learning on knowledge graphs

- Knowledge Graph Embedding
- Advancing KG tasks with text data

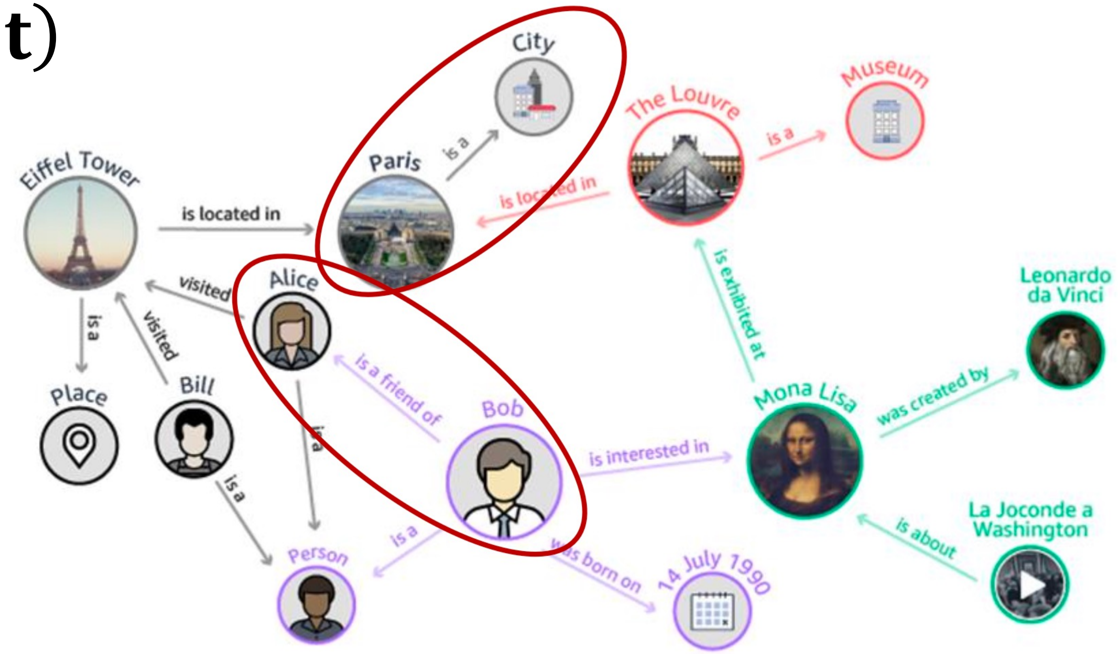
Knowledge Graphs

A collection of interlinked entities

- Objects, events or concepts
- Multiple types of entities and relations exist

Facts are represented as triplets $(\mathbf{h}, \boldsymbol{r}, \mathbf{t})$

- (‘Paris’, ‘is_a’, ‘city’)
- (‘Alice’, ‘is_friend_of’, ‘Bob’)



Knowledge Graph Embedding

Preliminaries

Goal: Encode (1) entities as low-dimensional vectors and (2) relations as parametric algebraic operations in the continuous space

How-to: Design a score function $f_{\mathbf{r}}(\mathbf{h}, \mathbf{t})$ w.r.t. such embedding vectors so that a true triplet receives higher score than a false one

KGE design rationale: Capture KG patterns

- Symmetry, antisymmetry, inversion and composition

Applications of knowledge graph embedding

- Knowledge graph completion
- Question answering
- Recommender system

Notation & Symbols

- h: head entity
- r: *relation*
- t: tail entity
- $f_{\mathbf{r}}(\mathbf{h}, \mathbf{t})$: the score function
- $d_{\mathbf{r}}(\mathbf{h}, \mathbf{t})$: the distance function
- True/positive triplet: (h, r, t)
- False/negative triplet: (h', r, t), (h, r, t'), (h', r, t')

Knowledge Graph Embedding

Preliminaries: Symmetric/Antisymmetric Relations

Symmetric/Antisymmetric Relations

- **Symmetric:** e.g., Marriage
- **Antisymmetric:** e.g., hasChild

Formally:

r is **Symmetric**: $r(x,y) \Rightarrow r(y,x)$ if $\forall\, x,y$

r is **Antisymmetric**: $r(x,y) \Rightarrow \neg r(y,x)$ if $\forall\, x,y$

Knowledge Graph Embedding

Preliminaries: Inverse Relations

Inverse Relations

- Hypernym and hyponym:
  - Color is the hypernym (r_2) of blue, and blue is the hyponym (r_1) of color
- Husband (r_2) and wife (r_1)

Formally:

r_1 **is inverse to relation** r_2**:** $r_2(x, y) \Rightarrow r_1(y, x)$ if $\forall\, x, y$

Knowledge Graph Embedding

Preliminaries: Composition Relations

Composition Relations

- My mother's husband is my father
- r_1: hasMother, r_2: hasHusband
- r_3: hasFather

Formally:

r_3 is a **composition** of relation r_1 and relation r_2:

$$r_1(x, y) \wedge r_2(y, z) \Rightarrow r_3(x, z) \text{ if } \forall\, x, y, z$$

KG Embedding Method #1: TransE

Embedding space: Each entity and relation as a low-dimensional vector in R^k

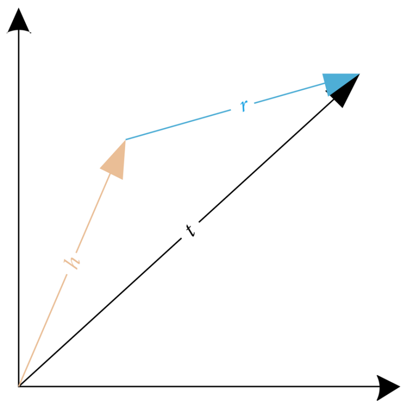
Key idea: Relation r as a translation from the head entity h to the tail entity t

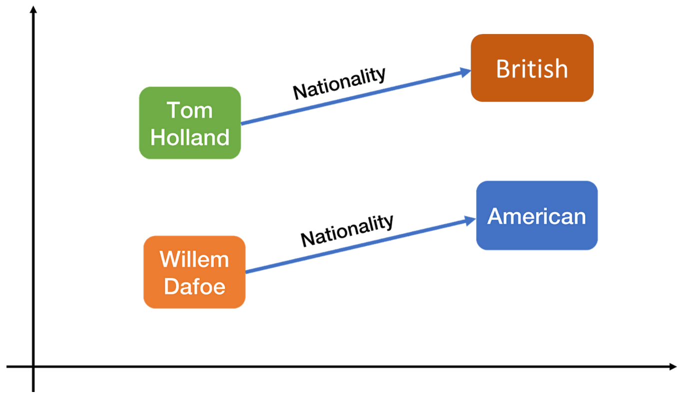
- An ideal/predicted tail entity: $\mathbf{t}_{\mathbf{pred}} = \mathbf{h} + \mathbf{r}$

Score function: $f_{\mathbf{r}}(\mathbf{h}, \mathbf{t}) = -d_r(\mathbf{h}, \mathbf{t}) = -||\mathbf{h} + \mathbf{r} - \mathbf{t}||$

Distance function: $d_r(\mathbf{h}, \mathbf{t}) = ||\mathbf{h} + \mathbf{r} - \mathbf{t}||$

- Triplet: $(\mathrm{h}, \mathrm{r}, \mathrm{t})$
- Embedding vectors: $\mathbf{h}, \mathbf{r}, \mathbf{t}$





Translating Embeddings for Modeling Multi-relational Data

KG Embedding Method #1: TransE

Training Process

For each positive triplet $(h, r, t) \in S$,

- Sample a set of corrupted triplets $(h, r, t') \in S'_{(h,r,t)}$ or $(h', r, t) \in S'_{(h,r,t)}$

Learning: Maximize a margin-based ranking criterion

- Minimize $L = \sum_{(h,r,t) \in S} \sum_{(h',r,t') \in S'_{(h,r,t)}} max(\gamma + d_r(\mathbf{h}, \mathbf{t}) - d_r(\mathbf{h}', \mathbf{t}'), 0)$

margin

- True triplet: $(\mathrm{h}, \mathrm{r}, \mathrm{t})$
- Corrupted triplets:
  - (h’, r, t), (h, r, t’), (h’, r, t’)
- γ: the margin

Translating Embeddings for Modeling Multi-relational Data

KG Embedding Method #1: TransE

Key Properties

Pros

- Can model antisymmetric relations: $\mathbf{h} + \mathbf{r} = \mathbf{t}$, but $\mathbf{t} + \mathbf{r} \neq \mathbf{h}$ if $\mathbf{r} \neq 0$
- Can model inverse relations: $\mathbf{h} + \mathbf{r_1} = \mathbf{t}, \mathbf{t} + \mathbf{r_2} = \mathbf{h}, \mathbf{r_1} = -\mathbf{r_2}$
- Can model composition relations: $\mathbf{r_3} = \mathbf{r_1} + \mathbf{r_2}$

Cons

- Cannot model symmetric relations: $\mathbf{h} + \mathbf{r} = \mathbf{t}, \mathbf{t} + \mathbf{r} = \mathbf{h}$, then $\mathbf{r} = 0$

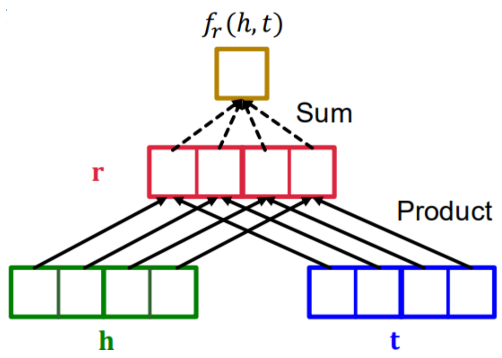
KG Embedding Method #2: DistMult

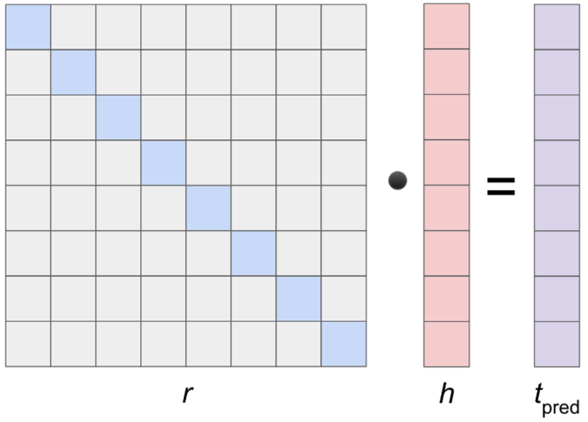
Embedding space: Each entity and relation as a low-dimensional vector in R^k

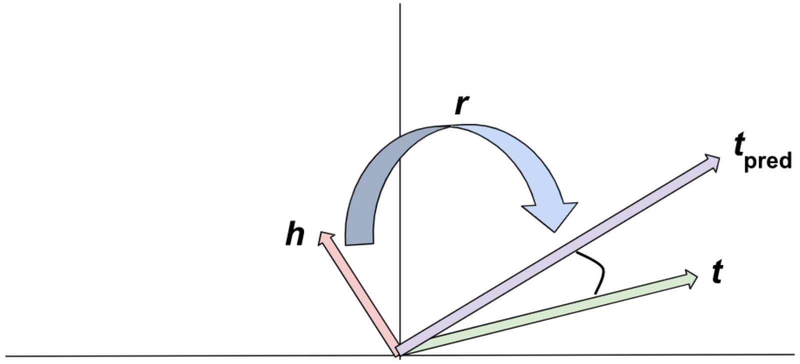
Key idea: Relation r defined as the elementwise weights of the head entity

Score function: $f_{\mathbf{r}}(\mathbf{h},\mathbf{t}) = <\mathbf{h},\mathbf{r},\mathbf{t}> = \sum_{\mathbf{j}} \mathbf{h_j} \cdot \mathbf{r_j} \cdot \mathbf{t_j}$

- Intuition: Can be viewed as a dot product between $\mathbf{h} \cdot \mathbf{r}$ and $\mathbf{t}$







Embedding Entities and Relations for Learning and Inference in Knowledge Bases

KG Embedding Method #2: DistMult

Key Properties

Pros

- Can model symmetric relations: $f_{\mathbf{r}}(\mathbf{h},\mathbf{t}) =< \mathbf{h},\mathbf{r},\mathbf{t} > = \sum_{\mathbf{j}} \mathbf{h}_{\mathbf{j}} \cdot \mathbf{r}_{\mathbf{j}} \cdot \mathbf{t}_{\mathbf{j}} =< \mathbf{t},\mathbf{r},\mathbf{h} > = f_{\mathbf{r}}(\mathbf{t},\mathbf{h})$

Cons

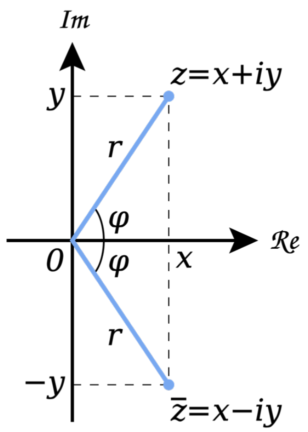
- Cannot model antisymmetric relations: $f_{\mathbf{r}}(\mathbf{h},\mathbf{t})$ and $f_{\mathbf{r}}(\mathbf{t},\mathbf{h})$ are always the same
- Cannot model inverse relations: if $< \mathbf{h},\mathbf{r_1},\mathbf{t} > =< \mathbf{t},\mathbf{r_2},\mathbf{h} >$, it means $\mathbf{r_1} = \mathbf{r_2}$
- Cannot model composition relations: it does not model a bijection mapping from h to t via relation r

KG Embedding Method #3: ComplEx

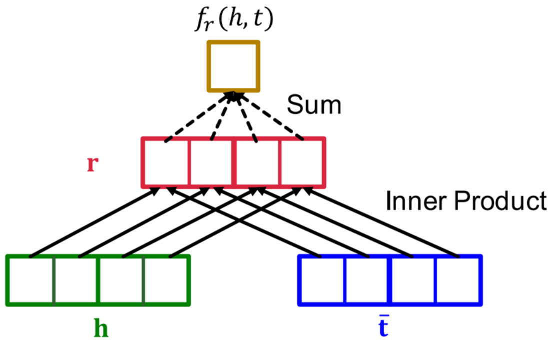
Embedding space: Each entity and relation as a low-dimensional vector in C^k

Key idea: Explore the asymmetry of the Hermitian dot product to accommodate antisymmetry

Score function: $f_{\mathbf{r}}(\mathbf{h}, \mathbf{t}) = \mathbf{Re}(\sum_{\mathbf{j}} \mathbf{h_j} \cdot \mathbf{r_j} \cdot \bar{\mathbf{t}}_{\mathbf{j}})$



Conjugate



Notation & Symbols

- $z = x + iy$: a point in complex space
  - $\mathrm{Re}(z) = x$: the real part
  - $\mathrm{Im}(z) = y$: the imaginary part
  - $i = \sqrt{-1}$
- Hermitian dot product $< \mathbf{h}, \bar{\mathbf{t}} >= \sum_{\mathbf{j}} \mathbf{h_j} \cdot \bar{\mathbf{t}}_{\mathbf{j}}$
  - j: the j-th embedding dimension

KG Embedding Method #3: ComplEx

Key Properties

Pros

- Can model antisymmetric relations: when $\mathbf{Re}(\mathbf{r}) = \mathbf{0}$ (e.g., $\mathbf{r}$ is purely imaginary)
- Can model symmetric relations: when $\mathbf{Im}(\mathbf{r}) = \mathbf{0}$ (e.g., $\mathbf{r}$ is purely real)
- Can model inverse: when $\mathbf{r_1}$ is the conjugate of $\mathbf{r_2}$

