Graph Pre-Training: From GPT-GNN to GraphMAE to GCC

Yuxiao Dong
Knowledge Engineering Group
(KEG)
CS, Tsinghua University

https://keg.cs.tsinghua.edu.cn/yuxiao

Joint Work with

Jiezhong Qiu, Ziniu Hu, Zhenyu Hou, Wenzheng Feng, Xiao Liu

Yukuo Cen, Weihua Hu, Jie Zhang, Chenhui Zhang, Yuyang Xie

Hao Ma, Wenjian Yu, Yizhou Sun, Jure Leskovec, Jie Tang

… …

Jiezhong Qiu

Ziniu Hu

Zhenyu Hou

Papers & code & data at https://keg.cs.tsinghua.edu.cn/yuxiao/

Graph Pre-Training

HGT (WWW’20)
- billion-scale GNN
- most cited WWW’20
- used in MSFT

GPT-GNN (KDD’20)
- autoreg pre-train
- used in MSFT
- used in FB

GraphMAE (KDD’22)
- masked pre-train
- most cited in KDD’22

GraphMAE2 (WWW’23)
- masked pre-train
- +4% age pred in gaming

GCC (KDD’20)
- contrastive pre-train
- top cited in KDD’20

Structural Embedding

metapath2vec (KDD’17)
- hetero. NE
- most cited in KDD’18

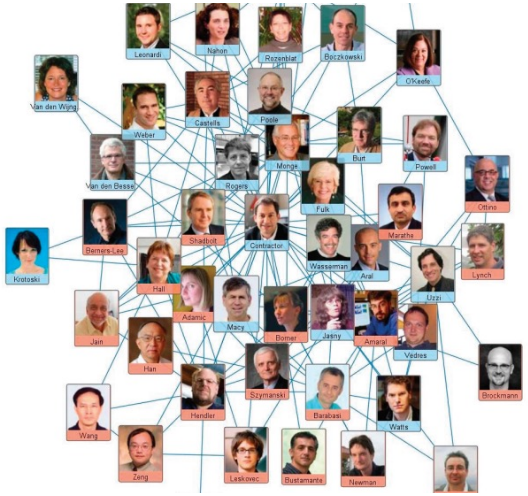
NetMF (WSDM’18)
- NE theory of DeepW, LINE, node2vec…
- top cited WSDM’18

NetSMF (WWW’19)
- 100M nodes
- best paper candidate
- used in MSFT

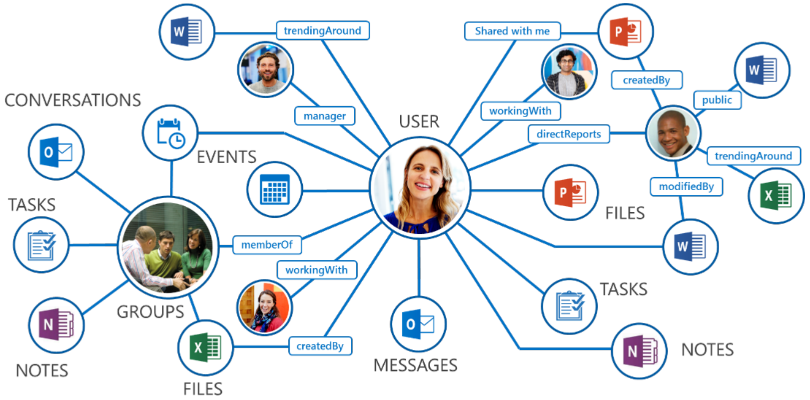
ProNE (IJCAI’19)
- 100M nodes
- 10-400X speedup

SketchNE (TKDE’23)
- **3.5B nodes in 1 hour w/ accuracy superiority**

Graphs in Society

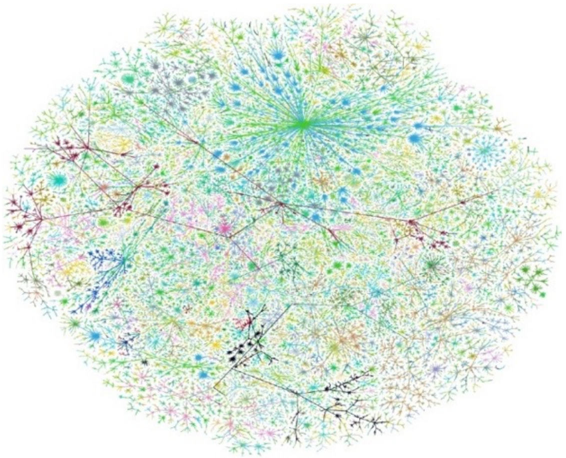


Academic Graph

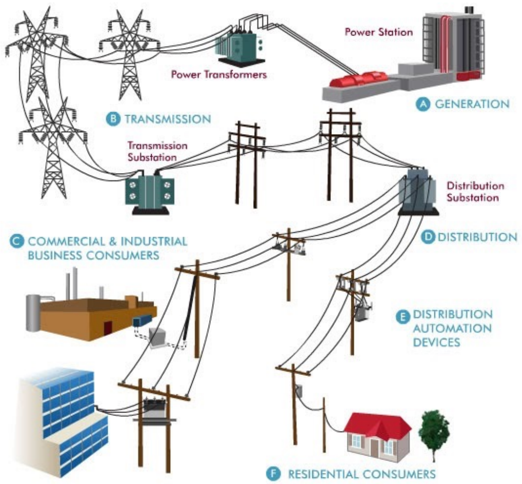


Social & Office Graph

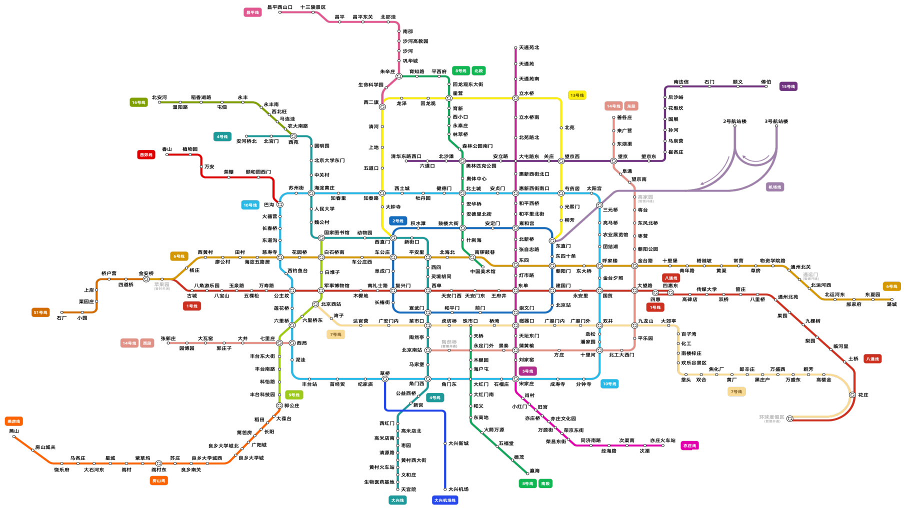
Knowledge Graph



Internet



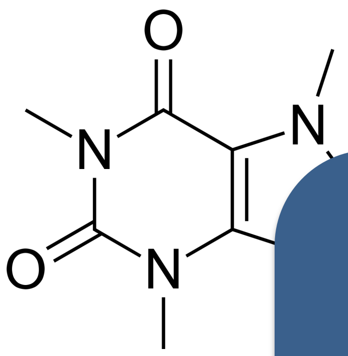
Electrical Grid Network



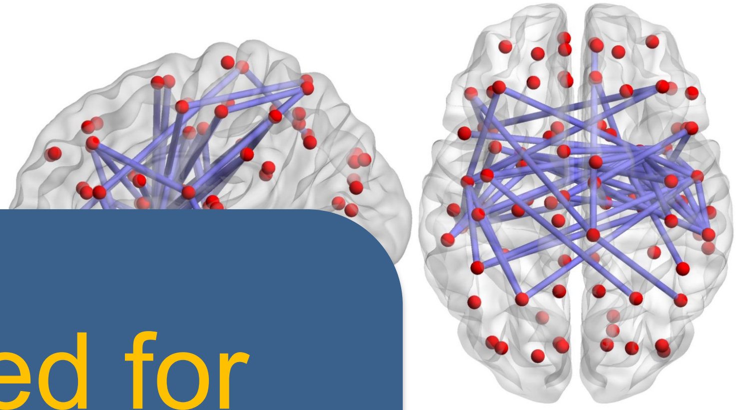
Transportation

figure credit: Web

Graphs in Nature

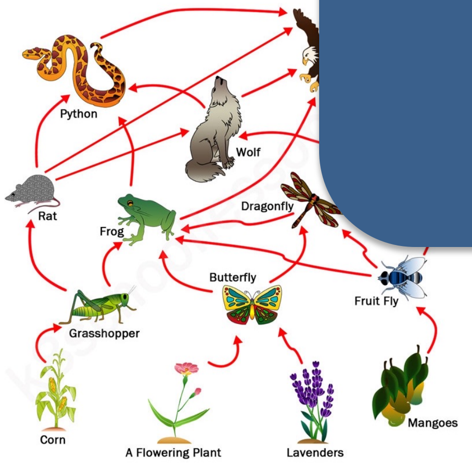


Molecules

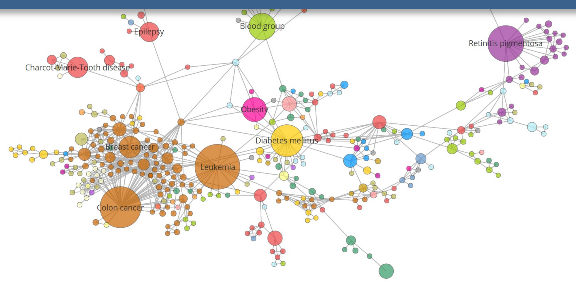


orks

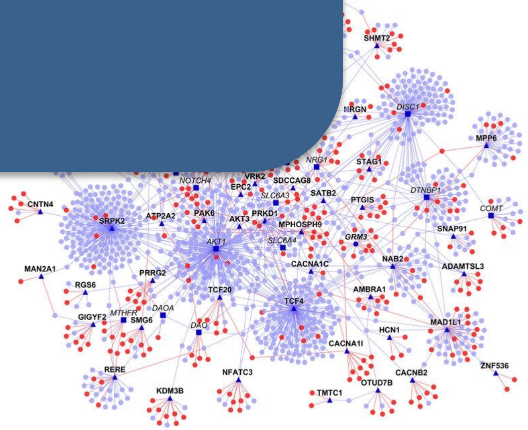
Graphs are widely used for abstracting complex systems of interacting objects!



Food Web



Human Disease Networks



Protein-Protein Interactions

figure credit: Web

Graph & Network Research

2014~2022
- Network Embedding [Perozzi et al.]
- Graph Conv. Networks [Kips & Welling]
- Graph Conv. N [Niepert, Defferrard et al.]

2010~2013
- Info. vs. Social (Twitter) [Kwak et al.]
- Signed Networks [Leskovec et al.]
- Semantic Social Networks [Tang et al.]
- 4 Deg. Of Separation [Backstrom et al.]
- Structural Diversity [Ugander et al.]
- Computational Social Science [Watts]

2005~2009
- Graph Evolution [Leskovec et al.]
- Social Influence Analysis [Tang et al.]
- Network Heterogeneity [Sun & Han]
- Network Embedding [Tang & Liu]
- Computer Social Science [Lazer et al.]

2000~2004
- Inf. Max'n [Domingos & Kempe et al.]
- Comm. Detection [Girvan & Newman]
- Network Motifs [Milo et al.]
- Link Pred. [Liben-Nowell & Kleinberg]

1998/9
- Small Worlds [Watts & Strogatz]
- Scale Free [Barabasi & Albert]
- Power Law [Faloutsos × 3]

1997
- HITS [Kleinberg]
- **PageRank [Page & Brin]**
- Hyperlink Vector Voting [Li]

1992
- Structural Hole [Burt]
- Dunbar's Number [Dunbar]

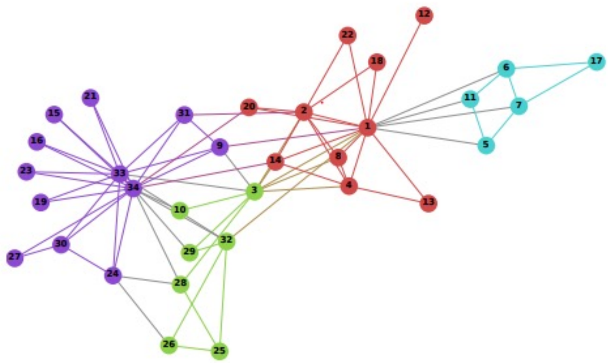
1970s
- The Strength Of Weak Tie [Granovetter]

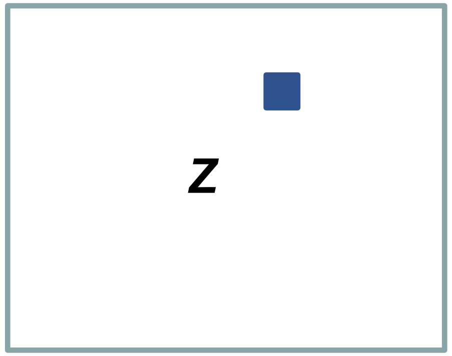
1960s
- **Small Worlds [Milgram]**

1950s
- Random Graph [Erdos, Renyi, Gilbert]
- Degree Sequence [Tuttle, Havel, Hakami]
- Homophily [Lazarsfeld & Merton]
- Balance Theory [Heider et al.]