Cons

- Cannot model composition relations: it does not model a bijection mapping from h to t via relation r

KG Embedding Method #4: RotatE

Embedding space: Each entity and relation as a low-dimensional vector in C^k

Key idea: Relations modelled as rotations in complex space

- An ideal tail entity: $\mathbf{t} = \mathbf{h} \circ \mathbf{r}$,

Define each relation r as an element-wise rotation from the head entity embedding $\mathbf{h}$ to the tail entity embedding $\mathbf{t}$, i.e.,

$$\mathbf{t} = \mathbf{h} \circ \mathbf{r}, \quad \text{where } |\mathbf{r_j}|=1$$

$\circ$ is the element-wise product. More specifically, we have $\mathbf{t_j} = \mathbf{h_j r_j}$, and

$$\mathbf{r_j} = e^{i\theta_{r,j}}$$

where $\theta_{r,j}$ is the phase angle of r in the j-th dimension

Notation & Symbols

- $i = \sqrt{-1}$
- j: the j-th embedding dimension

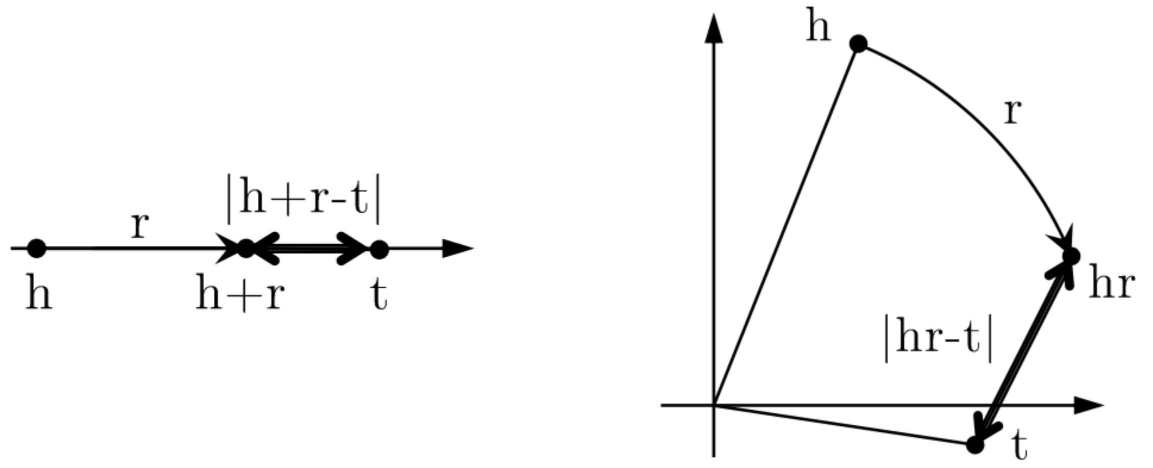
RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space

KG Embedding Method #4: RotatE

Geometric Interpretation

Score function: $f_{\mathbf{r}}(\mathbf{h},\mathbf{t}) = -||\mathbf{h}^{\circ}\,\mathbf{r} - \mathbf{t}||$

Define the distance function of RotatE as $d_r(\mathbf{h},\mathbf{t}) = ||\mathbf{h}^{\circ}\,\mathbf{r} - \mathbf{t}||$



(a) TransE models $\mathbf{r}$ as translation in real line.

(b) RotatE models $\mathbf{r}$ as rotation in complex plane.

KG Embedding Method #4: RotatE

Modeling the Relation Patterns

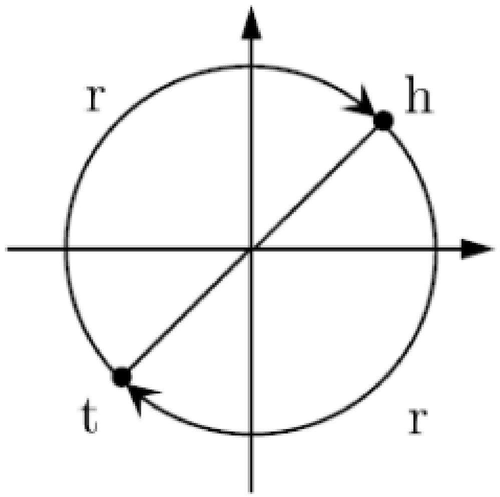
A relation r is symmetric if and only if $\mathbf{r}_\mathrm{j} = \pm 1$, i.e.,

$$\theta_{r,j} = 0 \; or \; \pi$$

(h, r, t) and (t, r, h)

An example on the space of C

$$\mathbf{r}_\mathrm{j} = -1 \text{ or } \theta_{r,j} = \pi$$



KG Embedding Method #4: RotatE

Modeling the Relation Patterns

Relation r is antisymmetric if and only if $\mathbf{r} \circ \mathbf{r} \neq \mathbf{1}$

Two relations r_1 and r_2 are inverse if and only if $\mathbf{r}_2 = \bar{\mathbf{r}}_1$, i.e.,

$$\theta_{2,j} = -\theta_{1,j}$$

A relation $\mathbf{r_3} = \mathrm{e}^{i\theta_3}$ is a composition of two relations $\mathbf{r_1} = \mathrm{e}^{i\theta_1}$ and $\mathbf{r_2} = \mathrm{e}^{i\theta_2}$ if and only if $\mathbf{r_3} = \mathbf{r_1} \circ \mathbf{r_2}$, i.e.,

$$\theta_3 = \theta_1 + \theta_2$$

KG Embedding Method #4: RotatE

Optimization

Negative sampling loss

- Make the score of true triplet as large as possible, but the score of false one as small as possible

$$L = -\log \sigma\big(\underbrace{\gamma - d_r(\mathbf{h}, \mathbf{t})}_{\text{Positive triplet}}\big) - \sum_{j=1}^{k} \frac{1}{k} \log \sigma\big(\underbrace{d_r(\mathbf{h}'_\mathbf{j}, \mathbf{t}'_\mathbf{j})}_{\text{Negative triplet}} - \underbrace{\gamma}_{\text{margin}}\big)$$

γ is a fixed margin, σ is the sigmoid function, and $(\mathbf{h}'_\mathbf{j}, \mathbf{r}, \mathbf{t}'_\mathbf{j})$ is the j-th negative triplet

- Controls the minimum separation between the scores of positive and negative triplets.
- Prevent model from over-fitting

KG Embedding Method #4: RotatE

Self-adversarial Negative Sampling

Traditionally, the negative samples are drawn in a uniform way

- Inefficient as training goes on since many samples are obviously false
- Does not provide useful information

A self-adversarial negative sampling

- Sample negative triplets according to the current embedding model
- Starts from easier samples to more and more difficult samples
- Curriculum Learning

$$p\left(h_j', r, t_j' \middle| \{(h_k, r_k, t_k)\}\right) = \frac{\exp \alpha f_r(h_j', t_j')}{\sum_k \exp \alpha f_r(h_k', t_k')}$$

α is the temperature of sampling. $f_r(h_j', t_j')$ measures the salience of the triplet

KG Embedding Method #4: RotatE

The Final Objective

Instead of sampling, treating the sampling probabilities as weights

- p is used to weight each negative sample

$$L = -\log\sigma\big(\gamma - \underbrace{d_r(\mathbf{h},\mathbf{t})}_{\text{Positive triplet}}\big) - \sum_{j=1}^{n} p(h'_j, r, t'_j)\log\sigma\big(\underbrace{d_r\big(\mathbf{h}'_\mathbf{j},\mathbf{t}'_\mathbf{j}\big)}_{\text{Negative triplet}} - \underbrace{\gamma}_{\text{margin}}\big)$$

Intuitions:

- the smaller the $d_r\big(\mathbf{h}'_\mathbf{j},\mathbf{t}'_\mathbf{j}\big)$ is,
- the more likely the triplet is true,
- the harder the negative triplet,
- the higher the weight *p*

Notation & Symbols

- *j*: the j-th negative sample
- *n*: the number of negative samples
- γ: the margin

RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space

Empirical Comparisons of KG Embedding

Observation

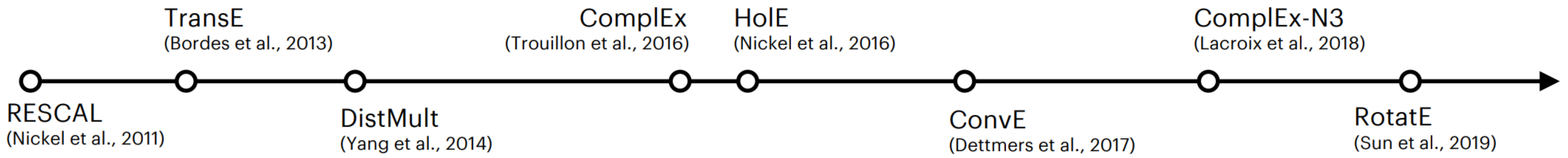
	FB15k-237					WN18RR				
	MR	MRR	H@1	H@3	H@10	MR	MRR	H@1	H@3	H@10
TransE	357	.294	-	-	.465	3384	.226	-	-	.501
DistMult	254	.241	.155	.263	.419	5110	.43	.39	.44	.49
ComplEx	339	.247	.158	.275	.428	5261	.44	.41	.46	.51
ConvE	244	.325	.237	.356	.501	4187	.43	.40	.44	.52
pRotatE	178	.328	.230	.365	.524	**2923**	.462	.417	.479	.552
RotatE	**177**	**.338**	**.241**	**.375**	**.533**	3340	**.476**	**.428**	**.492**	**.571**

RotatE performs the best, on both datasets, on different metrics

KG Embedding Method: Summary

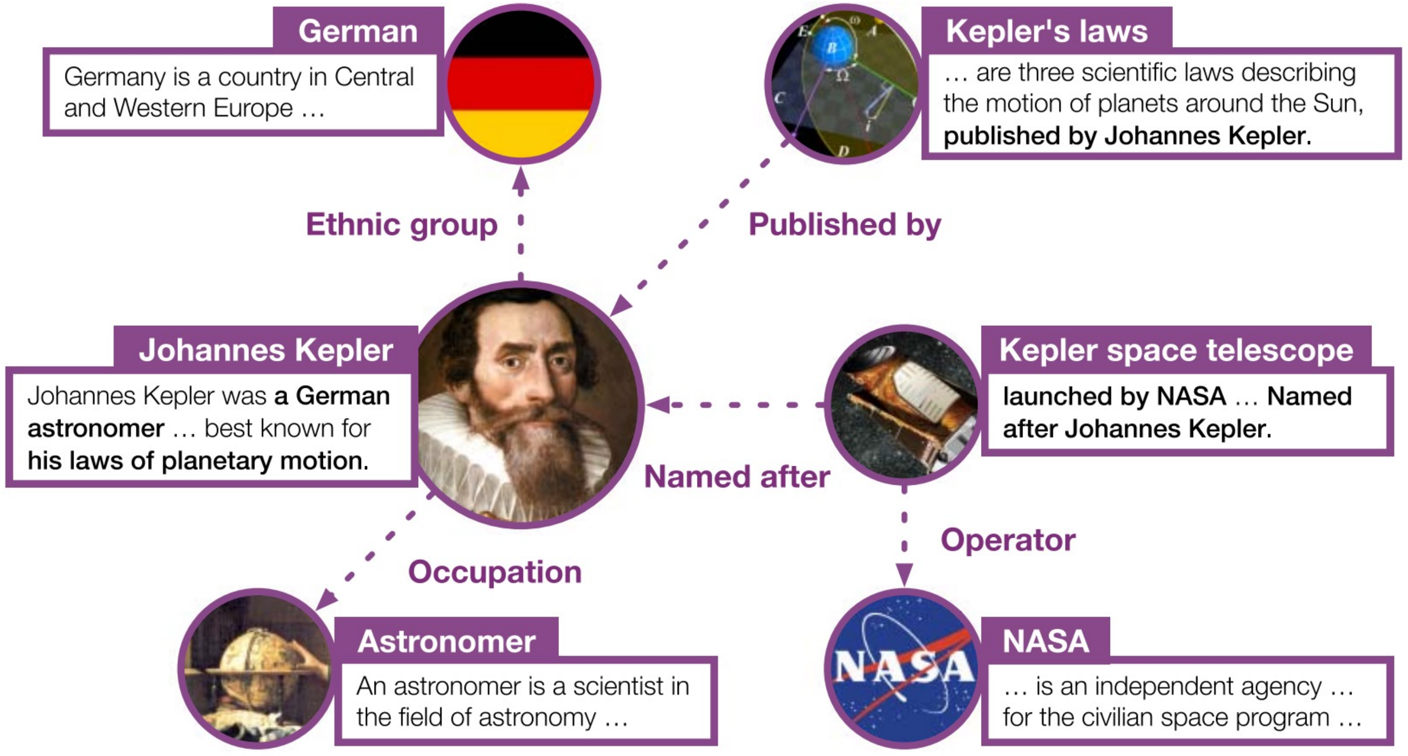
Model	Score Function	Symmetry	Antisymmetry	Inversion	Composition
TransE	$-\|\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|\|$	✗	✓	✓	✓
DistMult	$<\mathbf{h},\mathbf{r},\mathbf{t}>$	✓	✗	✗	✗
ComplEx	$\mathbf{Re}(<\mathbf{h},\mathbf{r},\bar{\mathbf{t}}>)$	✓	✓	✓	✗
RotatE	$-\|\|\mathbf{h}\circ\mathbf{r}-\mathbf{t}\|\|$	✓	✓	✓	✓

(Some) KGE models in recent literature:



Advancing KG tasks with text data

An example of a KG with entity descriptions



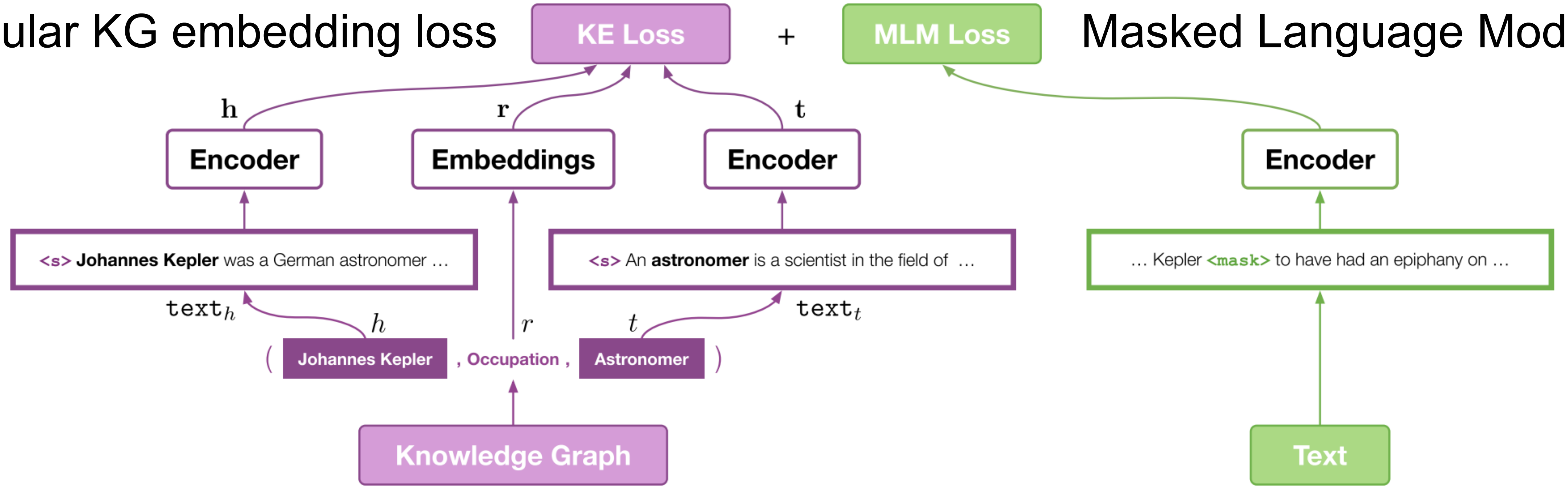
- Descriptions contain abundant information about entities
- Descriptions can help to represent the relational facts between them

KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation

Advancing KG tasks with text data

Train encoder with both KG and NLP losses: KEPLER

Regular KG embedding loss



Masked Language Modeling

KEPLER encodes text descriptions as entity embeddings, and train the shared encoder via MLM loss

KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation

Graph Learning Enhanced by External Knowledge

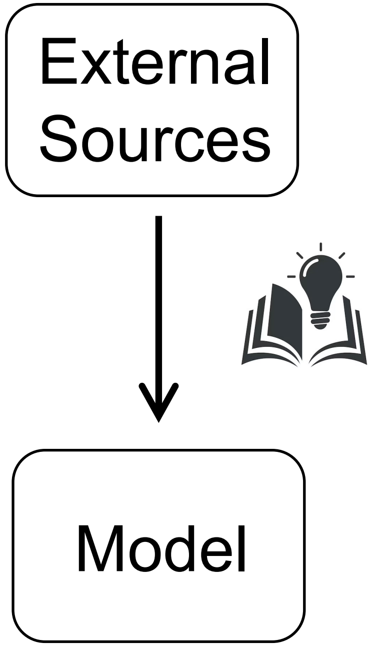
Chapter summary

Graph learning on text-rich graphs

- Graph learning on textual-node graphs
- Graph learning on textual-edge graphs

Graph learning on knowledge graphs

- Knowledge Graph Embedding
- Advancing KG tasks with text data



Tutorial Outline

- Preliminaries and Foundations
- Graph Learning Enhanced by Knowledge from Data
- Graph Learning Enhanced by Knowledge from Models
- Graph Learning Enhanced by Knowledge from Humans and Domains
- Graph Learning Enhanced by Knowledge from External Sources
- Knowledge-enhanced Graph Learning for Real-world Applications
- Summary and Future Directions

Applications

1) Knowledge-enhanced Graph Learning for Recommendation

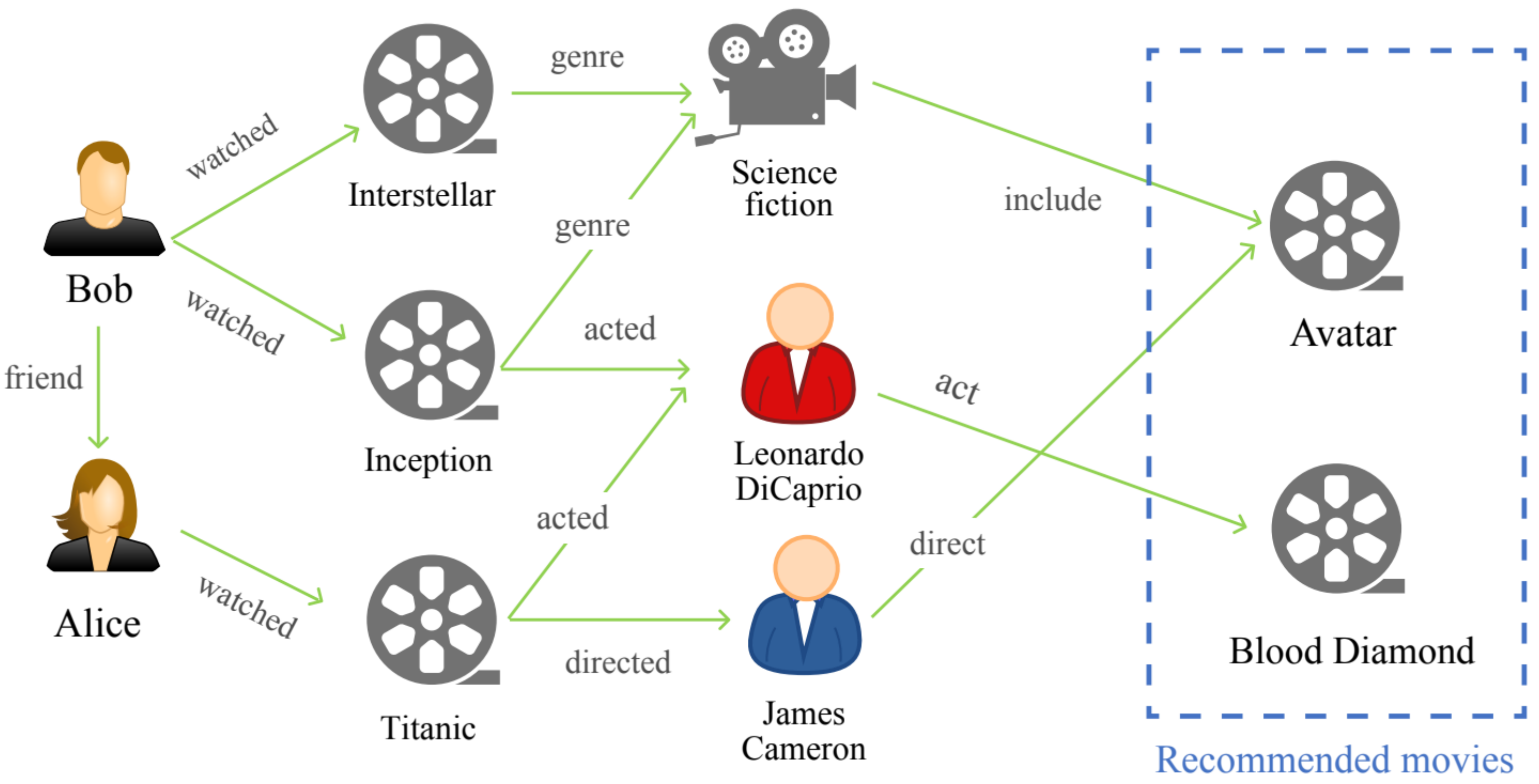
- Path-based recommendation methods
- Propagation-based recommendation methods

2) Knowledge-enhanced Graph Learning for Natural Language Processing (NLP)

- Natural Language Understanding
- Commonsense Reasoning
- Advancing LLMs with Knowledge

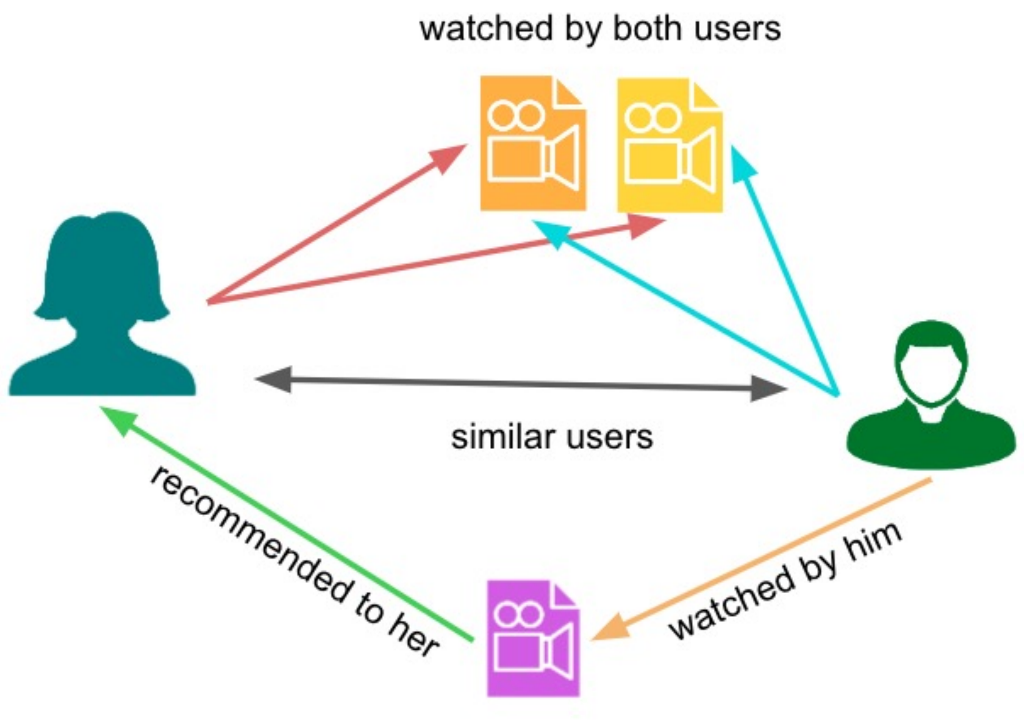
Knowledge-enhanced Graph Learning for Recommendation

An example KG for recommendation



Knowledge-enhanced Graph Learning for Recommendation

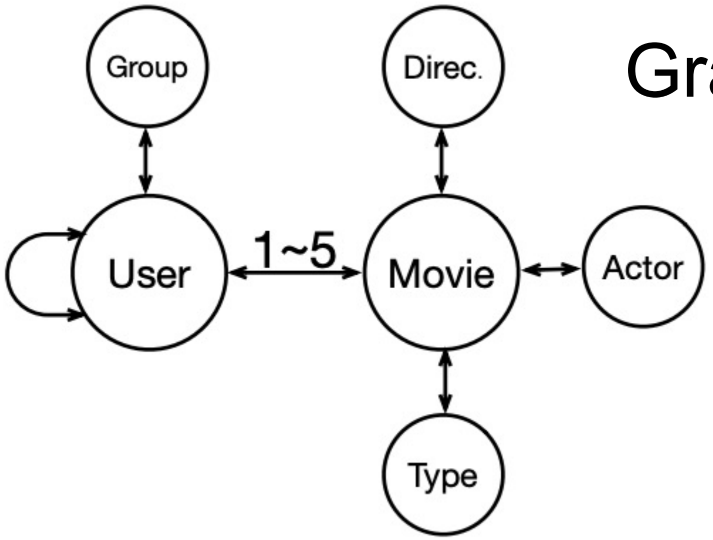
Preliminary: traditional collaborative filtering



If a person A has the same preference as a person B on a product, A is more likely to have B's preference on a different product than that of a randomly chosen person

Knowledge-enhanced Graph Learning for Recommendation

Path-based recommendation methods: SemRec



Graph schema of Douban data

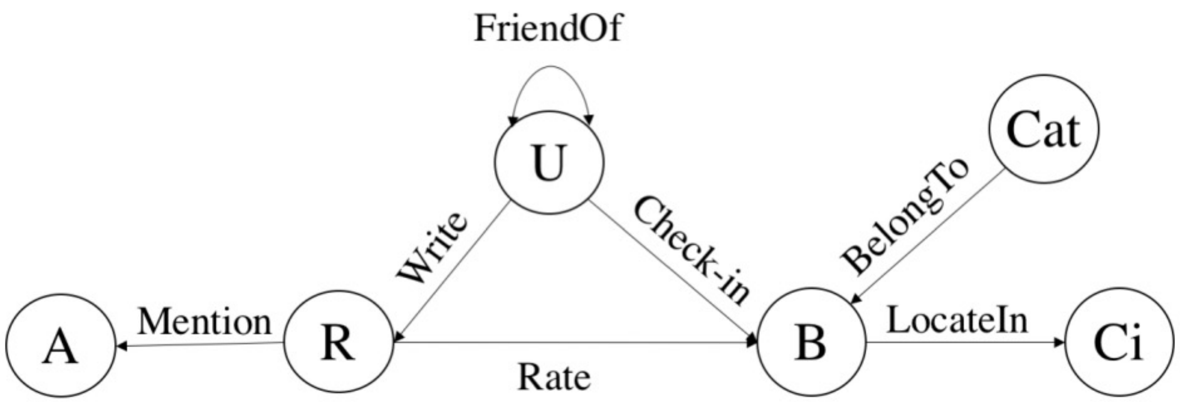
- Metapaths captures different semantic knowledge
- Users appeared in same metapath can have similar ratings

Meta Path	Semantic Meaning	Recommendation Model
UU	friends of the target user	Social recommendation
UGU	users in the same group of the target user	Member recommendation
UMU	users who view the same movies with the target user	Collaborative recommendation
UMTMU	users who view the movies having the same types with that of the target user	Content recommendation

Semantic Path based Personalized Recommendation on Weighted Heterogeneous Information Networks

Problems of SemRec

Metapaths might fail to capture the complex structure and all the semantic knowledge



Graph schema of Yelp data

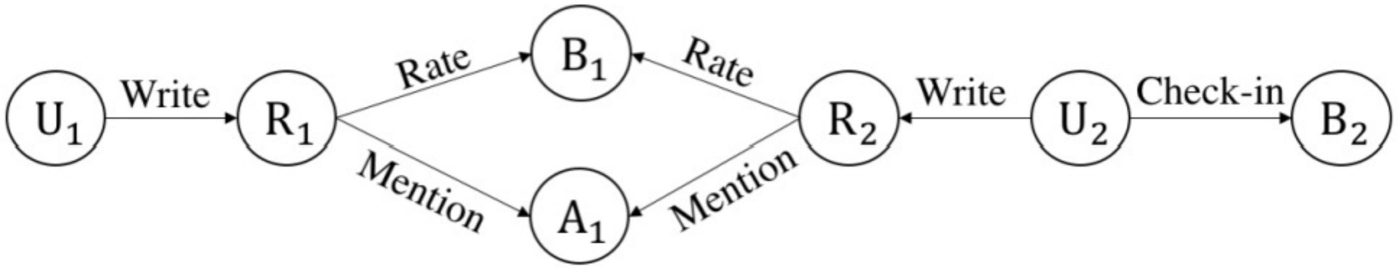
A: aspect extracted from reviews
R: reviews
U: users
B: business
Cat: category of item
Ci: city.

Knowledge-enhanced Graph Learning for Recommendation

Address problems of SemRec: FMG

Metapaths might fail to capture the complex structure and all the semantic knowledge

What if $R_1$and R_2 give the same rating for a business B_1 and mention the same aspect A_1?