1930s
- Sociogram [Moreno]

Graph Representation Learning





~~hand-crafted~~ **latent** feature matrix

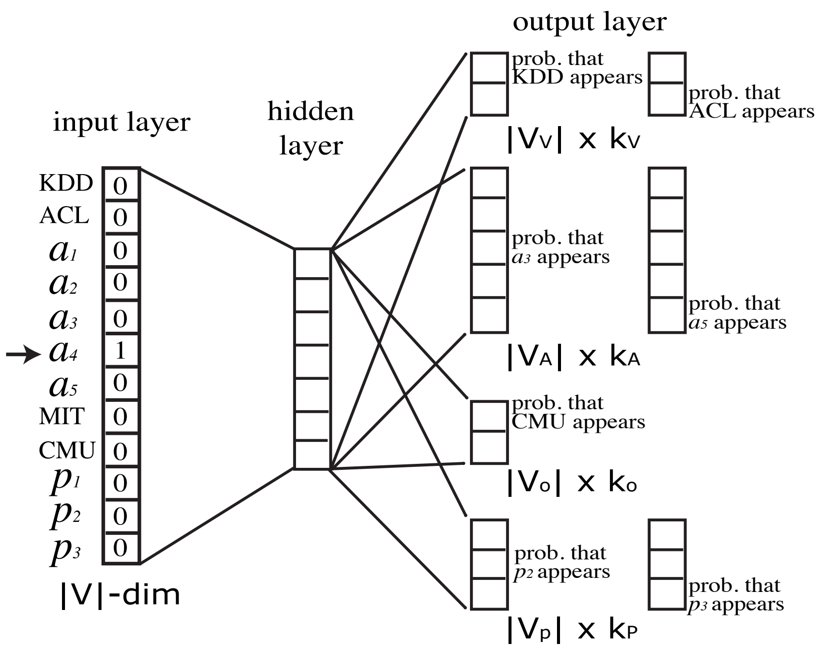
feature ~~engineering~~ **learning**

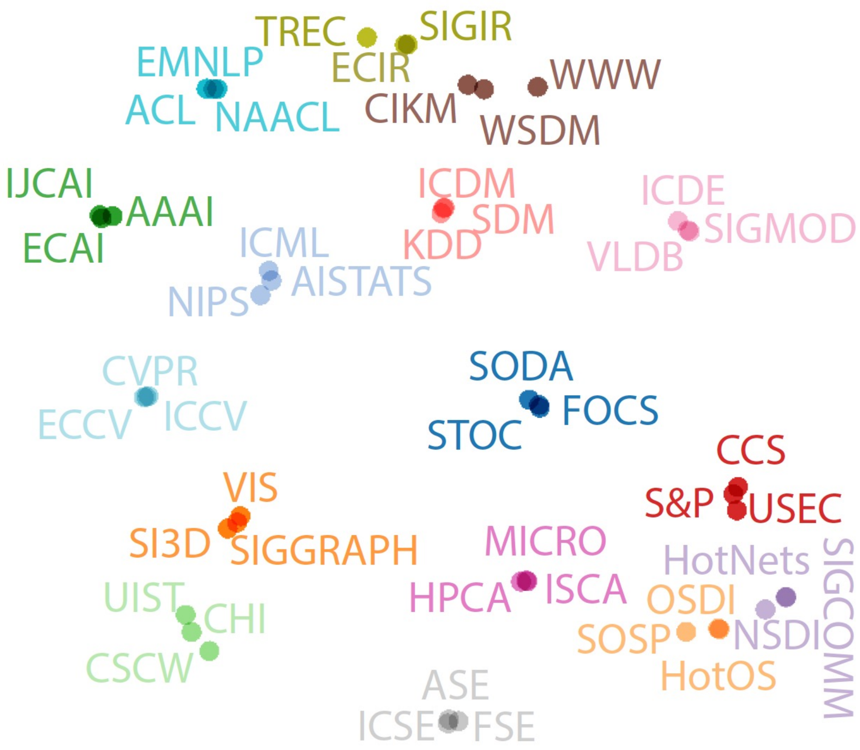
Graph & Network applications

- Node classification
- Link prediction
- Community detection
- Anomaly detection
- Social influence
- Graph evolution
- … …

- Input: a network $G = (V, E)$
- Output: $\boldsymbol{Z} \in R^{|V| \times k}, k \ll |V|$, k-dim vector $\boldsymbol{Z}_v$ for each node *v*.

Graph Representation Learning: An Example

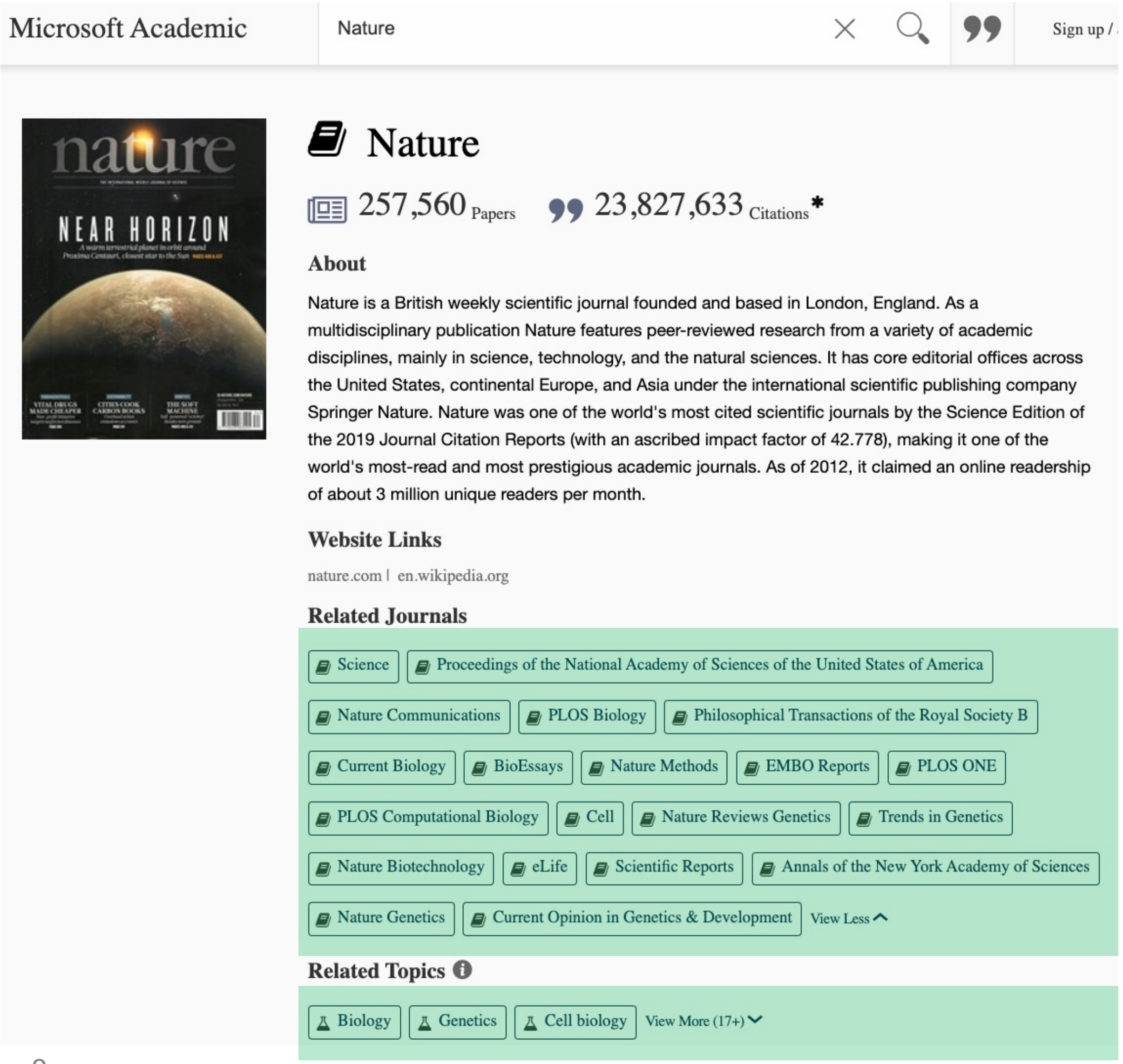




- Input: a graph $G = (V, E)$
- Output: $\boldsymbol{Z} \in R^{|V| \times k}$, $k \ll |V|$, k-dim vector $\boldsymbol{Z}_v$ for each node v.

Dong, Chawla, Swami. metapath2vec: scalable representation learning for heterogeneous networks. In *KDD* 2017.

Graph Representation Learning: An Example



Microsoft Academic | Nature | Sign up /

Nature

257,560 Papers | 23,827,633 Citations*

About

Nature is a British weekly scientific journal founded and based in London, England. As a multidisciplinary publication Nature features peer-reviewed research from a variety of academic disciplines, mainly in science, technology, and the natural sciences. It has core editorial offices across the United States, continental Europe, and Asia under the international scientific publishing company Springer Nature. Nature was one of the world's most cited scientific journals by the Science Edition of the 2019 Journal Citation Reports (with an ascribed impact factor of 42.778), making it one of the world's most-read and most prestigious academic journals. As of 2012, it claimed an online readership of about 3 million unique readers per month.

Website Links

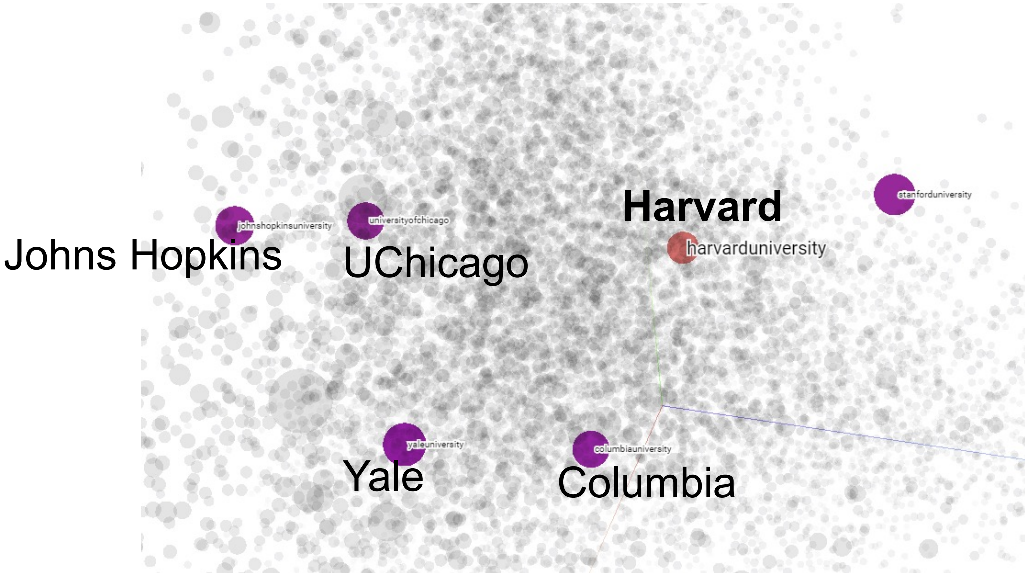
nature.com | en.wikipedia.org

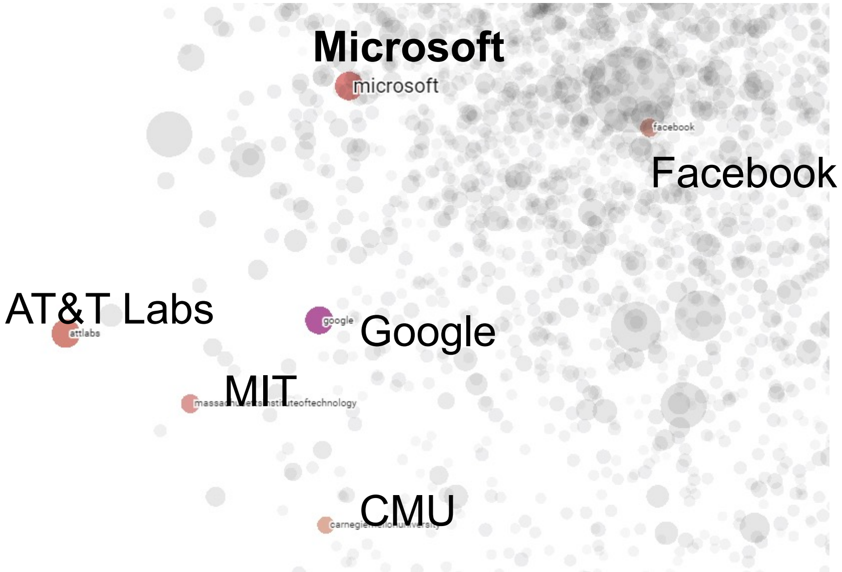
Related Journals

Science | Proceedings of the National Academy of Sciences of the United States of America | Nature Communications | PLOS Biology | Philosophical Transactions of the Royal Society B | Current Biology | BioEssays | Nature Methods | EMBO Reports | PLOS ONE | PLOS Computational Biology | Cell | Nature Reviews Genetics | Trends in Genetics | Nature Biotechnology | eLife | Scientific Reports | Annals of the New York Academy of Sciences | Nature Genetics | Current Opinion in Genetics & Development | View Less

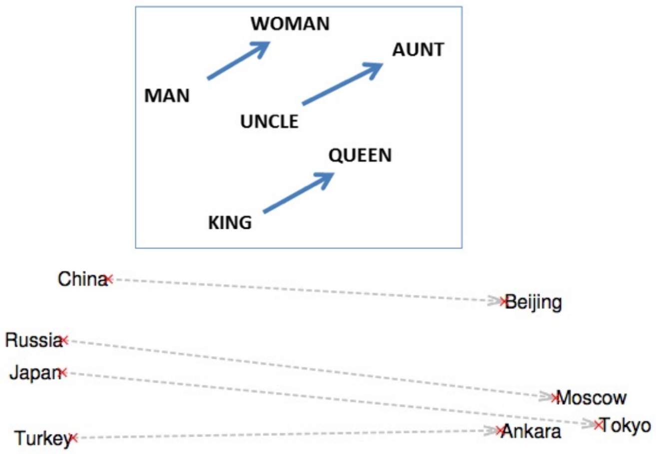
Related Topics

Biology | Genetics | Cell biology | View More (17+)

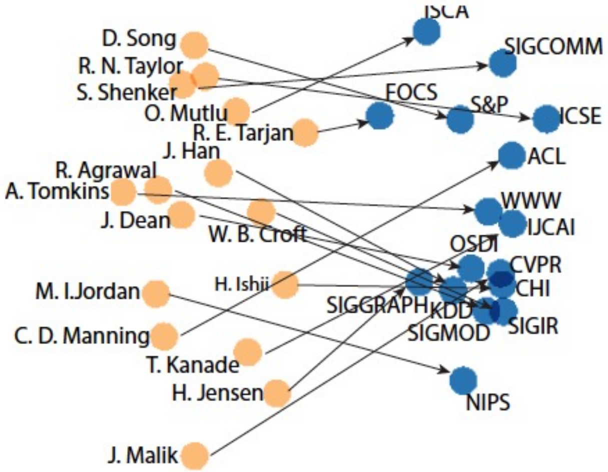




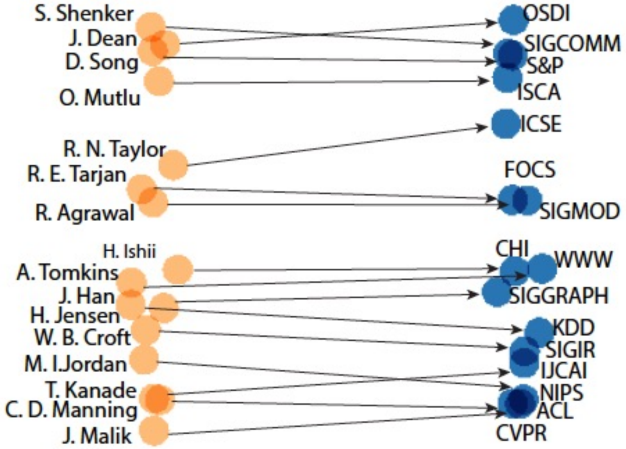
Graph Representation Learning: An Example



word2vec [Mikolov, 2013]



DeepWalk / node2vec

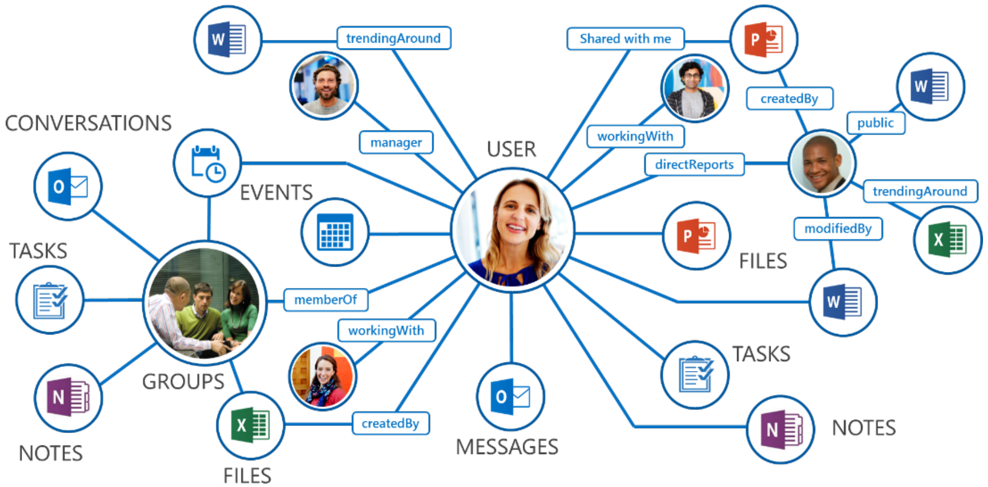


metapath2vec

Dong, Chawla, Swami. metapath2vec: scalable representation learning for heterogeneous networks. In *KDD* 2017.

(Tr)**Billion-Scale**, Heterogeneous, Dynamic, **No Labels, Many Tasks**

Microsoft/AMiner Academic Graph



Microsoft Office Graph

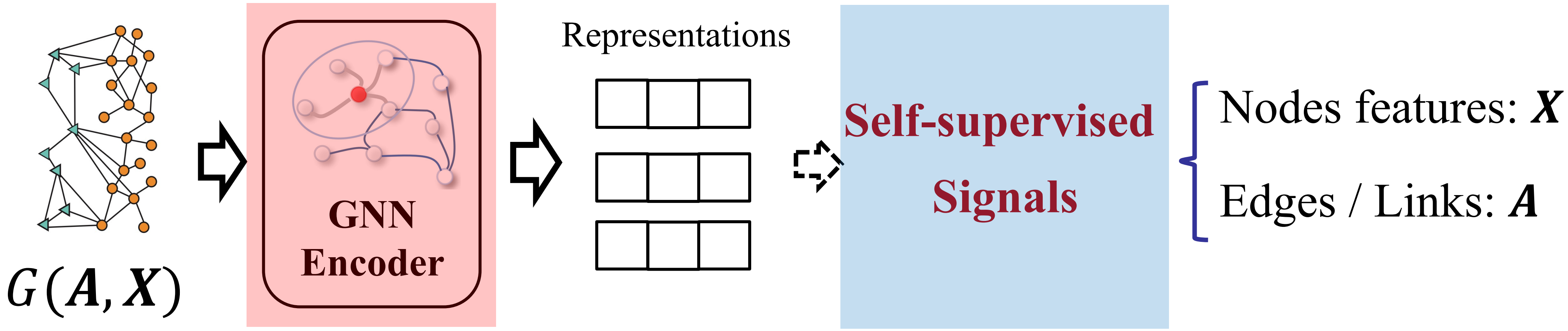
LinkedIn Economic Graph

Facebook Entity Graph

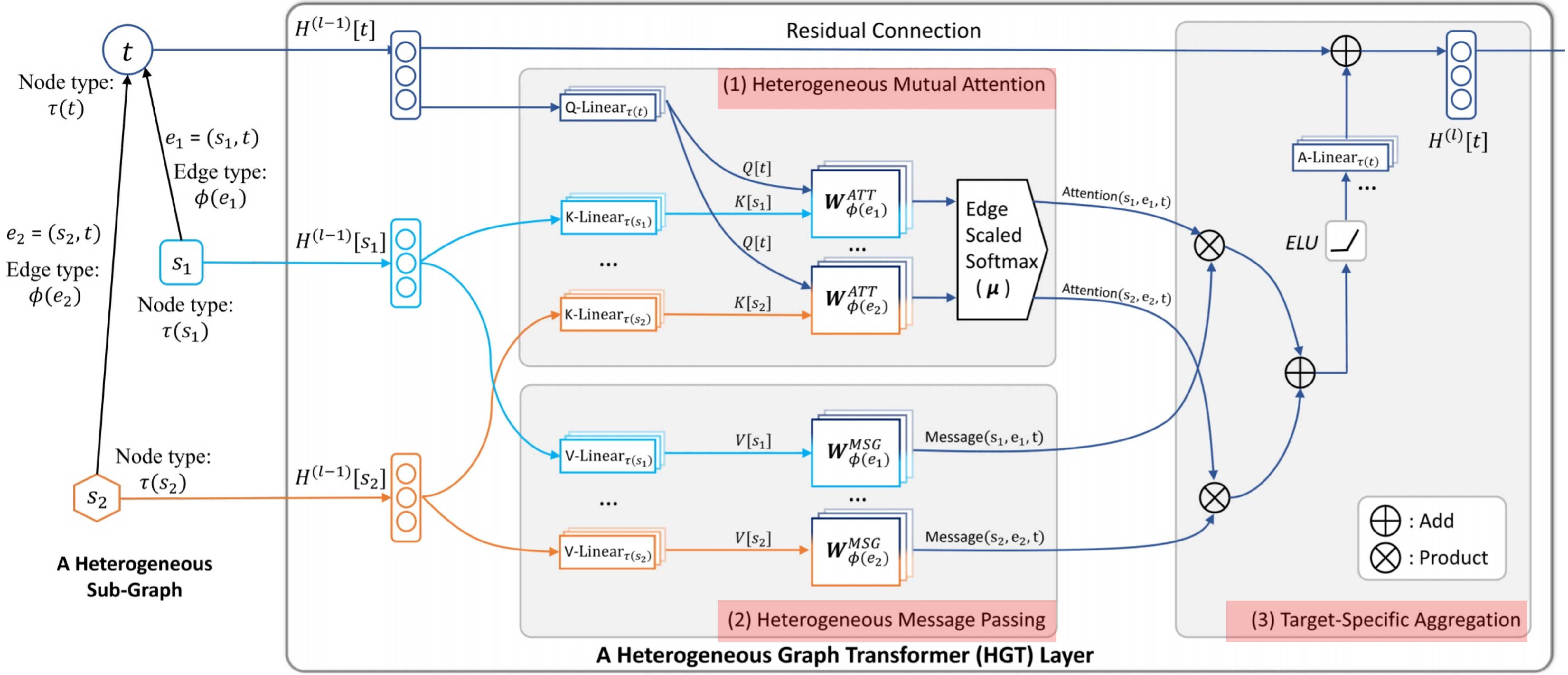
Figure Credit: Microsoft/LinkedIn/Facebook

GNN Pre-Training

- **Supervise graph models by using unlabeled data**
  - Task-specific labeled data — *Expensive to obtain*
  - Unlabeled data — *Abundant*
- **How to utilize unlabeled data effectively?**
  - Self-supervised learning



Heterogeneous Graph Transformer (HGT)



Hu, et al. Heterogeneous Graph Transformer. **WWW 2020.** Most cited in WWW’20.

Case Study

Experiments done w/o 2020 data!