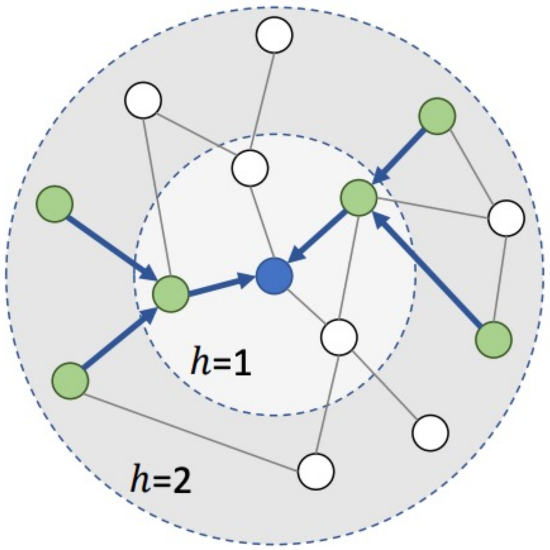
Use metagraph, a directed acyclic graph that contains more semantic knowledge

Path-based recommendation methods: Limitations

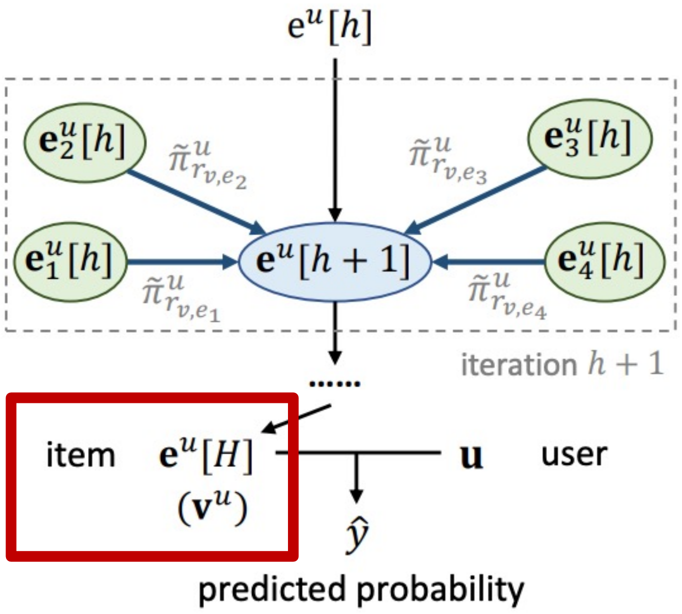
- The choice of metapath and metagraph need to be defined manually
- Metapath and metagraph varying across datasets
- It might require labor-intensive feature engineering to extract relevant paths and structures

Knowledge-enhanced Graph Learning for Recommendation

Knowledge graph-aware recommendation methods: KGCN



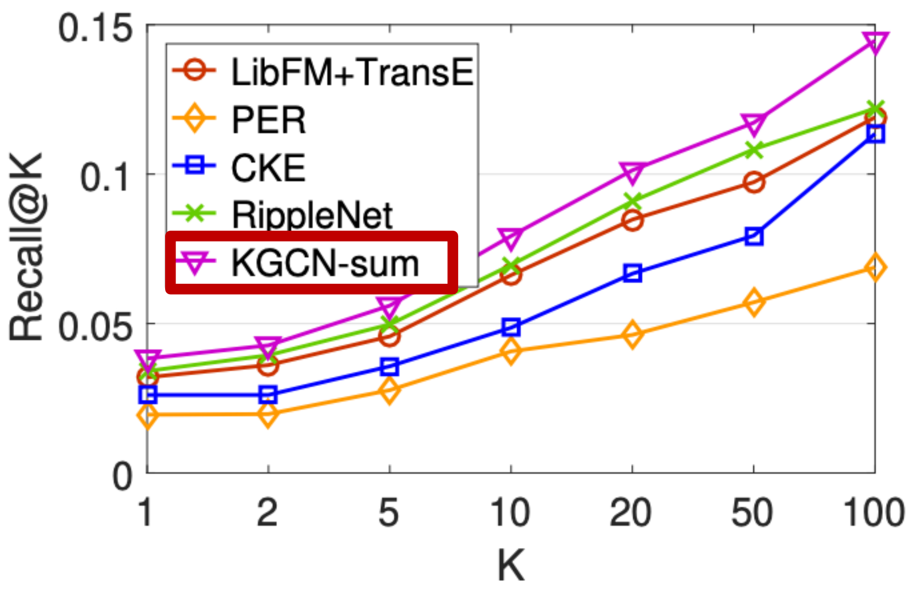
A two-layer receptive field



The framework of KGCN

- Learning item representations by aggregation on KG
- Use other items to enhance the representation of target item

Knowledge graph-aware recommendation methods: KGCN

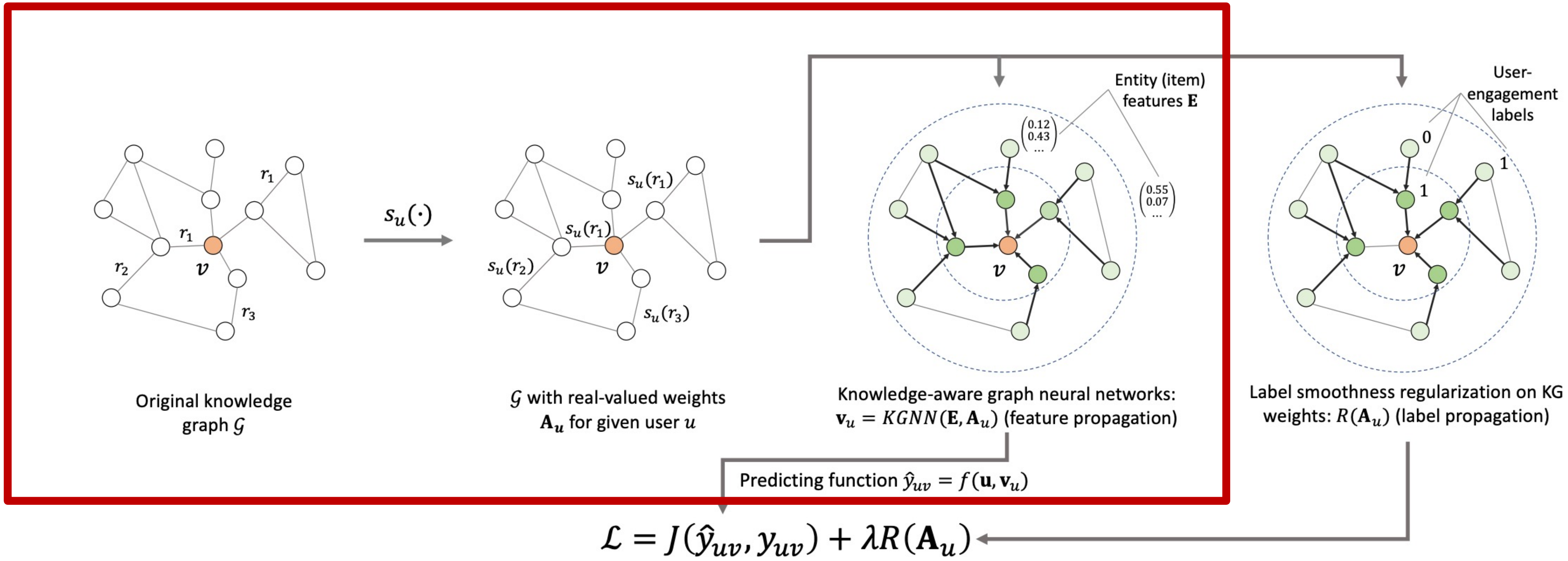


The results of top-K recommendation on Book-Crossing dataset

KGCN achieves the best results, demonstrating the benefit of incorporating knowledge from KG

Knowledge-enhanced Graph Learning for Recommendation

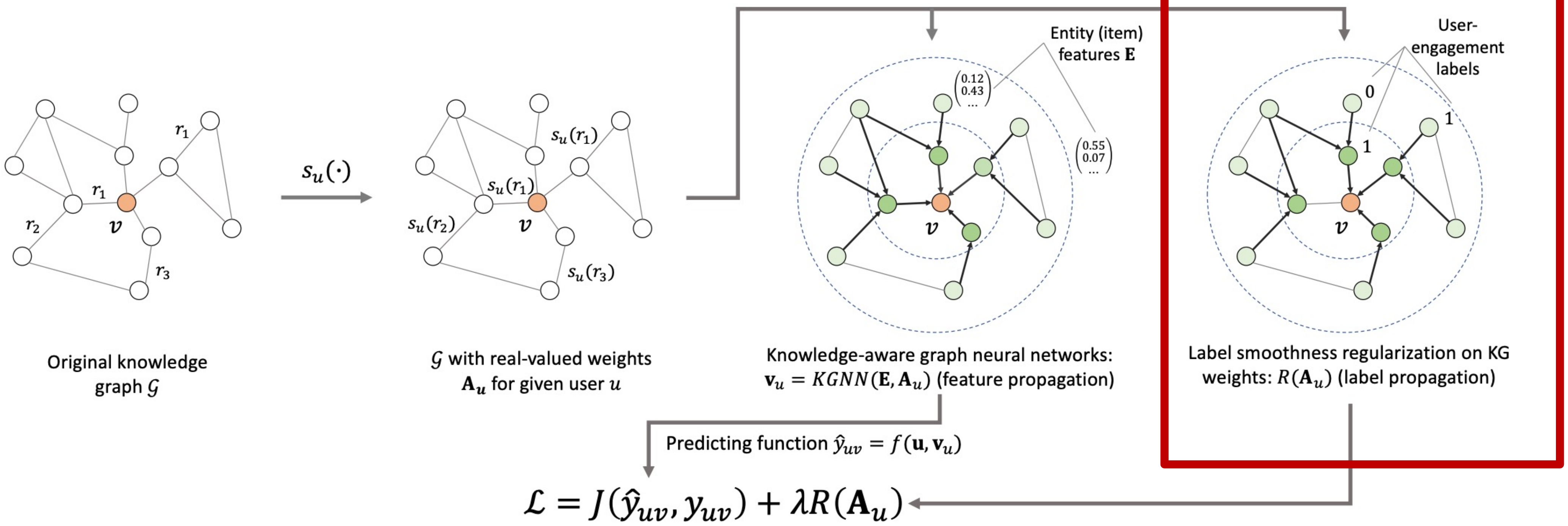
Knowledge graph-aware recommendation methods: KGNN-LS



KGNN-LS is Similar to KGCN. First extract a graph of interest to user u, then use GNNs to learn on this user-specific graph

Knowledge-enhanced Graph Learning for Recommendation

Knowledge graph-aware recommendation methods: KGNN-LS

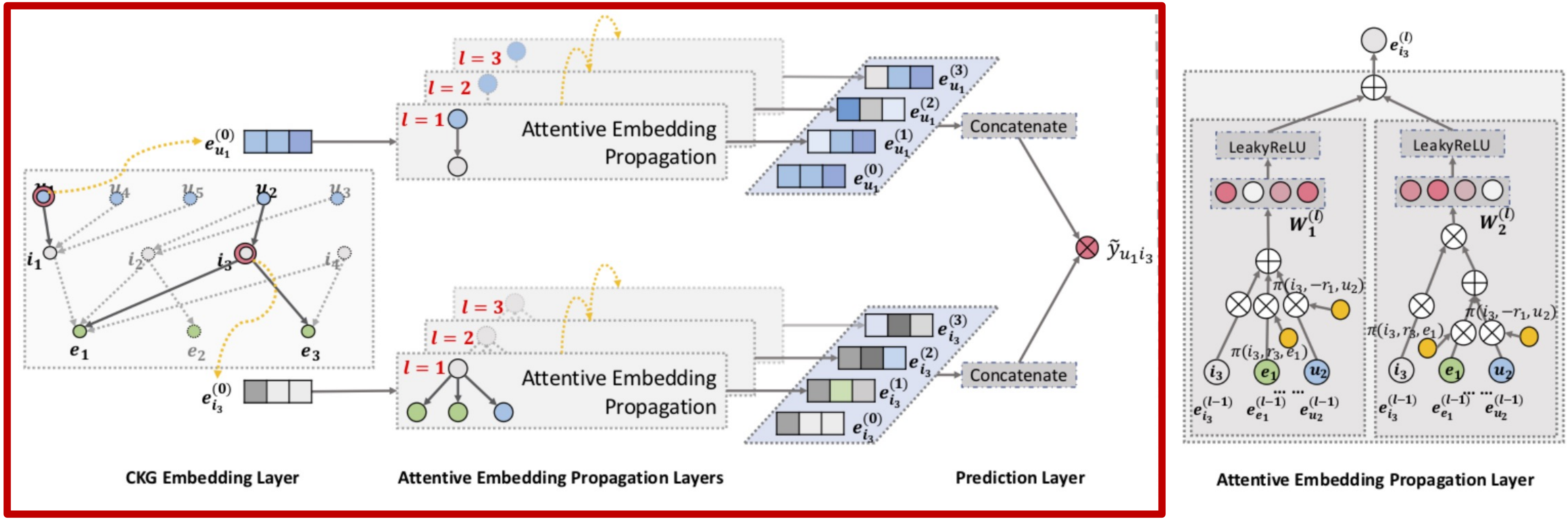


One step further, KGNN-LS utilizes label smoothness: adjacent items in KG are likely to have similar user preferences

Knowledge-aware Graph Neural Networks with Label Smoothness Regularization for Recommender Systems

Knowledge-enhanced Graph Learning for Recommendation

Knowledge graph-aware recommendation methods: KGAT

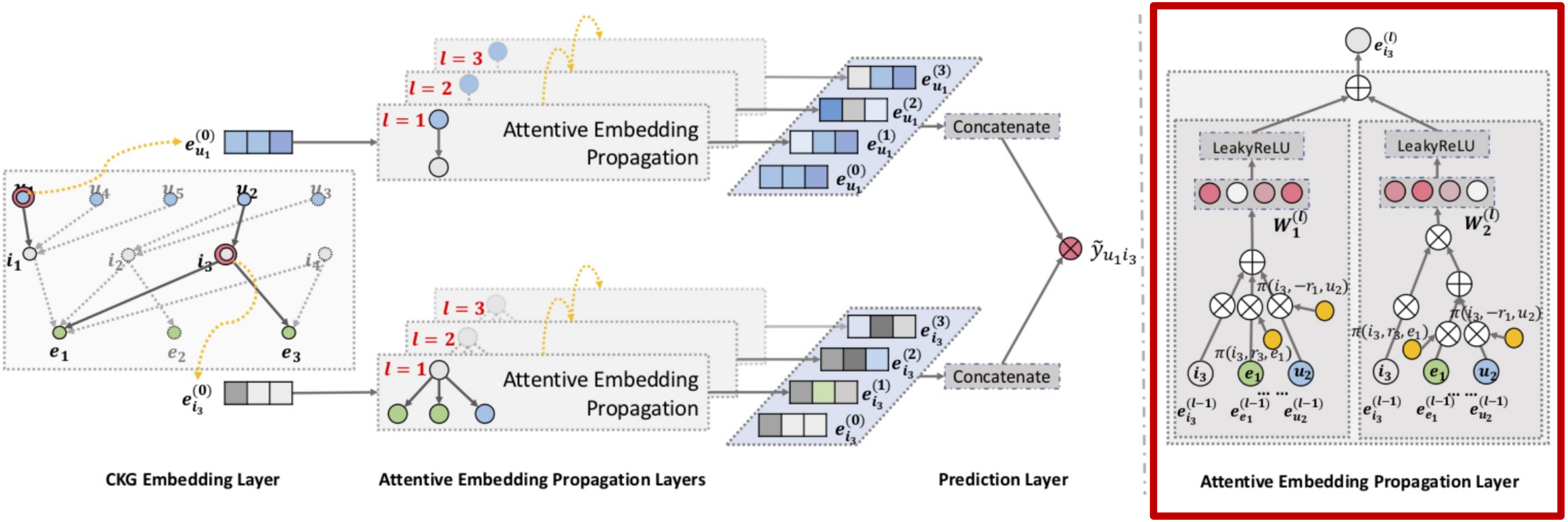


Utilize both item-item connections from KG and user-item direct interaction to learn user and item representations

KGAT: Knowledge Graph Attention Network for Recommendation

Knowledge-enhanced Graph Learning for Recommendation

Knowledge graph-aware recommendation methods: KGAT



Aggregate neighbor information with attention mechanism: reveal the importance of high-order connections