Venue	Time	Top−5 Most Similar Venues
WWW	2000	SIGMOD, VLDB, NSDI, GLOBECOM, SIGIR
	2010	GLOBECOM, KDD, CIKM, SIGIR, SIGMOD
	2020	KDD, GLOBECOM, SIGIR, WSDM, SIGMOD
KDD	2000	SIGMOD, ICDE, ICDM, CIKM, VLDB
	2010	ICDE, WWW, NeurIPS, SIGMOD, ICML
	2020	NeurIPS, SIGMOD, WWW, AAAI, EMNLP
NeurIPS	2000	ICCV, ICML, ECCV, AAAI, CVPR
	2010	ICML, CVPR, ACL, KDD, AAAI
	2020	ICML, CVPR, ICLR, ICCV, ACL

DB + Networking + IR

⇩

DM + Networking + IR + DB

DB + DM

⇩

ML + DB + Web + AI + NLP!!!

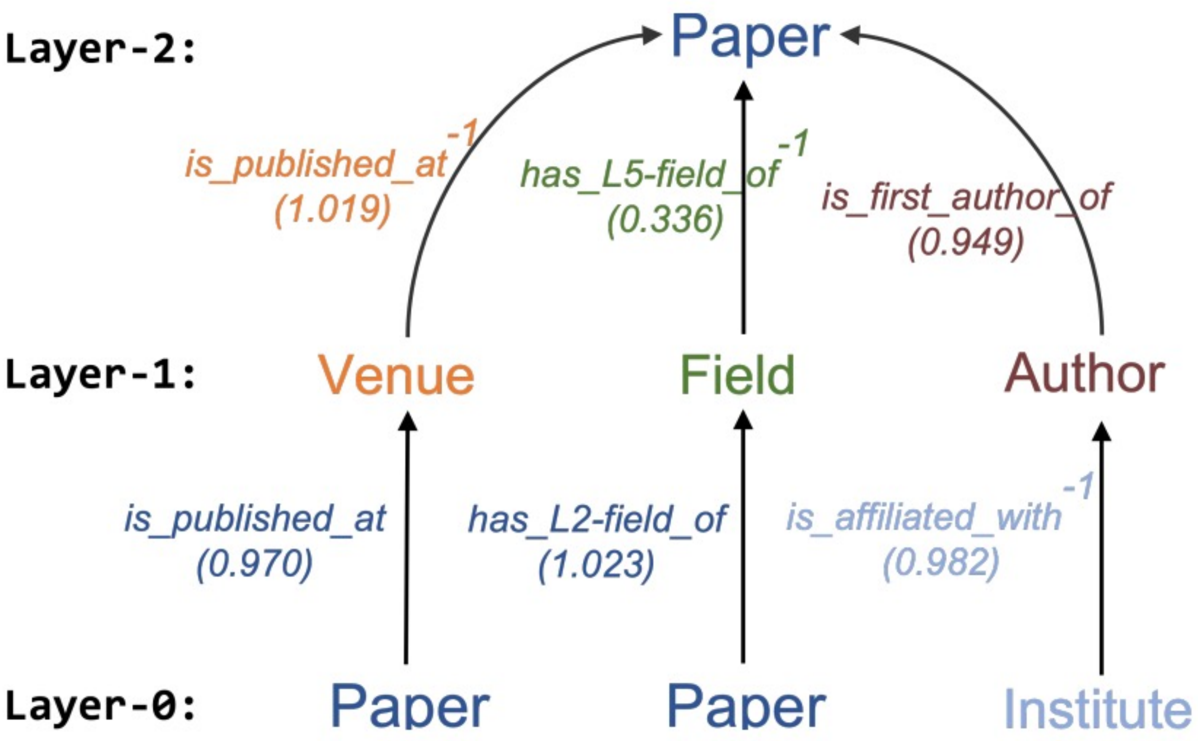
CV + ML + AI

⇩

ML + CV + DL + NLP

Hu, Dong, Wang, Sun. Heterogeneous Graph Transformer. **WWW 2020**

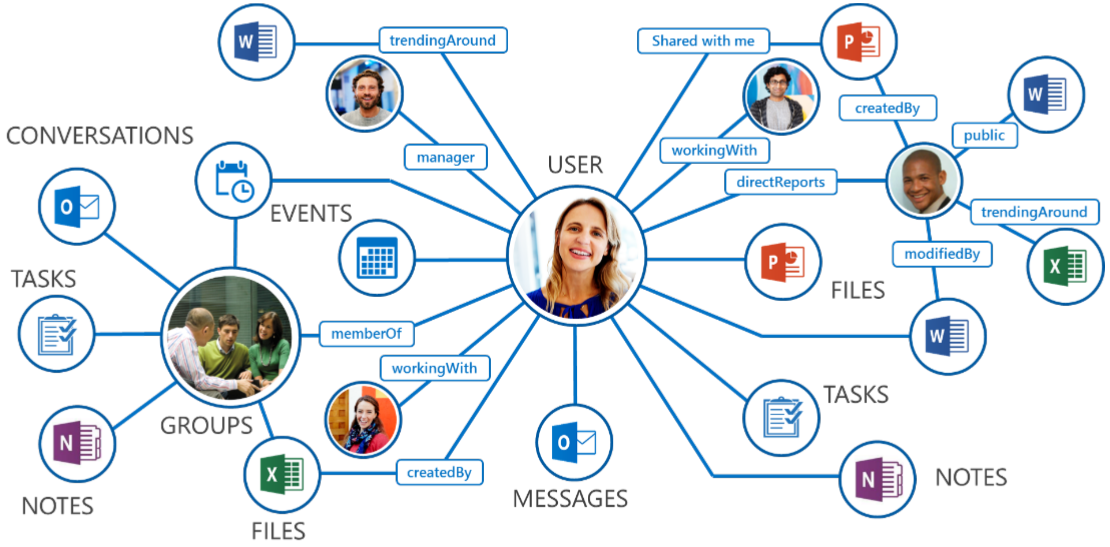
What is the Best Part of HGT?



Learn meta-paths & their weights implicitly and automatically!

Hu, Dong, Wang, Sun. Heterogeneous Graph Transformer. **WWW 2020.**

Powering the Microsoft Office Graph



One enterprise graph (monthly)

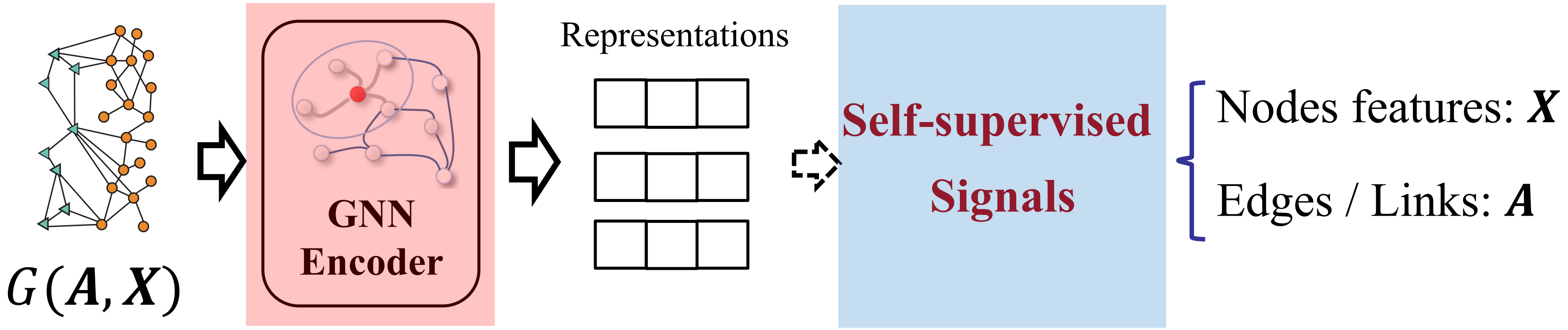
- 1.6 billion entities
  - 7 types of entities
- 7.8 trillion edges

Anomaly detection on Microsoft Office Graph

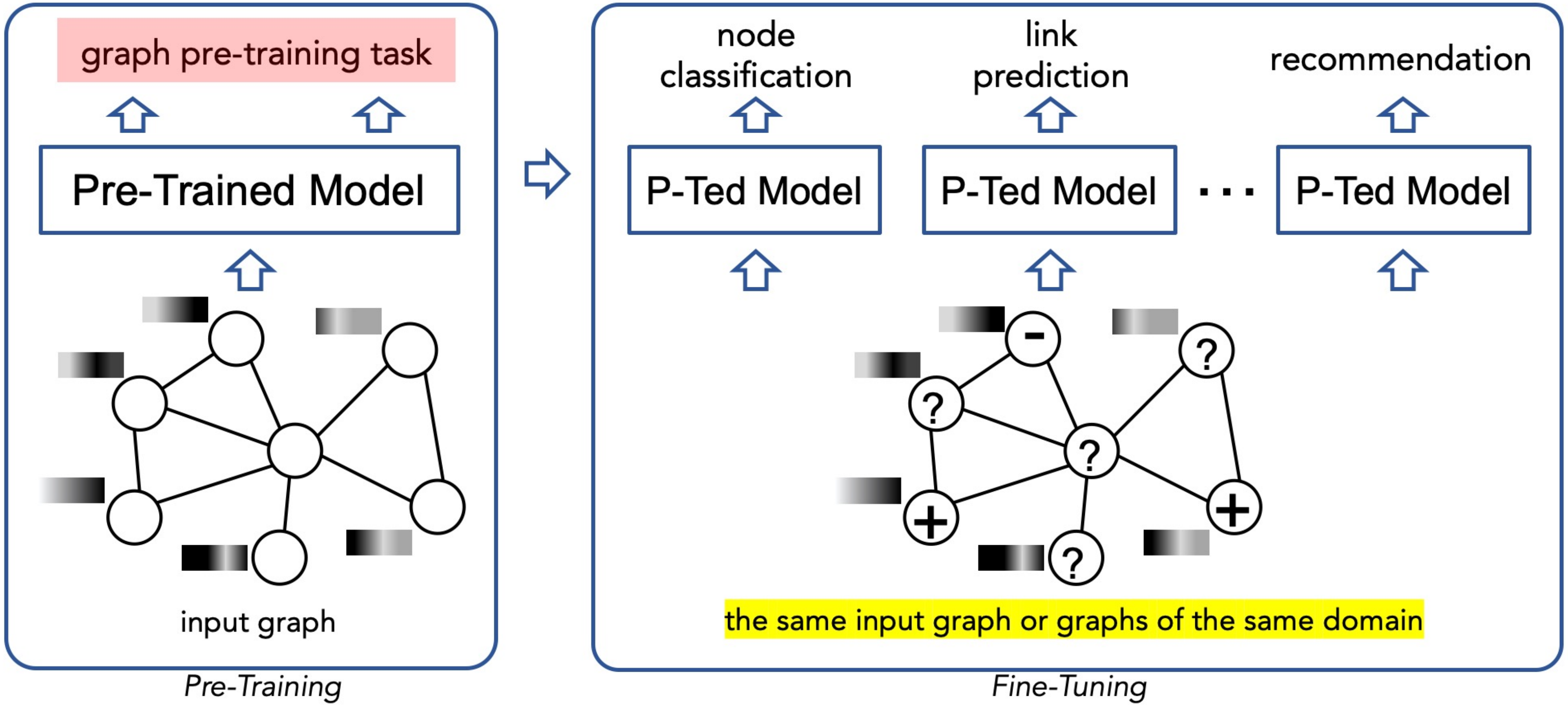
	Prec.	Recall	F1	Accu.
GraphSage	+0.00	+0.09	+0.06	+0.03
Graph Attention	+0.01	+0.11	+0.08	+0.03
HGT	+0.01	+0.30	+0.19	+0.07

GNN Pre-Training

- **Supervise graph models by using unlabeled data**
  - Task-specific labeled data — *Expensive to obtain*
  - Unlabeled data — *Abundant*
- **How to utilize unlabeled data effectively?**
  - Self-supervised learning



GNN Pre-Training

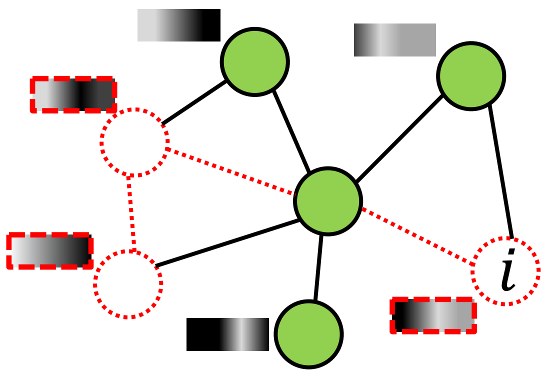


1.Hu, Dong, Wang, Chang, Sun. GPT-GNN: Generative Pre-Training of Graph Neural Networks. **KDD** 2020.

GPT-GNN: Generative Pre-Training of GNNs

- Model the graph distribution $p(G;\theta)$ by learning to reconstruct the input graph.
  - Factorize the graph likelihood into two terms:
    - Attribute Generation
    - Edge Generation

$$\log p_\theta(X, E) = \sum_{i=1}^{|\mathcal{V}|} \log p_\theta(X_i, E_i \mid X_{<i}, E_{<i}).$$



attribute and edge masked
input graph

$$p_\theta(X_i, E_i | X_{<i}, E_{<i})$$
$$= p_\theta(X_i | X_{<i}, E_{<i}) \cdot p_\theta(E_i | X_{<i}, E_{<i})$$

?

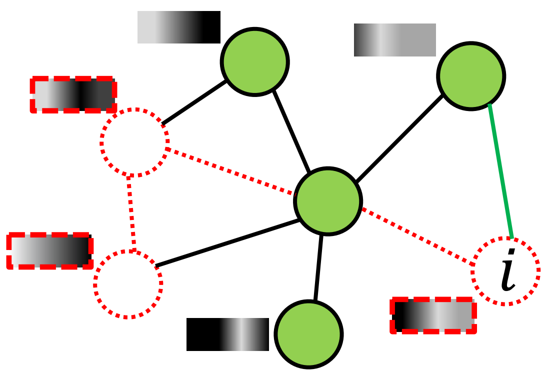
Lose the dependency between X_i and E_i

1.Hu, Dong, Wang, Chang, Sun. GPT-GNN: Generative Pre-Training of Graph Neural Networks. **KDD** 2020.

GPT-GNN: Generative Pre-Training of GNNs

- Model the graph distribution $p(G;\theta)$ by learning to reconstruct the input graph.
  - Factorize the graph likelihood into two terms:
    - Attribute Generation: given observed edges, generate node attributes
    - Edge Generation: given observed edges and generated attributes, generate masked edges

$$\log p_\theta(X, E) = \sum_{i=1}^{|\mathcal{V}|} \log p_\theta(X_i, E_i \mid X_{<i}, E_{<i}).$$

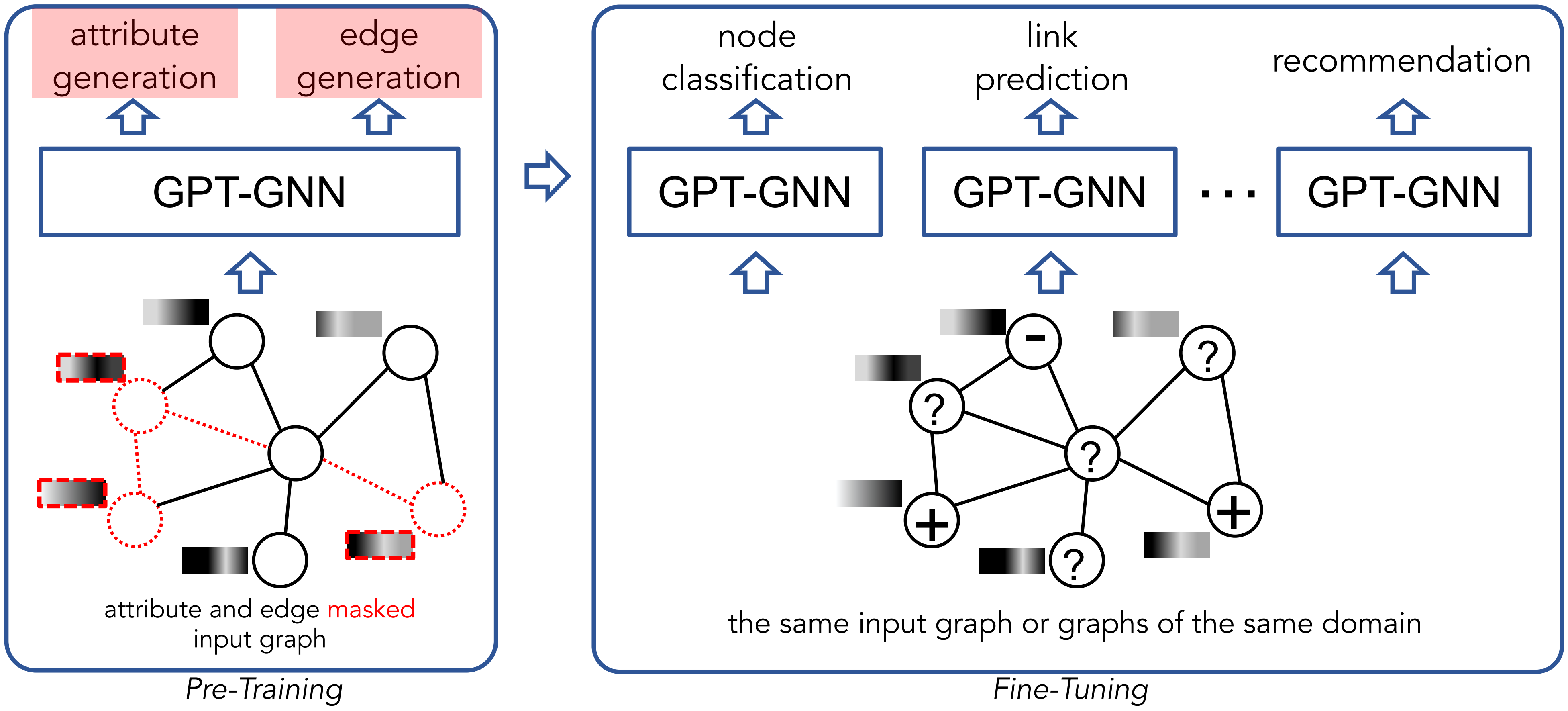


attribute and edge masked
input graph

$$\begin{aligned}
&p_\theta(X_i, E_i \mid X_{<i}, E_{<i}) \\
&= \sum_o p_\theta(X_i, E_{i,\neg o} \mid E_{i,o}, X_{<i}, E_{<i}) \cdot p_\theta(E_{i,o} \mid X_{<i}, E_{<i}) \\
&= \mathbb{E}_o \left[p_\theta(X_i, E_{i,\neg o} \mid E_{i,o}, X_{<i}, E_{<i}) \right] \\
&= \mathbb{E}_o \Big[\underbrace{p_\theta(X_i \mid E_{i,o}, X_{<i}, E_{<i})}_{\text{1) generate attributes}} \cdot \underbrace{p_\theta(E_{i,\neg o} \mid E_{i,o}, X_{\le i}, E_{<i})}_{\text{2) generate edges}} \Big].
\end{aligned}$$

1.Hu, Dong, Wang, Chang, Sun. GPT-GNN: Generative Pre-Training of Graph Neural Networks. **KDD** 2020.

GPT-GNN: Generative Pre-Training of GNNs



GPT-GNN: Generative Pre-Training of GNNs

- Data: Microsoft Academic Graph

	Pre-Train	Fine-Tune
Tasks:	• Attribute Generation • Edge Generation	• Inferring the topic of each paper • Inferring the venue of each paper • Author name disambiguation

Base GNN model: Heterogeneous Graph Transformer (HGT)

GPT-GNN: Generative Pre-Training of GNNs

- Data: Microsoft Academic Graph

	Pre-Train	Fine-Tune
No Transfer:	CS Academic Graph	CS Academic Graph
Field Transfer:	Med, Bio, Physics…	CS Academic Graph
Time Transfer:	CS before 2014	CS after 2014
Time + Field Transfer:	Med, Bio, Physics… before 2014	CS after 2014

1.Hu, Dong, Wang, Chang, Sun. GPT-GNN: Generative Pre-Training of Graph Neural Networks. **KDD** 2020.

GPT-GNN: Generative Pre-Training of GNNs

	Downstream Dataset	OAG		
	Evaluation Task	Paper–Field	Paper–Venue	Author ND
	No Pre-train	.346±.149	.598±.122	.813±.105
Field Transfer	GAE	.403±.114	.626±.093	.836±.084
	GraphSAGE (unsp.)	.368±.125	.609±.096	.818±.092
	Graph Infomax	.387±.112	.612±.097	.827±.084
	GPT-GNN (Attr)	.396±.118	.623±.105	.834±.086
	GPT-GNN (Edge)	.413±.109	.635±.096	.842±.093
	GPT-GNN	**.420±.107**	**.641±.098**	**.848±.102**
Time Transfer	GAE	.384±.117	.619±.101	.828±.095
	GraphSAGE (unsp.)	.352±.121	.601±.105	.815±.093
	Graph Infomax	.369±.116	.606±.102	.821±.089
	GPT-GNN (Attr)	.374±.114	.614±.098	.826±.089
	GPT-GNN (Edge)	.397±.105	.629±.102	.836±.088
	GPT-GNN	**.405±.108**	**.635±.101**	**.840±.093**
Time + Field Transfer	GAE	.371±.124	.611±.108	.821±.102
	GraphSAGE (unsp.)	.349±.130	.602±.118	.812±.097
	Graph Infomax	.360±.121	.600±.102	.815±.093
	GPT-GNN (Attr)	.364±.115	.609±.103	.824±.094
	— (w/o node separation)	.347±.128	.601±.102	.813±.108
	GPT-GNN (Edge)	.390±.116	.622±.104	.830±.105
	— (w/o adaptive queue)	.376±.121	.617±.115	.828±.104
	GPT-GNN	**.397±.112**	**.628±.108**	**.833±.102**

- **All pre-training frameworks** help the performance of GNNs
  - GAE, GraphSage, Graph Infomax
  - GPT-GNN

- **GPT-GNN helps the most** by achieving a relative performance gain of 9.1% over the base model without pre-training

- **Both self-supervised tasks in GPT-GNN** help the pre-training framework
  - Attribute generation
  - Edge generation

1.Hu, Dong, Wang, Chang, Sun. GPT-GNN: Generative Pre-Training of Graph Neural Networks. **KDD** 2020.

GPT-GNN: Generative Pre-Training of GNNs

- Data: Microsoft Academic Graph

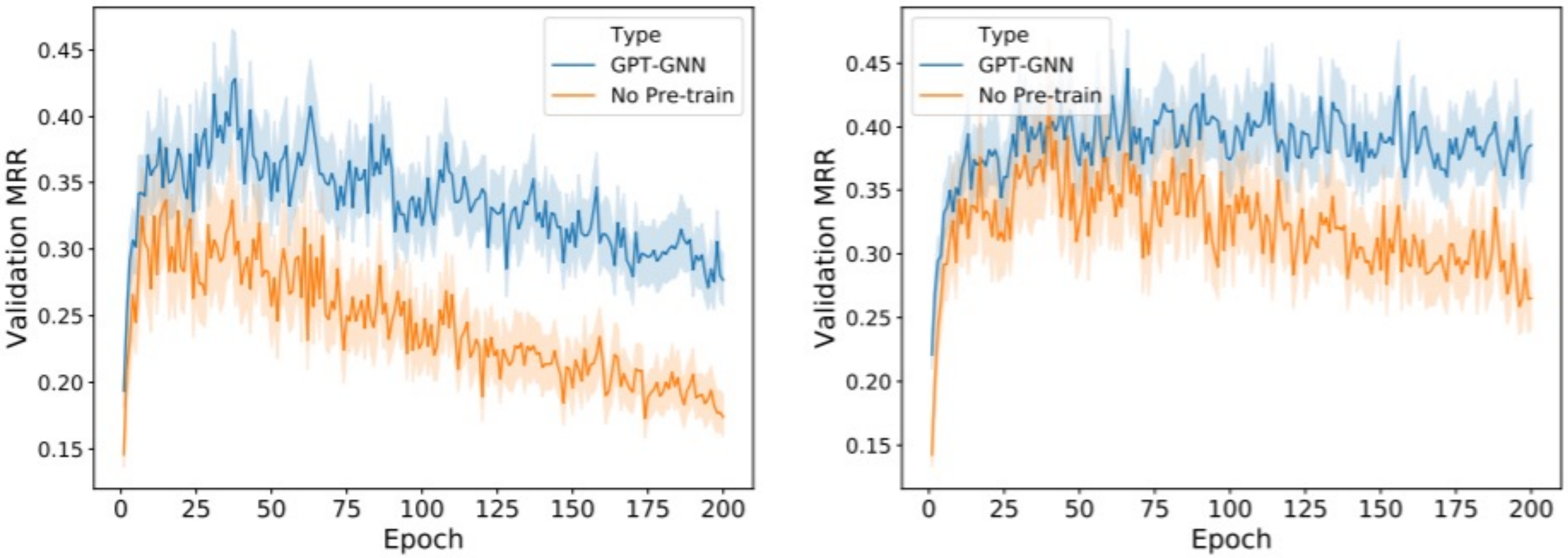
	Pre-Train	Fine-Tune
Tasks:	• Attribute Generation • Edge Generation	• Inferring the topic of each paper • Inferring the venue of each paper • Author name disambiguation

Base GNN model: Heterogeneous Graph Transformer (HGT)

Model	HGT	GCN	GAT	RGCN	HAN
No Pre-train	**.346**	.327	.318	.296	.332
GPT-GNN	**.420**	.359	.382	.351	.406
Relative Gain	21.4%	9.8%	20.1%	18.9%	22.3%

1.Hu, Dong, Wang, Chang, Sun. GPT-GNN: Generative Pre-Training of Graph Neural Networks. **KDD** 2020.