KGAT: Knowledge Graph Attention Network for Recommendation

Knowledge graph-aware recommendation methods: KGAT

Recommendation Performance Comparison

	Amazon-Book		Last-FM		Yelp2018	
	recall	ndcg	recall	ndcg	recall	ndcg
FM	0.1345	0.0886	0.0778	0.1181	0.0627	0.0768
NFM	**0.1366**	0.0913	**0.0829**	0.1214	0.0660	0.0810
CKE	0.1343	0.0885	0.0736	0.1184	0.0657	0.0805
CFKG	0.1142	0.0770	0.0723	0.1143	0.0522	0.0644
MCRec	0.1113	0.0783	-	-	-	-
RippleNet	0.1336	0.0910	0.0791	0.1238	**0.0664**	**0.0822**
GC-MC	0.1316	0.0874	0.0818	**0.1253**	0.0659	0.0790
KGAT	**0.1489***	**0.1006***	**0.0870***	**0.1325***	**0.0712***	**0.0867***
%Improv.	8.95%	10.05%	4.93%	5.77%	7.18%	5.54%

KGAT achieves the best performance

KGAT: Knowledge Graph Attention Network for Recommendation

Applications

1) Knowledge-enhanced Graph Learning for Recommendation

- Path-based recommendation methods
- Propagation-based recommendation methods

2) Knowledge-enhanced Graph Learning for Natural Language Processing (NLP)

- Natural Language Understanding
- Commonsense Reasoning
- Advancing LLMs with Knowledge

Knowledge-enhanced Graph Learning for NLP

- A language model (LM) learns **how to express**

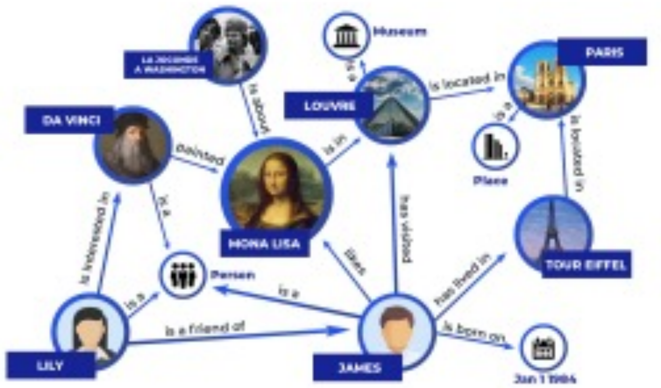
  I go school to to want. ✗

  I want to go to school. ✓

- Knowledge indicates **what to express**

  Q: Where is the painting **Mona Lisa**?

  A: It is in **Louvre, Paris.**

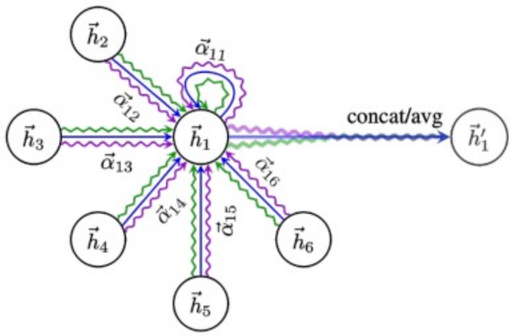


Knowledge-enhanced Graph Learning for NLP

To integrate knowledge into Language Models

- Step 1: Ground language into related knowledge
  - ➢ String matching, NER, Entity linking, information retrieval
  - ➢ Identify concepts and relations in the knowledge source
- Step 2: Represent knowledge
  - ➢ Concept descriptions, Graph embeddings
- Step 3: Fuse knowledge representation into language model
  - ➢ Concatenate concept descriptions into input
  - ➢ Append embeddings into input embeddings

The **pen** is on the **desk**.




\+ Input

Knowledge-enhanced Graph Learning for NLP

What are the knowledge sources for NLP

WIKIDATA

WIKTIONARY
the open content based dictionary

Wikipedia-based knowledge

UMLS Unified Medical Language System

AMiner

Domain-specific knowledge

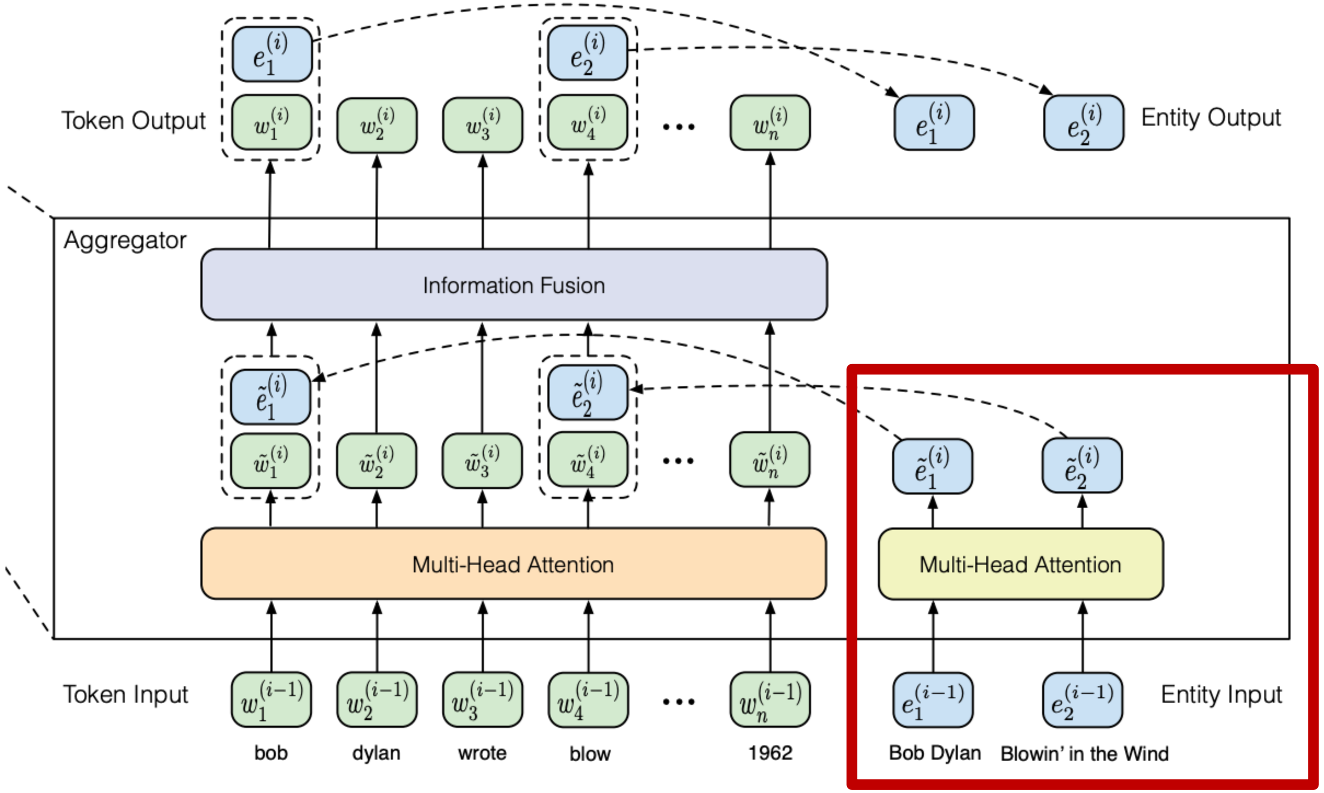
ConceptNet

OPEN MIND Common Sense

Commonsense knowledge

Knowledge-enhanced Graph Learning for NLP

Natural Language Understanding: ERNIE



Bob Dylan wrote ***Blowin' in the Wind*** in 1962

- Use TransE to compute entity embeddings from Wikidata
- Concatenate token input and entity embeddings for learning

ERNIE: Enhanced Language Representation with Informative Entities

Knowledge-enhanced Graph Learning for NLP

Natural Language Understanding: ERNIE

Model	Acc.	Macro	Micro
NFGEC (Attentive)	54.53	74.76	71.58
NFGEC (LSTM)	55.60	75.15	71.73
BERT	52.04	75.16	71.63
ERNIE	**57.19**	**76.51**	**73.39**

Entity typing task on the FIGER dataset

Model	FewRel			TACRED		
	P	R	F1	P	R	F1
CNN	69.51	69.64	69.35	70.30	54.20	61.20
PA-LSTM	-	-	-	65.70	64.50	65.10
C-GCN	-	-	-	69.90	63.30	66.40
BERT	85.05	85.11	84.89	67.23	64.81	66.00
ERNIE	88.49	88.44	**88.32**	69.97	66.08	**67.97**

Relation classification task

Model	MNLI-(m/mm) 392k	QQP 363k	QNLI 104k	SST-2 67k
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	-	93.5
ERNIE	84.0/83.2	71.2	91.3	93.5

Model	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k
$\text{BERT}_{\text{BASE}}$	52.1	85.8	88.9	66.4
ERNIE	52.3	83.2	88.2	68.8

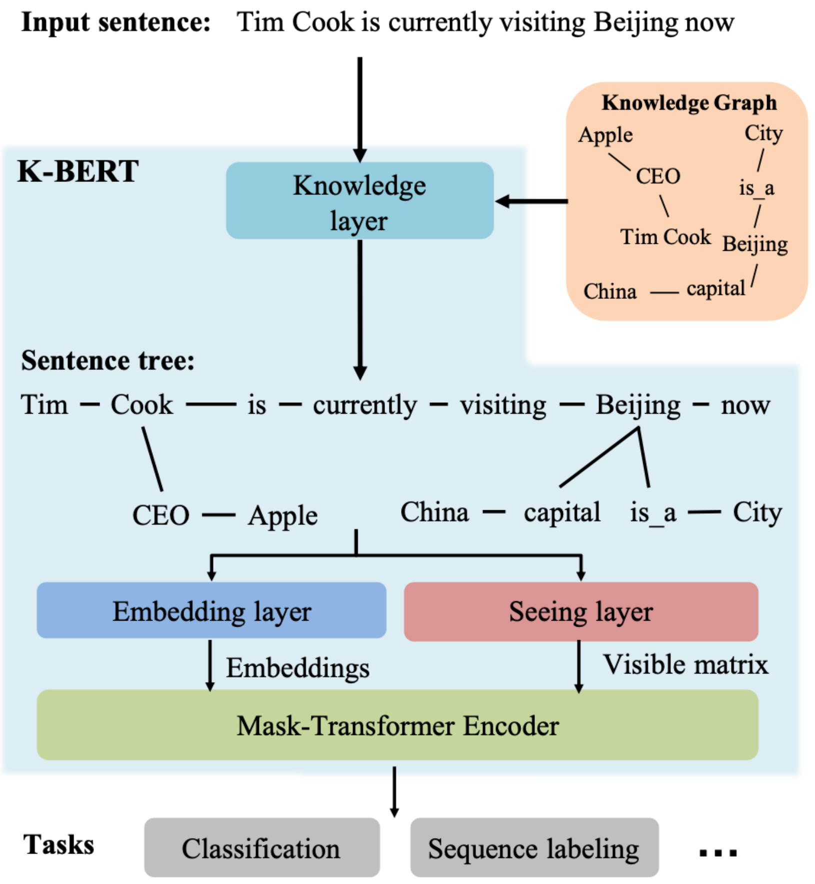
General Language Understanding Evaluation (GLUE) benchmark

ERNIE outperforms BERT across tasks

ERNIE: Enhanced Language Representation with Informative Entities

Knowledge-enhanced Graph Learning for NLP

Natural Language Understanding: K-BERT



- K-BERT injects relevant triples from the KG and transform the original sentence into a knowledge-rich sentence tree for encoding
- K-BERT is equipped with an editable KG, which can be adapted to its application domain. For example, a medical KG can be used to grant the K-BERT with medical knowledge

K-BERT: Enabling Language Representation with Knowledge Graph

Knowledge-enhanced Graph Learning for NLP

Natural Language Understanding: K-BERT

Results on different domain-specific tasks

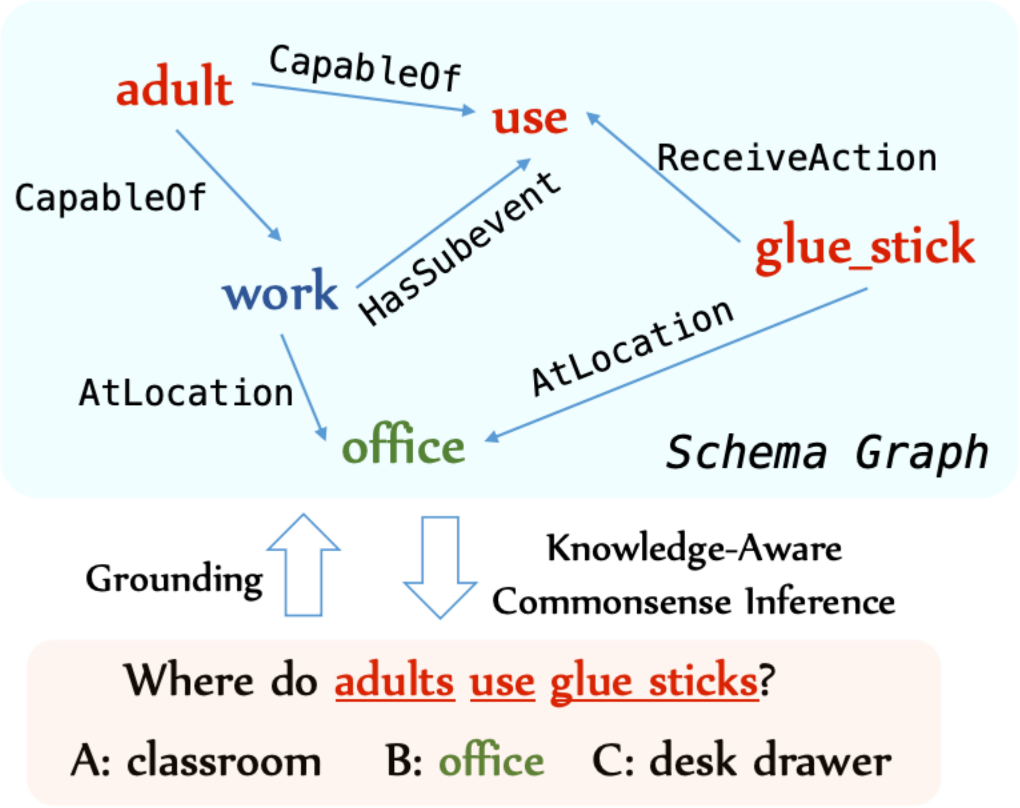
Models\Datasets	Finance_Q&A			Law_Q&A			Finance_NER			Medicine_NER		
	$P.$	$R.$	$F1$	$P.$	$R.$	$F1$	$P.$	$R.$	$F1$	$P.$	$R.$	$F1$
Pre-trained on WikiZh by Google.												
Google BERT	81.9	86.0	83.9	83.1	90.1	86.4	84.8	87.4	86.1	91.9	93.1	92.5
K-BERT (HowNet)	83.3	84.4	83.9	83.7	91.2	87.3	86.3	89.0	**87.6**	93.2	93.3	93.3
K-BERT (CN-DBpedia)	81.5	88.6	**84.9**	82.1	93.8	**87.5**	86.1	88.7	87.4	93.9	93.8	93.8
K-BERT (MedicalKG)	-	-	-	-	-	-	-	-	-	94.0	94.4	**94.2**

K-BERT benefits from the domain knowledge in KGs and performs well across different domain-specific tasks

K-BERT: Enabling Language Representation with Knowledge Graph

Knowledge-enhanced Graph Learning for NLP

Commonsense reasoning: KagNet

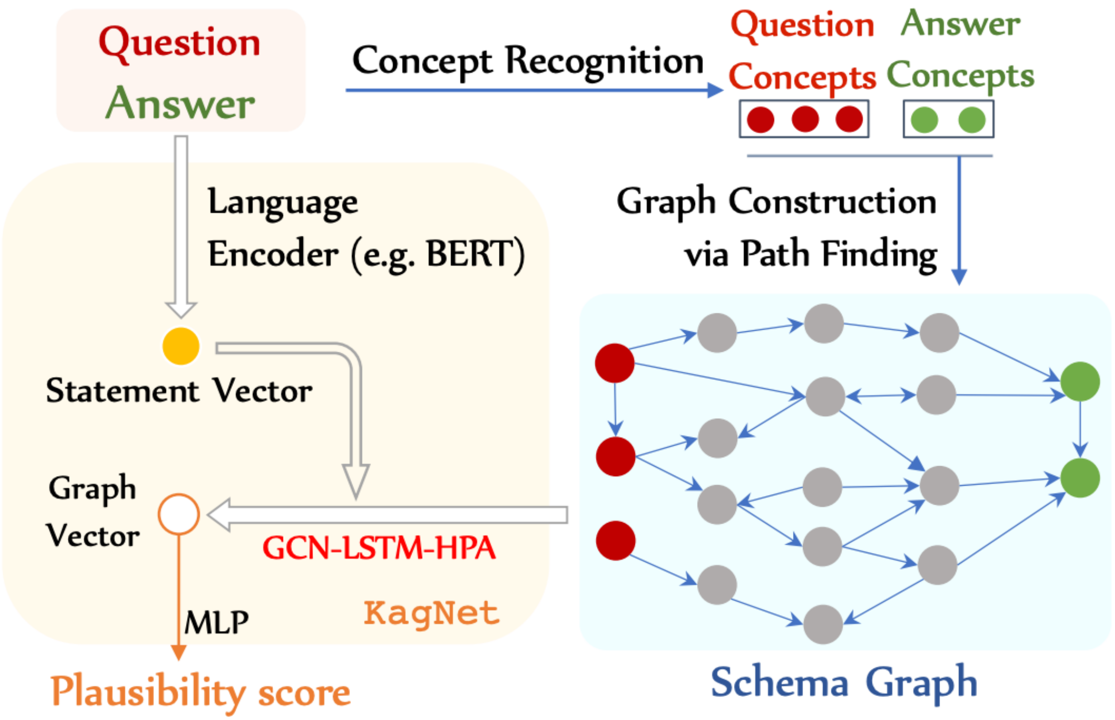


An example of using external commonsense knowledge for inference in natural language commonsense questions

KagNet: Knowledge-Aware Graph Networks for Commonsense Reasoning

Knowledge-enhanced Graph Learning for NLP

Commonsense reasoning: KagNet



- For each pair of question and answer candidate, KagNet retrieves a graph from external knowledge graphs (e.g. ConceptNet)
- The retrieved graph contains relevant knowledge for determining the plausibility of a given answer choice

Knowledge-enhanced Graph Learning for NLP

Commonsense reasoning: KagNet

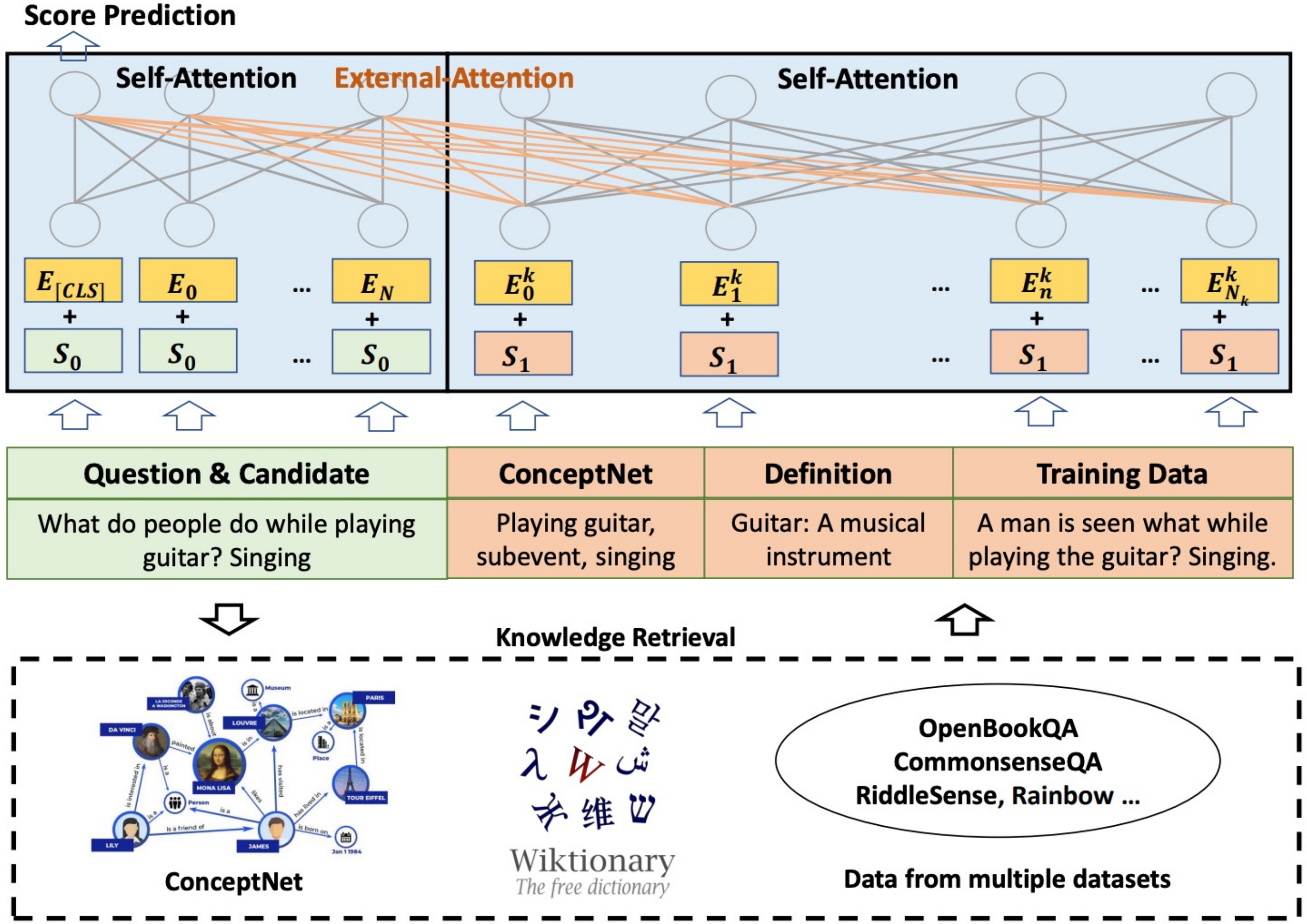
Model	10(%) of IHtrain IHdev-Acc.(%)	10(%) of IHtrain IHtest-Acc.(%)	50(%) of IHtrain IHdev-Acc.(%)	50(%) of IHtrain IHtest-Acc.(%)	100(%) of IHtrain IHdev-Acc.(%)	100(%) of IHtrain IHtest-Acc.(%)
Random guess	20.0	20.0	20.0	20.0	20.0	20.0
GPT-FINETUNING	27.55	26.51	32.46	31.28	47.35	45.58
GPT-**KAGNET**	28.13	**26.98**	33.72	**32.33**	48.95	**46.79**
BERT-BASE-FINETUNING	30.11	29.78	38.66	36.83	53.48	53.26
BERT-BASE-**KAGNET**	31.05	**30.94**	40.32	**39.01**	55.57	**56.19**
BERT-LARGE-FINETUNING	35.71	32.88	55.45	49.88	60.61	55.84
BERT-LARGE-**KAGNET**	36.82	**33.91**	58.73	**51.13**	62.35	**57.16**
Human Performance	-	*88.9*	-	*88.9*	-	*88.9*

Using KagNet outperforms fine-tuning the language models themselves

KagNet: Knowledge-Aware Graph Networks for Commonsense Reasoning

Knowledge-enhanced Graph Learning for NLP

Commonsense reasoning: KEAR



Related knowledge is retrieved from external sources, e.g., knowledge graph, dictionary and training data, using the input as the key and then integrated with the input

Human Parity on CommonsenseQA: Augmenting Self-Attention with External Attention

Knowledge-enhanced Graph Learning for NLP

Commonsense reasoning: KEAR

Method	E-l+VAT	D-xxl	DV3-l
Base	82.1	83.8	84.6
+ KG	85.2	86.4	86.7
+ Dictionary	83.8	84.0	85.1
+ Training data	84.0	86.4	87.1

E-l+VAT: ELECTRA-large with VAT
D-xxl: DeBERTa-xxlarge
DV3-l: DeBERTaV3-large

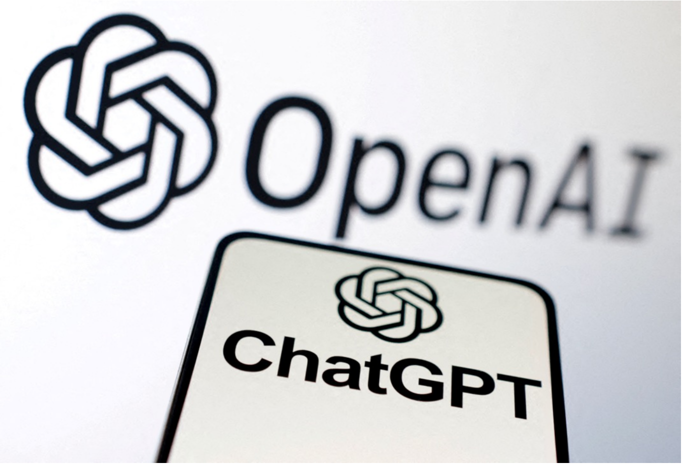
Results of applying external attention to different knowledge sources

- These knowledge sources bring gains in commonsense reasoning accuracy across all base encoder models
- Generally, knowledge graphs contribute the most

Advancing LLMs with Knowledge

What are LLMs (Large Language Models)?

A type of machine learning model that can perform a variety of language tasks.



Advancing LLMs with Knowledge

Why LLMs Need External Knowledge?

⚠ Limitations

The problem of hallucination → May occasionally generate incorrect information (from OpenAI)

YI **You**

Were there fossil fuels in the ground when humans evolved?

~300 Million Years Ago

~300 Thousand Years Ago

Options:

(a) This was only created by humans

(b) Humans predate fossil fuel formation

(c) Significant supplies accumulated prior → Correct Answer

(d) None of these

Advancing LLMs with Knowledge

Why LLMs Need External Knowledge?

The problem of hallucination

You

Were there fossil fuels in the ground when humans evolved?

Options:
(a) This was only created by humans
(b) Humans predate fossil fuel formation
(c) Significant supplies accumulated prior
(d) None of these

Wrong Answer

ChatGPT

(b) Humans predate fossil fuel formation

Explanations that Make No Sense

Fossil fuels, such as coal, oil, and natural gas, are formed from the remains of ancient plants and microorganisms that lived millions of years ago. The process of fossil fuel formation takes a very long time, and it began long after the evolution of early humans. Therefore, humans predate the formation of fossil fuels, making option (b) the correct answer.

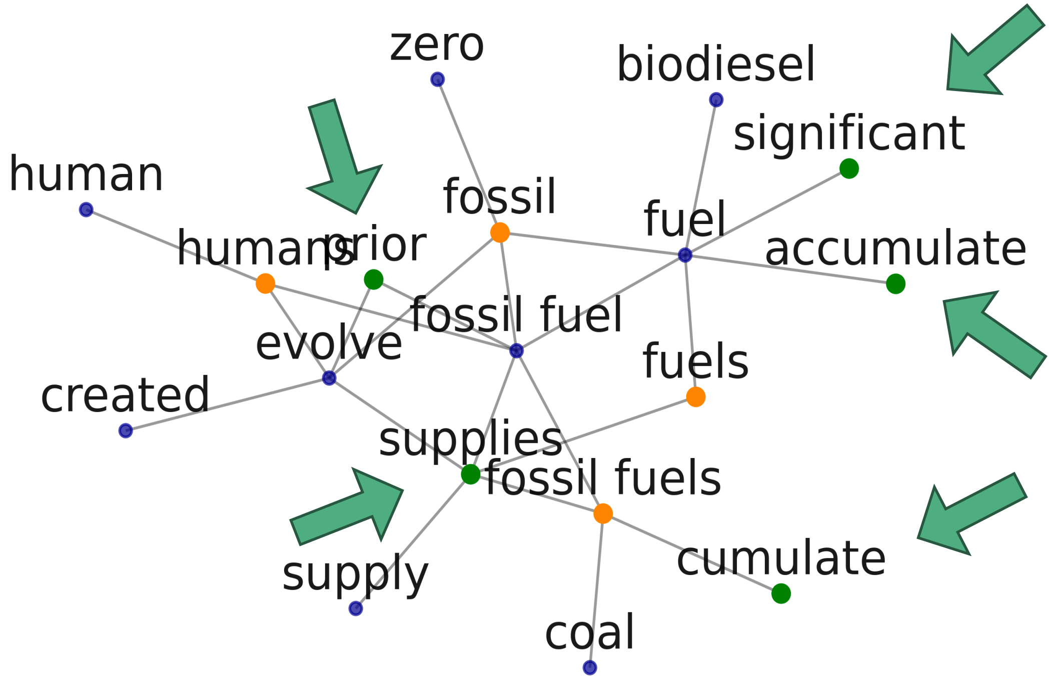
Advancing LLMs with Knowledge

How External Knowledge Assists?

Provides Factual Information

Were there fossil fuels in the ground when humans evolved?