The Promise of Graph Pre-Training!



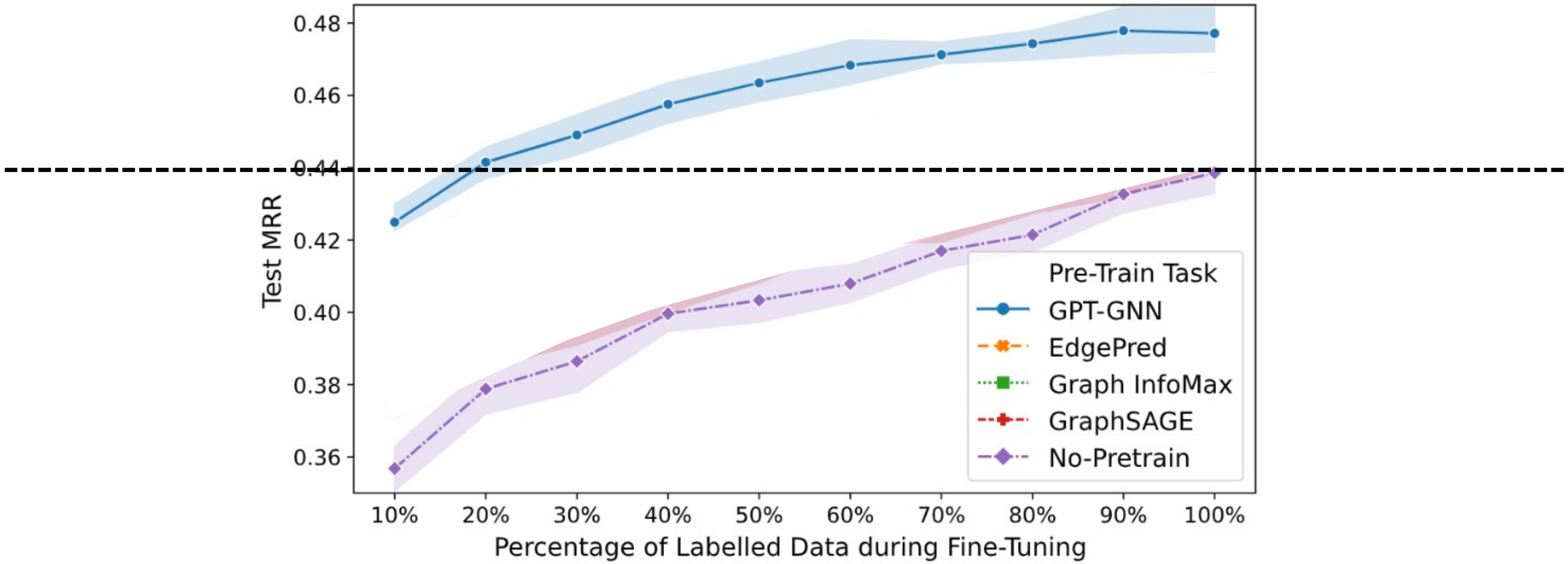
(a) Data Percentage: 10%

(b) Data Percentage: 20%

Predict Paper Title	GroundTruth Paper Title
person recognition system using automatic probabilistic classification	person re-identification by probabilistic relative distance comparison
a novel framework using spectrum sensing in wireless systems	a secure collaborative spectrum sensing strategy in cyber physical systems
a efficient evaluation of a distributed data storage service storage	an empirical analysis of a large scale mobile cloud storage service
parameter control in wireless sensor networks networks networks	optimal parameter estimation under controlled communication over sensor networks
a experimental system for for to the analysis of graphics	an interactive computer graphics approach to surface representation

The Promise of Graph Pre-Training!

During fine-turning



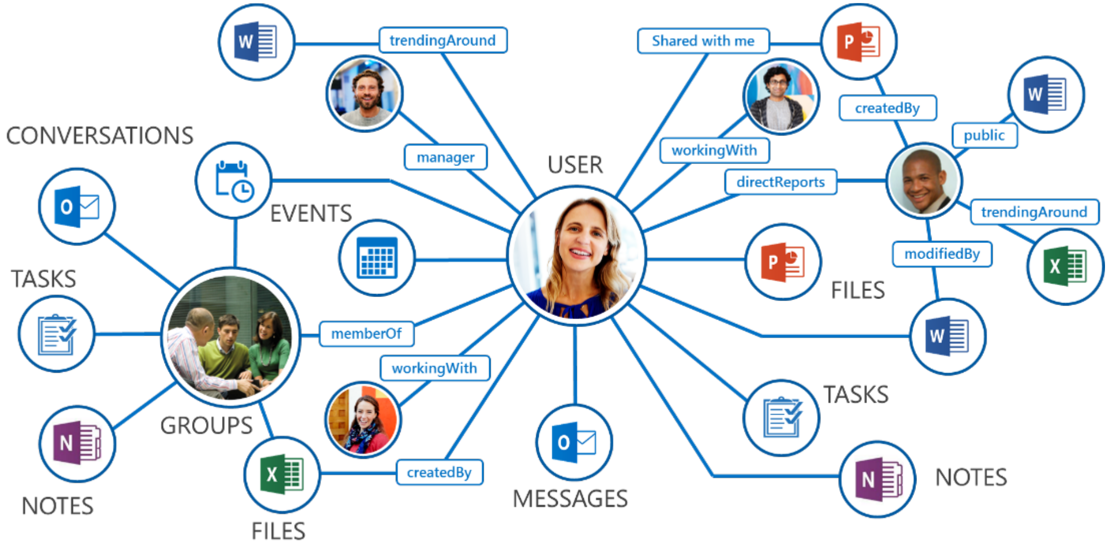
The GNN model **w/o** pre-training with **100%** training data

VS

The pre-trained GNN model with **10-20%** training data

1.Hu, Dong, Wang, Chang, Sun. GPT-GNN: Generative Pre-Training of Graph Neural Networks. **KDD** 2020.

Powering the Microsoft Office Graph



One enterprise graph (monthly)

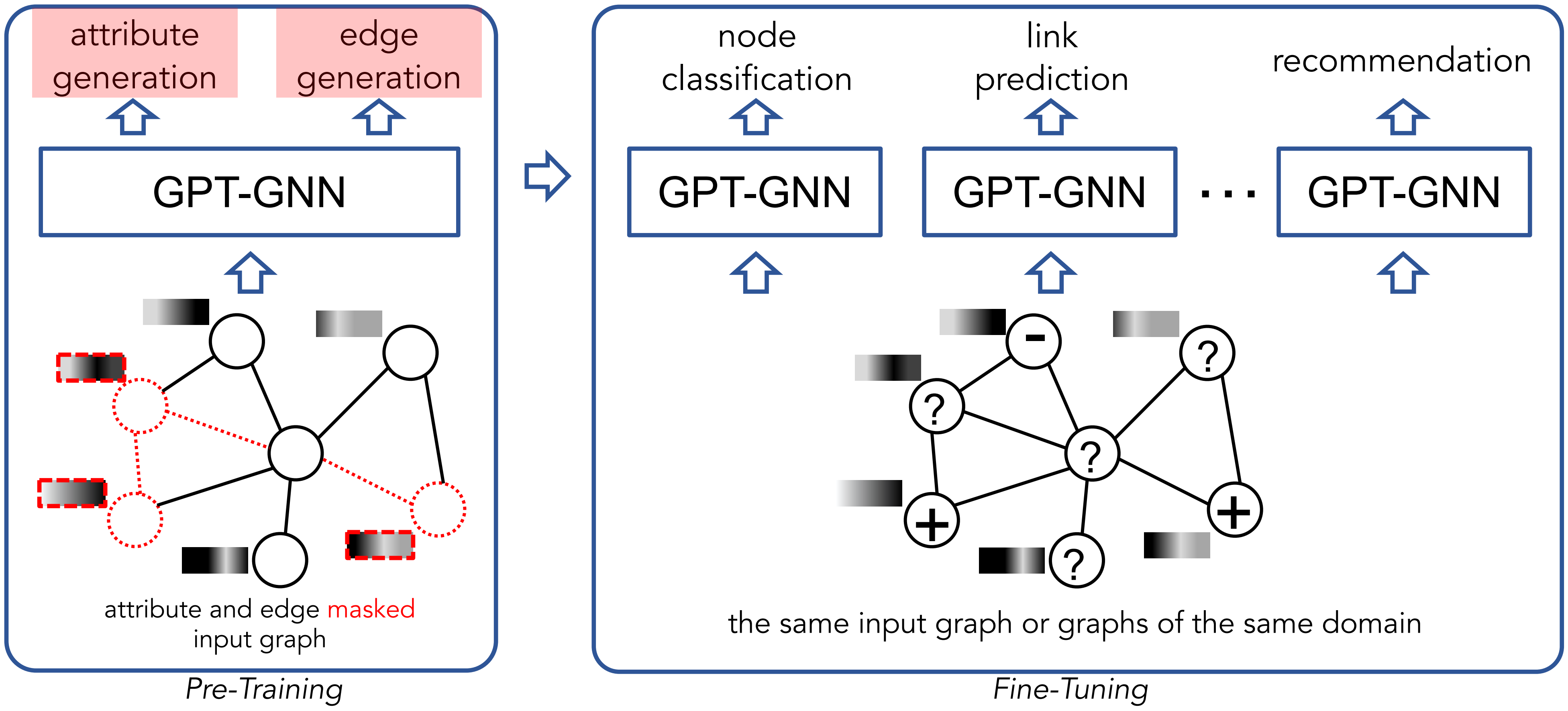
- 1.6 billion entities
  - 7 types of entities
- 7.8 trillion edges

Anomaly detection on Microsoft Office Graph

	Prec.	Recall	F1	Accu.
GraphSage	+0.00	+0.09	+0.06	+0.03
Graph Attention	+0.01	+0.11	+0.08	+0.03
HGT	+0.01	+0.30	+0.19	+0.07

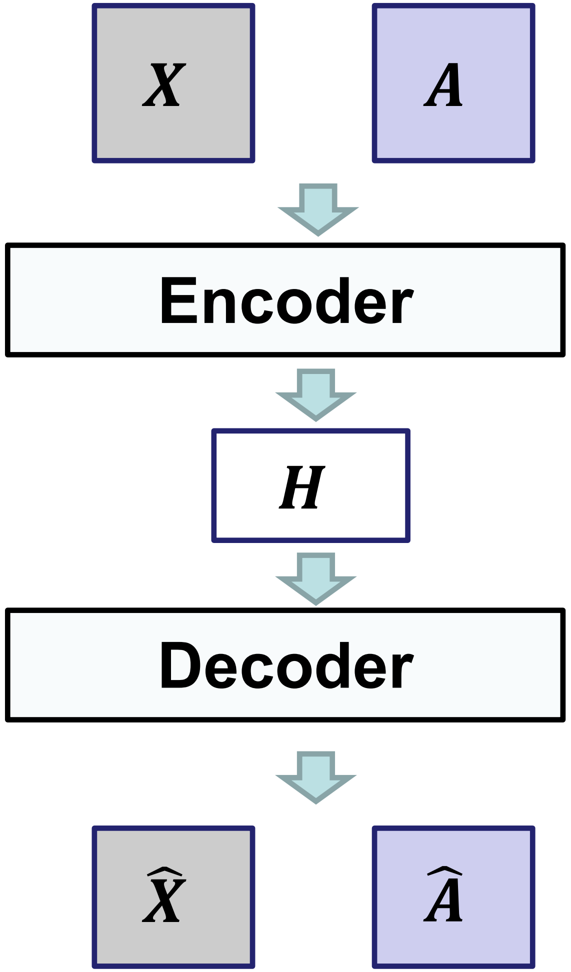
Pre-trained HGT on one enterprise ⇨ Other enterprise customers w/o data access