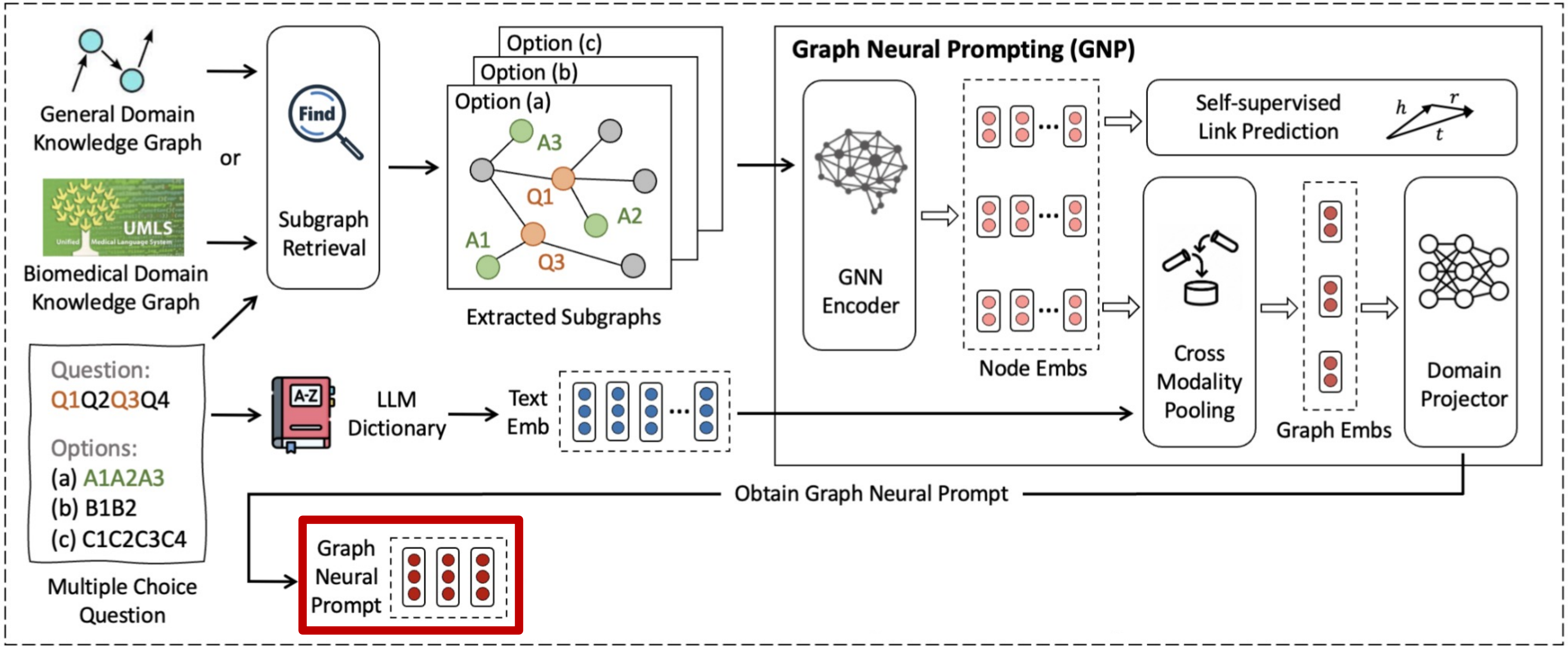
(c) Significant supplies accumulated prior



Retrieved Knowledge for Question from ConceptNet

Advancing LLMs with Knowledge

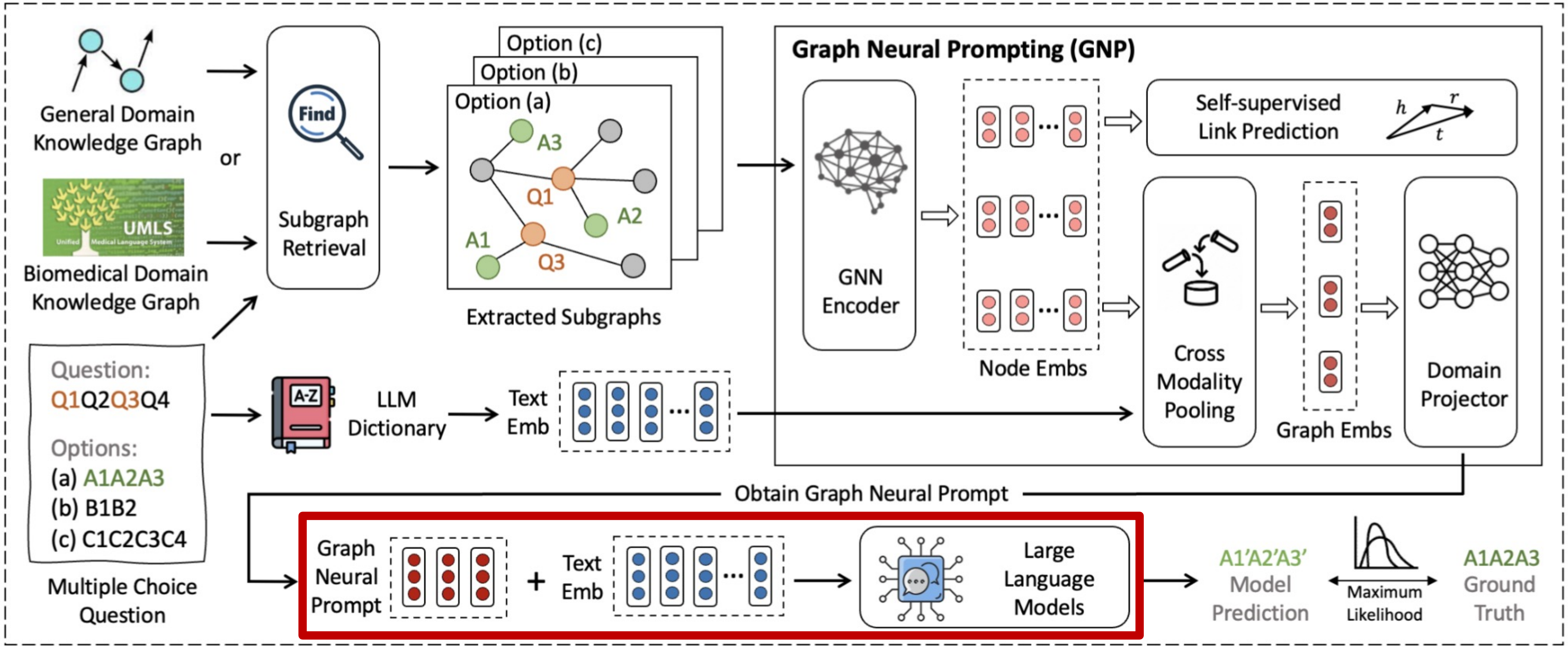
Graph Neural Prompting (GNP)



Extracted Knowledge

Advancing LLMs with Knowledge

Graph Neural Prompting (GNP)



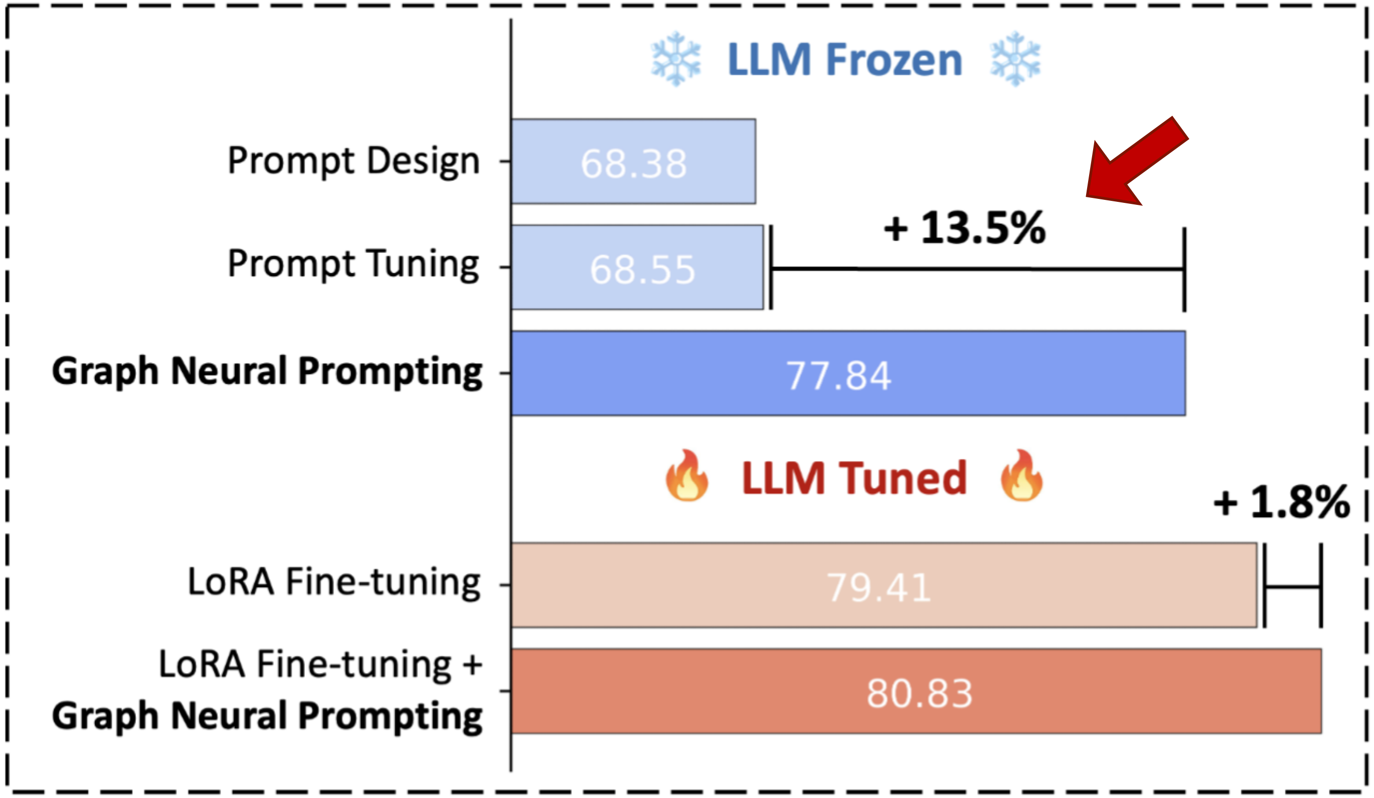
Extracted Knowledge ⟶ Real-life Applications

Advancing LLMs with Knowledge

Graph Neural Prompting (GNP)

w/o Knowledge

w/ Knowledge



Significant Improvement

Results Averaged Across 4 Commonsense and 2 Biomedical Datasets

Applications

Chapter summary

1) Knowledge-enhanced Graph Learning for Recommendation

- Path-based recommendation methods
- Propagation-based recommendation methods

2) Knowledge-enhanced Graph Learning for Natural Language Processing (NLP)

- Natural Language Understanding
- Commonsense Reasoning
- Advancing LLMs with Knowledge

Tutorial Outline

- Preliminaries and Foundations
- Graph Learning Enhanced by Knowledge from Data
- Graph Learning Enhanced by Knowledge from Models
- Graph Learning Enhanced by Knowledge from Humans and Domains
- Graph Learning Enhanced by Knowledge from External Sources
- Knowledge-enhanced Graph Learning for Real-world Applications
- Summary and Future Directions

Summary: Knowledge-enhanced Graph Learning

Implicit Knowledge

Knowledge from Data

Data —Knowledge→ Model

Data Knowledge

Knowledge from Models

Teacher Model —Knowledge→ Student Model

Learned Knowledge

Explicit Knowledge

Knowledge from Humans and Domains

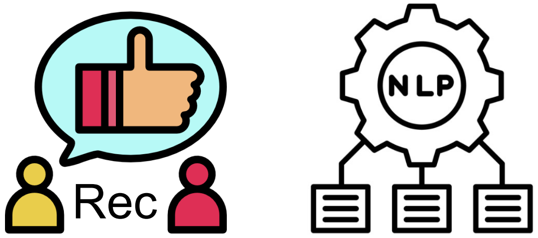
Human/ Domain —Knowledge→ Model

Domain Knowledge

Knowledge from External Sources

External Sources —Knowledge→ Model

External Knowledge



Real-world Applications

Future Directions

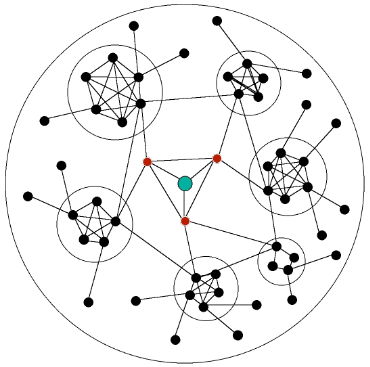
Summary: Knowledge-enhanced Graph Learning

Knowledge from Data

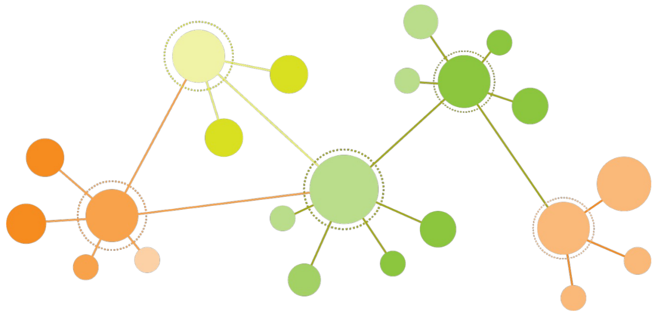
Data —Knowledge→ Model

Data Knowledge

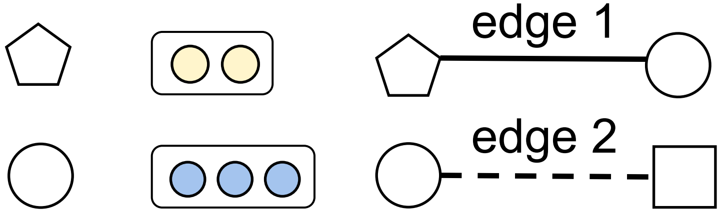
Knowledge from data



Local communities



Node positions



Heterogeneous Information

	p1	p2	p3
p1	1	1	0
p2	1	1	1
p3	0	1	1

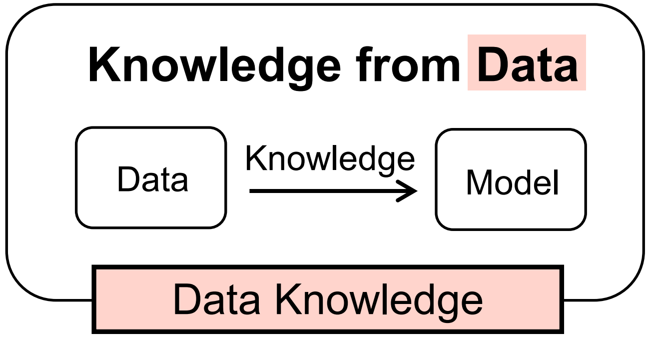
Co-author

	p1	p2	p3
p1	1	1	0
p2	1	1	0
p3	0	0	1

Co-venue

Higher-order Semantics

Summary: Knowledge-enhanced Graph Learning

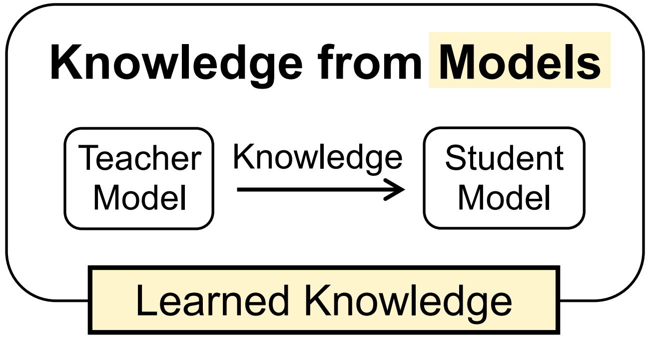


Knowledge can be obtained from

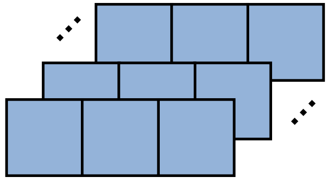
1) **single-instance** level perception
   - Node sampling for node positions (P-GNN)

2) **multiple-instance** level perception
   - Path sampling for positional and semantic information (HGMAE)
   - Subgraph sampling for community information (SEAL, WalkPool)

Summary: Knowledge-enhanced Graph Learning



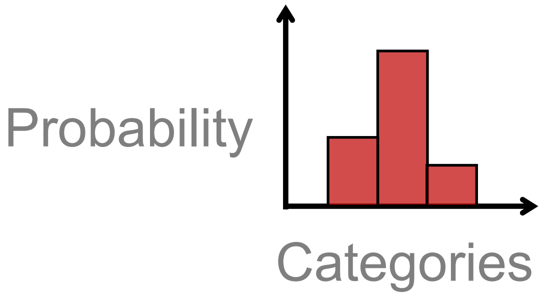
Knowledge from models



Learned Embeddings

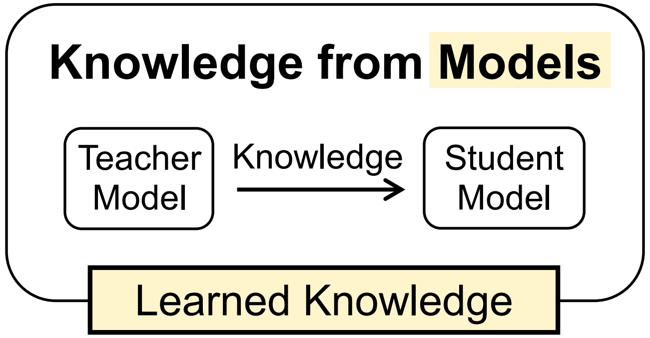
	p1	p2	p3
p1	1	.3	.7
p2	.3	1	.2
p3	.7	.2	1

Data Similarity



Logits

Summary: Knowledge-enhanced Graph Learning



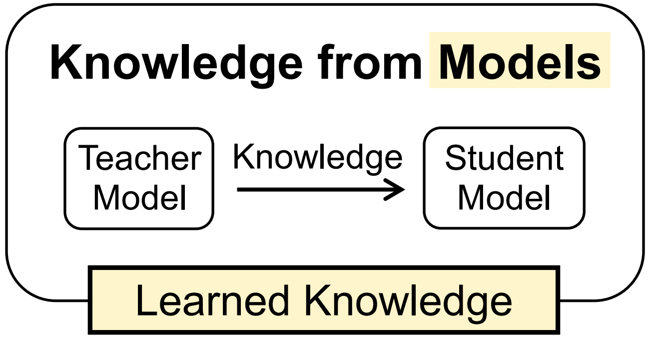
How to obtain knowledge from

- Logits (TinyGNN, BGNN)
- Embeddings (GraphAKD)
- Structures (LSP, G-CRD)

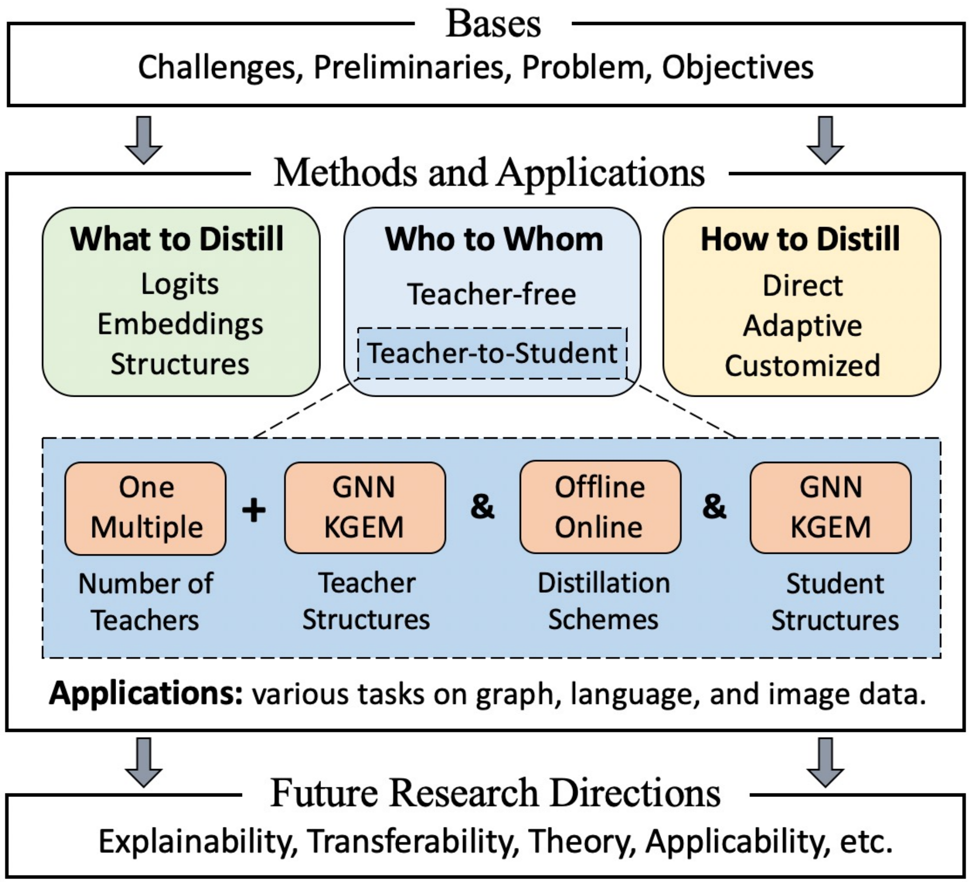
A recent learning strategy

- distilling an MLP to replace GNNs (GLNN, NOSMOG)

Summary: Knowledge-enhanced Graph Learning

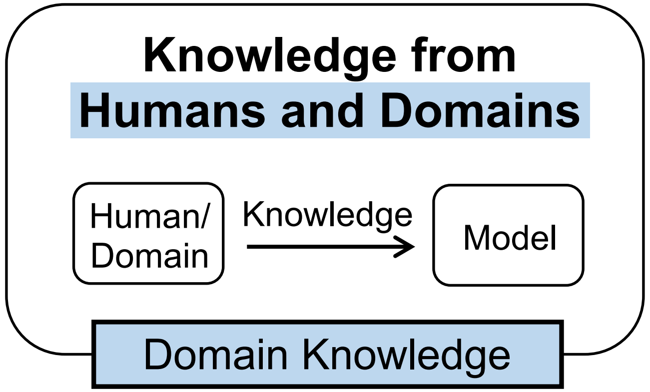


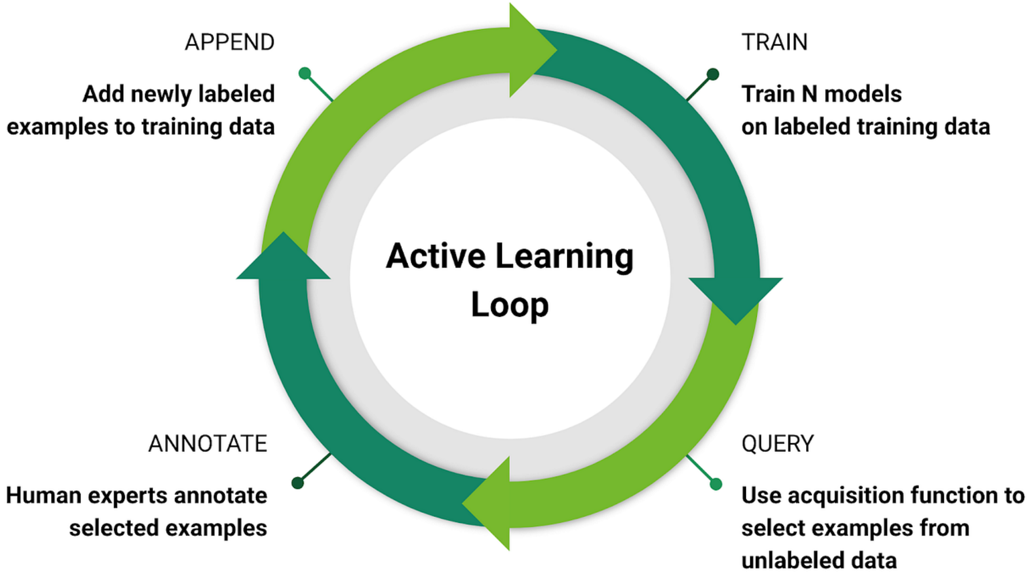
More details can be found in this survey



Knowledge Distillation on Graphs: A Survey

Summary: Knowledge-enhanced Graph Learning

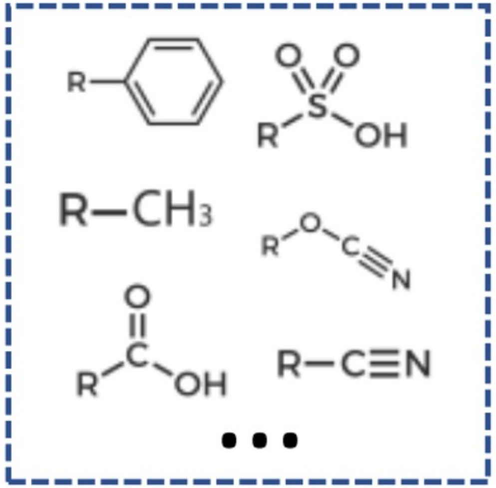




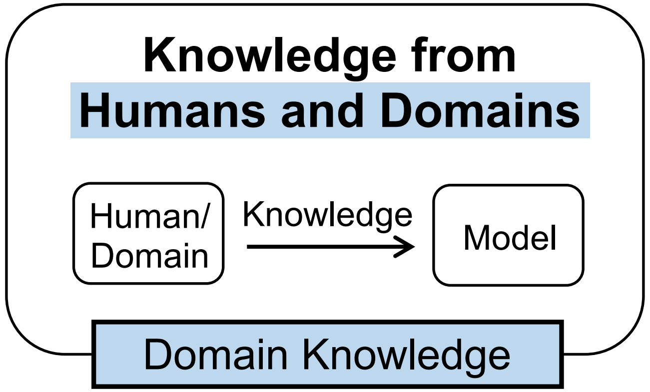
Graph active learning with human knowledge

Domain knowledge such as using motif to represent functional groups in chemistry

Semantic motifs from domain knowledge



Summary: Knowledge-enhanced Graph Learning



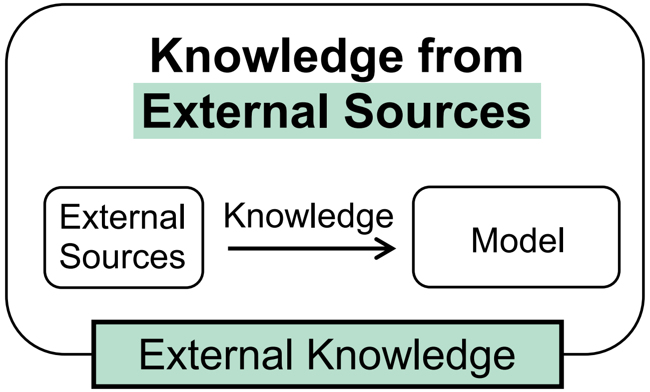
Graph learning enhanced by human feedback

- Graph active learning with human knowledge (ALG, IGP, OWGAL)

Graph learning enhanced by domain knowledge

- Chemistry domain knowledge for molecular property prediction (GROVER, MGSSL)

Summary: Knowledge-enhanced Graph Learning



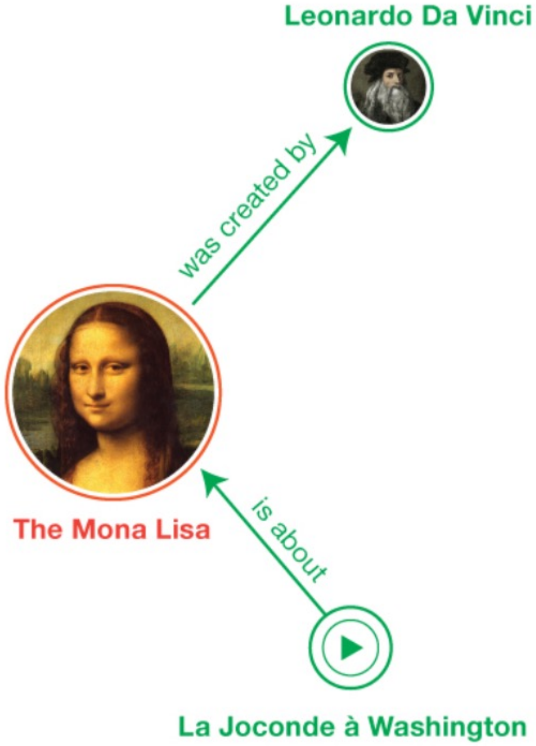
Knowledge from external sources

Text

Wikipedia

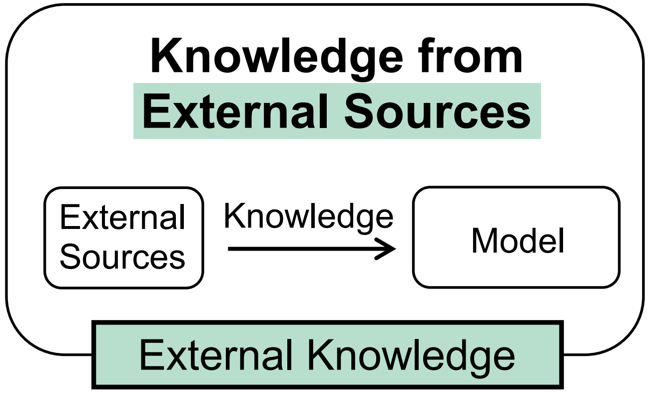
Image

Video



Knowledge triplets

Summary: Knowledge-enhanced Graph Learning



Graph learning on text-rich graphs

- Graph learning on textual-node graphs (PATTON, Heterformer)
- Graph learning on textual-edge graphs (Edgeformers)

Graph learning on knowledge graphs

- Knowledge Graph Embedding (TransE, DistMult, ComplEx, RotatE)
- Advancing KG tasks with text data (KEPLER)

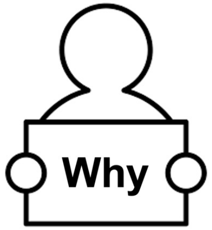
Summary: Knowledge-enhanced Graph Learning

What are the benefits of utilizing knowledge

- Reduce reliance on massive data and intricate model
- Performance ↑

- Trustworthiness ↑

Robustness

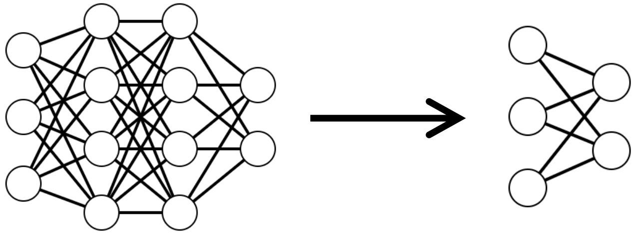


Explainability

- Efficiency ↑

NO LABELS

Less labeled data



Smaller/simpler model

Future Directions

1) How to obtain high-quality knowledge?

Knowledge sources can contain several issues such as Noises, biases, imbalances, missing or incorrect knowledge

Therefore, several research questions arise:

- How to detect and mitigate these issues?
- How to extract high-quality knowledge given these issues?
- How to verify the accuracy and authenticity of the knowledge?
- How to construct knowledge bases without these issues?

Future Directions

2) How to Integrating knowledge from different sources?

Existing works mainly consider leveraging knowledge from one source, while ignoring the fact that various sources can provide different knowledge

Therefore, several research questions arise:

- How to identify the right knowledge sources?
- How to prioritize and extract knowledge from different sources?
- How to combine knowledge with different modalities or formats?
- How to integrate knowledge in a complementary manner?

Future Directions

3) How to handle new knowledge?

New knowledge emerges incrementally and continuously in different time periods

Therefore, several research questions arise:

- How to collect, measure, and evaluate new knowledge?
- How to dynamically update new knowledge into knowledge base?
- How to forget old and irrelevant knowledge?
- How to balance and encode new knowledge?

Thanks for listening!

Tutorial Website:

yijuntian.com/tutorial

Yijun is looking for job opportunities