GPT-GNN: Generative Pre-Training of GNNs



Graph AutoEncoder

- $G = (V, A, X)$
  - $A \in \{0,1\}^{N\times N}$: adjacency matrix
  - $X \in \mathbb{R}^{N\times d}$: node features
- Encoding
  - $H = f_E(A, X)$
- Decoding
  - $G' = f_D(A, H)$
- Reconstruction objectives
  - graph structure (link)
  - node features

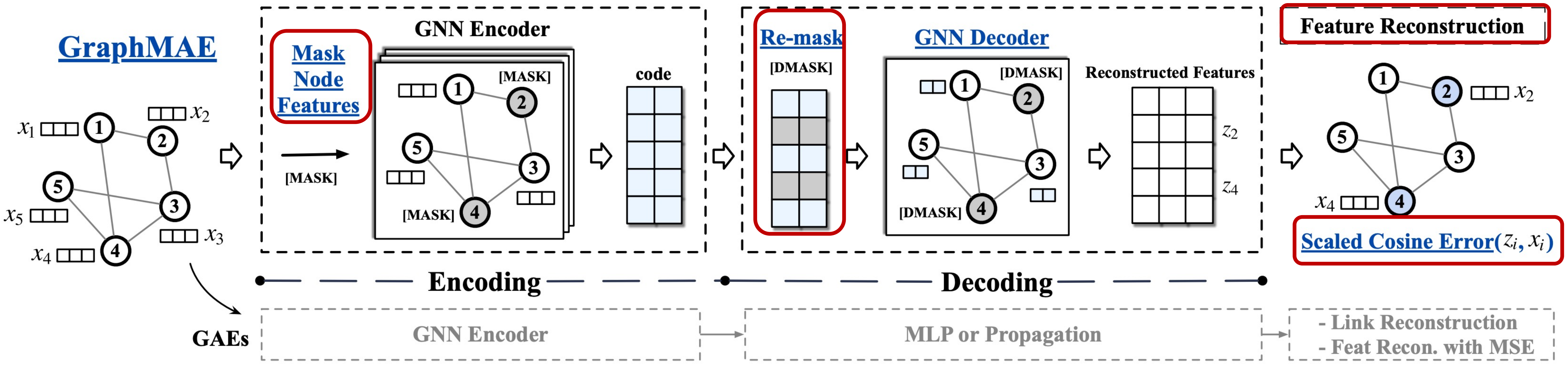


Reconstruction Target **Decoding Strategy**

Methods	Feat. Loss	AE	No Struc.	Mask Feat.	GNN Decoder	Re-mask Dec.	Space
VGAE [20]	n/a	✓	-	-	-	-	$O(N^2)$
ARVGA [26]	n/a	✓	-	-	-	-	$O(N^2)$
MGAE [42]	MSE	✓	-	✓	-	-	$O(N)$
GALA [27]	MSE	✓	✓	-	✓	-	$O(N)$
GATE [31]	MSE	✓	-	-	✓	-	$O(N)$
AttrMask [16]	CE	✓	✓	✓	-	-	$O(N)$
GPT-GNN [17]	MSE	-	-	✓	-	-	$O(N)$
AGE [3]	n/a	✓	-	-	-	-	$O(N^2)$
NodeProp [18]	MSE	✓	✓	✓	-	-	$O(N)$

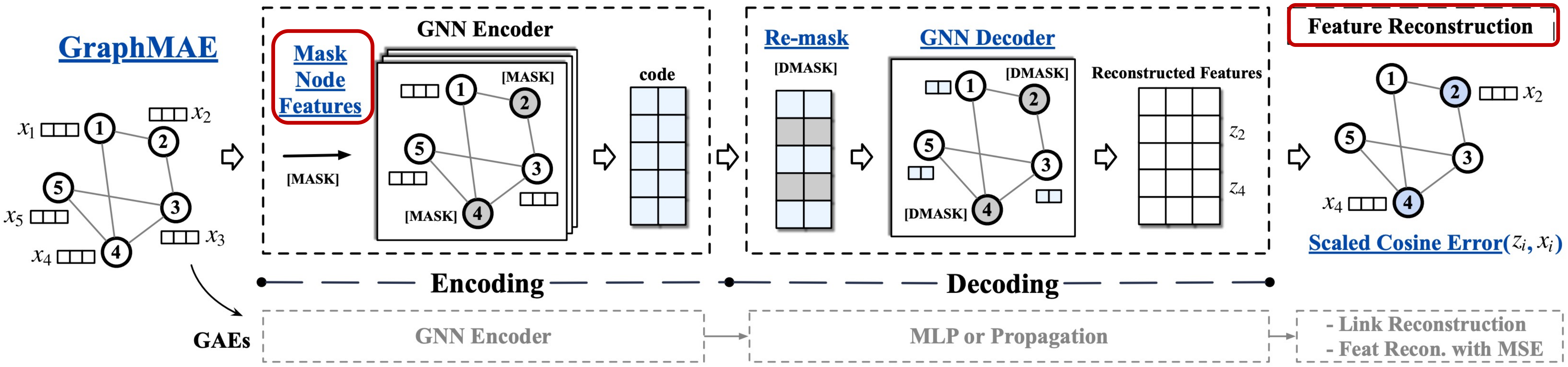
Error Function **Reconstruction Method**

GraphMAE



Hou et al. "GraphMAE: Self-supervised masked graph autoencoders." KDD’22

Masked Feature Reconstruction



– Feature construction as the learning objective

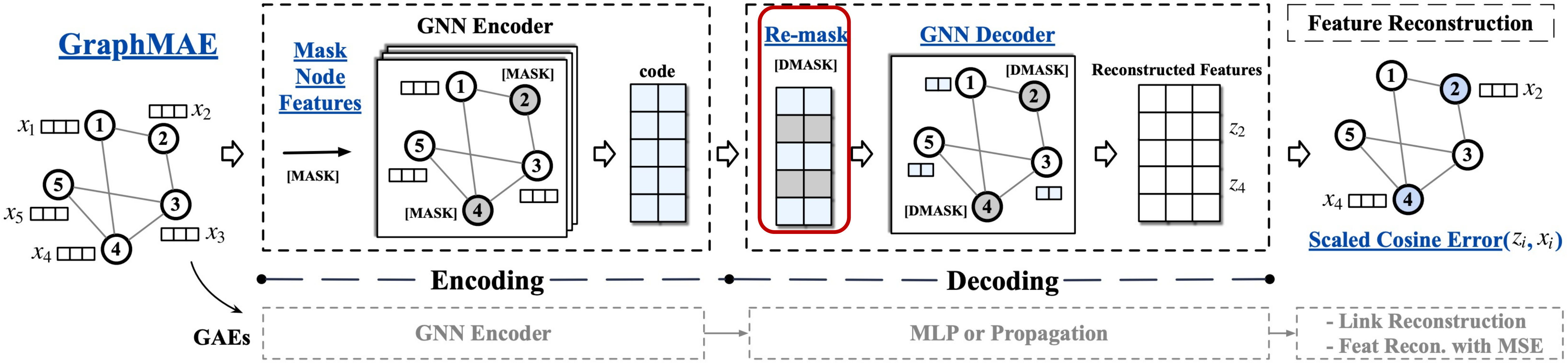
– Masked feature reconstruction

1. Sample a subset of nodes $\widetilde{V} \subset V$
2. Replace node feature with [MASK]

$$\widetilde{\boldsymbol{x}}_i = \begin{cases} \boldsymbol{x}_{[M]} & v_i \in \widetilde{\mathcal{V}} \\ \boldsymbol{x}_i & v_i \notin \widetilde{\mathcal{V}} \end{cases}$$

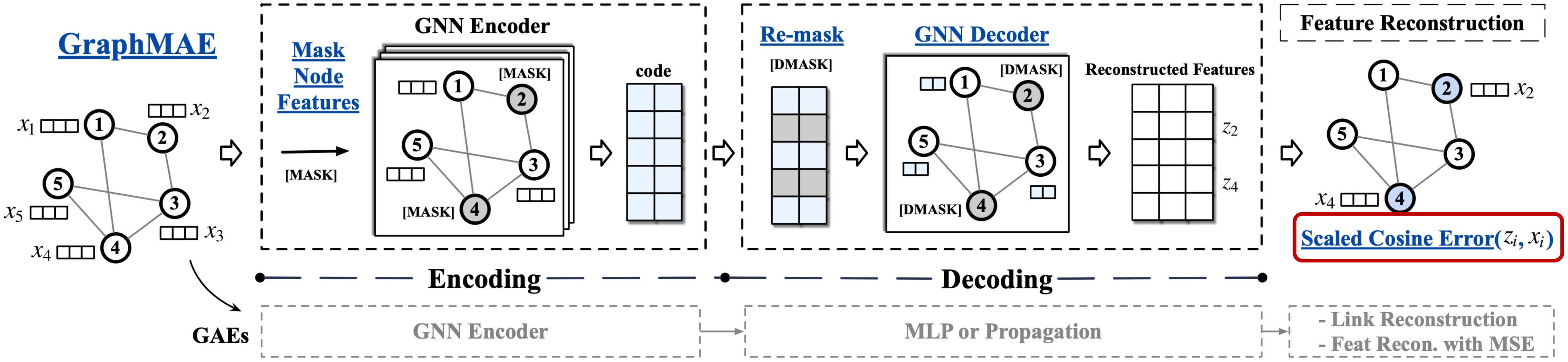
- $H = f_E(A, \tilde{X})$

GNNs as Decoder with Re-Mask Decoding



- Use a GNN as the decoder
  - A more expressive decoder helps reconstruct low informative features
- Re-mask node features before decoder
  - Re-mask the “masked” nodes
- $\widetilde{H} = \mathrm{Remask}(H),\ Z = f_D(A, \widetilde{H})$ $\quad \widetilde{\boldsymbol{h}}_i = \begin{cases} \boldsymbol{h}_{[M]} & v_i \in \widetilde{\mathcal{V}} \\ \boldsymbol{h}_i & v_i \notin \widetilde{\mathcal{V}} \end{cases}$

Scaled Cosine Error as the Criterion



- MSE fails, especially for continuous features
  - *Sensitivity & low selectivity*

$$L_{MSE} = \frac{1}{|\tilde{V}|}\sum_{v_i \in \tilde{V}} (x_i - z_i)^2$$

- Scaled cosine error as the criterion
  - *Cosine error & scaled coefficient*

$$\mathcal{L}_{\mathrm{SCE}} = \frac{1}{|\widetilde{\mathcal{V}}|}\sum_{v_i \in \widetilde{\mathcal{V}}} \left(1 - \frac{\boldsymbol{x}_i^T \boldsymbol{z}_i}{\|\boldsymbol{x}_i\| \cdot \|\boldsymbol{z}_i\|}\right)^{\gamma}, \ \gamma \geq 1,$$

Reconstruction Target

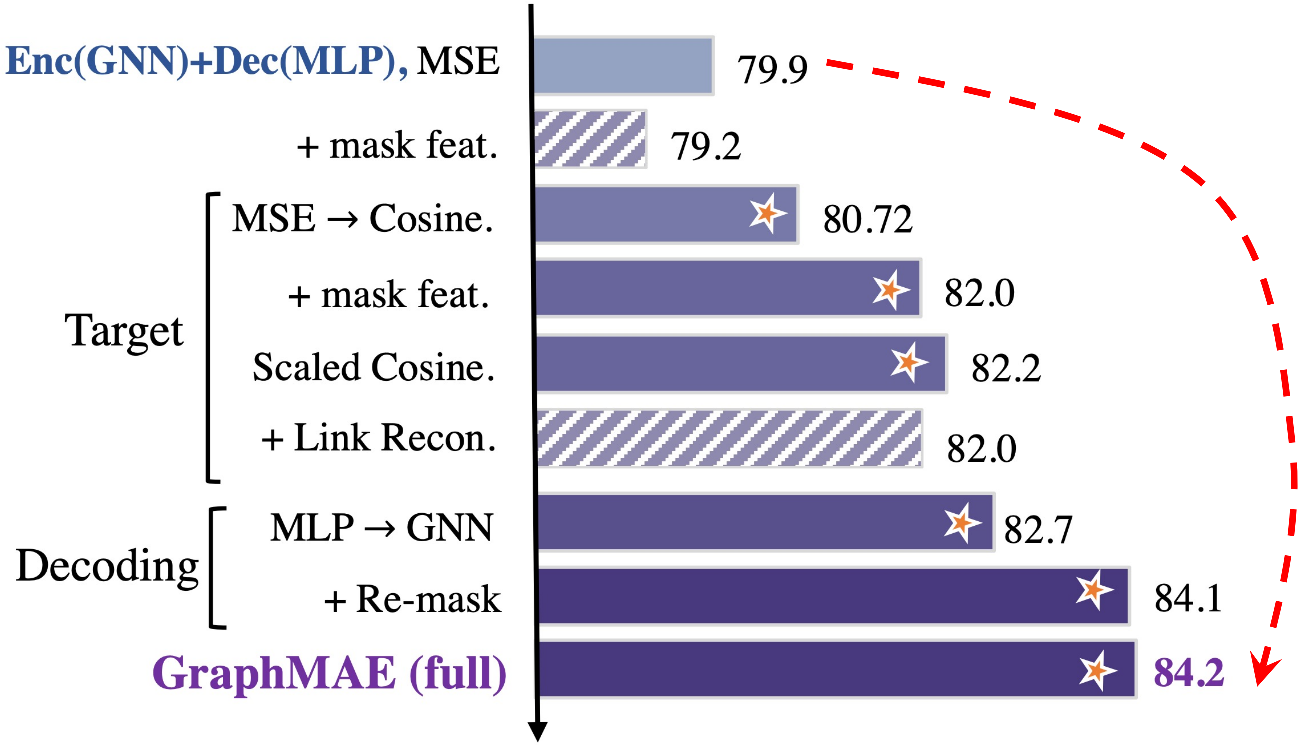
Decoding Strategy

Methods	Feat. Loss	AE	No Struc.	Mask Feat.	GNN Decoder	Re-mask Dec.	Space
VGAE [20]	n/a	✓	-	-	-	-	$O(N^2)$
ARVGA [26]	n/a	✓	-	-	-	-	$O(N^2)$
MGAE [42]	MSE	✓	-	✓	-	-	$O(N)$
GALA [27]	MSE	✓	✓	-	✓	-	$O(N)$
GATE [31]	MSE	✓	-	-	✓	-	$O(N)$
AttrMask [16]	CE	✓	✓	✓	-	-	$O(N)$
GPT-GNN [17]	MSE	-	-	✓	-	-	$O(N)$
AGE [3]	n/a	✓	-	-	-	-	$O(N^2)$
NodeProp [18]	MSE	✓	✓	✓	-	-	$O(N)$
GraphMAE	SCE	✓	✓	✓	✓	✓	$O(N)$

Error Function

Reconstruction Method

GraphMAE



(b) The effect of GraphMAE designs on the performance on Cora dataset.

Table 4: Ablation studies of decoder type, re-mask and reconstruction criterion on node- and graph-level benchmarks.

	Dataset	Node-Level			Graph-Level	
		Cora	PubMed	Arxiv	MUTAG	IMDB-B
COMP.	GraphMAE	84.2	81.1	71.75	88.19	75.52
	w/o mask	79.7	77.9	70.97	82.58	74.42
	w/o re-mask	82.7	80.0	71.61	86.29	74.42
	w/ MSE	79.1	73.1	67.44	86.30	74.04
Decoder	MLP	82.2	80.4	71.54	87.16	73.94
	GCN	81.3	79.1	71.59	87.78	74.54
	GIN	81.8	80.2	71.41	88.19	75.52
	GAT	84.2	81.1	71.75	86.27	74.04

Downstream Tasks

Node Classification

Table 1: Experiment results in unsupervised representation learning for node classification. We report Micro-F1(%) score for PPI and accuracy(%) for the other datasets.

	Dataset	Cora	CiteSeer	PubMed	Ogbn-arxiv	PPI	Reddit
Supervised	GCN	81.5	70.3	79.0	71.74±0.29	75.7±0.1	95.3±0.1
	GAT	83.0±0.7	72.5±0.7	79.0±0.3	72.10±0.13	97.30±0.20	96.0±0.1
Self-supervised	GAE	71.5±0.4	65.8±0.4	72.1±0.5	-	-	-
	GPT-GNN	80.1±1.0	68.4±1.6	76.3±0.8	-	-	-
	GATE	83.2±0.6	71.8±0.8	80.9±0.3	-	-	-
	DGI	82.3±0.6	71.8±0.7	76.8±0.6	70.34±0.16	63.80±0.20	94.0±0.10
	MVGRL	83.5±0.4	73.3±0.5	80.1±0.7	-	-	-
	GRACE[1]	81.9±0.4	71.2±0.5	80.6±0.4	71.51±0.11	69.71±0.17	94.72±0.04
	BGRL[1]	82.7±0.6	71.1±0.8	79.6±0.5	71.64±0.12	73.63±0.16	94.22±0.03
	InfoGCL	83.5±0.3	**73.5±0.4**	79.1±0.2	-	-	-
	CCA-SSG[1]	84.0±0.4	73.1±0.3	81.0±0.4	71.24±0.20	73.34±0.17	95.07±0.02
	GraphMAE	**84.2±0.4**	73.4±0.4	**81.1±0.4**	**71.75±0.17**	**74.50±0.29**	**96.01±0.08**

Graph Classification

Table 2: Experiment results in unsupervised representation learning for graph classification. We report accuracy(%) for all datasets.

	Dataset	IMDB-B	IMDB-M	PROTEINS	COLLAB	MUTAG	REDDIT-B	NCI1
Supervised	GIN	75.1±5.1	52.3±2.8	76.2±2.8	80.2±1.9	89.4±5.6	92.4±2.5	82.7±1.7
	DiffPool	72.6±3.9	-	75.1±3.5	78.9±2.3	85.0±10.3	92.1±2.6	-
Graph Kernels	WL	72.30±3.44	46.95±0.46	72.92±0.56	-	80.72±3.00	68.82±0.41	80.31±0.46
	DGK	66.96±0.56	44.55±0.52	73.30±0.82	-	87.44±2.72	78.04±0.39	80.31±0.46
Self-supervised	graph2vec	71.10±0.54	50.44±0.87	73.30±2.05	-	83.15±9.25	75.78±1.03	73.22±1.81
	Infograph	73.03±0.87	49.69±0.53	74.44±0.31	70.65±1.13	89.01±1.13	82.50±1.42	76.20±1.06
	GraphCL	71.14±0.44	48.58±0.67	74.39±0.45	71.36±1.15	86.80±1.34	89.53±0.84	77.87±0.41
	JOAO	70.21±3.08	49.20±0.77	74.55±0.41	69.50±0.36	87.35±1.02	85.29±1.35	78.07±0.47
	GCC	72.0	49.4	-	78.9	-	**89.8**	-
	MVGRL	74.20±0.70	51.20±0.50	-	-	89.70±1.10	84.50±0.60	-
	InfoGCL	75.10±0.90	51.40±0.80	-	80.00±1.30	**91.20±1.30**	-	80.20±0.60
	GraphMAE	**75.52±0.66**	**51.63±0.52**	**75.30±0.39**	**80.32±0.46**	88.19±1.26	88.01±0.19	**80.40±0.30**

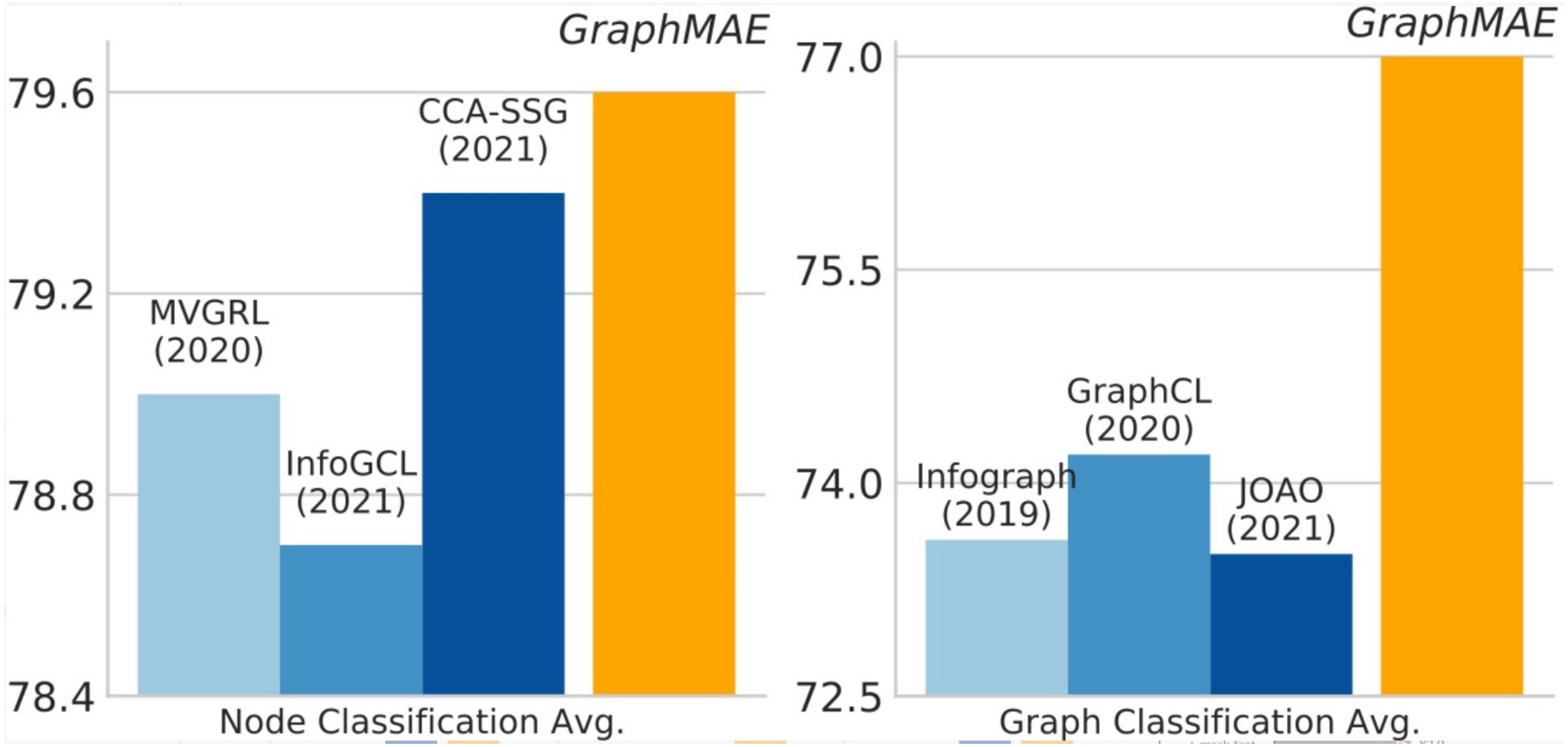
Transfer Learning

Table 3: Experiment results in transfer learning on molecular property prediction benchmarks. The model is first pre-trained on ZINC15 and then finetuned on the following datasets. We report ROC-AUC(%) scores.

	BBBP	Tox21	ToxCast	SIDER	ClinTox	MUV	HIV	BACE	Avg.
No-pretrain	65.5±1.8	74.3±0.5	63.3±1.5	57.2±0.7	58.2±2.8	71.7±2.3	75.4±1.5	70.0±2.5	67.0
ContextPred	64.3±2.8	75.7±0.7	63.9±0.6	60.9±0.6	65.9±3.8	75.8±1.7	77.3±1.0	79.6±1.2	70.4
AttrMasking	64.3±2.8	**76.7±0.4**	**64.2±0.5**	61.0±0.7	71.8±4.1	74.7±1.4	77.2±1.1	79.3±1.6	71.1
Infomax	68.8 ±0.8	75.3 ±0.5	62.7 ±0.4	58.4 ±0.8	69.9±3.0	75.3 ±2.5	76.0 ±0.7	75.9 ±1.6	70.3
GraphCL	69.7±0.7	73.9±0.7	62.4±0.6	60.5±0.9	76.0±2.7	69.8±2.7	**78.5±1.2**	75.4±1.4	70.8
JOAO	70.2±1.0	75.0±0.3	62.9±0.5	60.0±0.8	81.3±2.5	71.7±1.4	76.7±1.2	77.3±0.5	71.9
GraphLoG	**72.5±0.8**	75.7±0.5	63.5±0.7	**61.2±1.1**	76.7±3.3	76.0±1.1	77.8±0.8	**83.5±1.2**	73.4
GraphMAE	72.0±0.6	75.5±0.6	64.1±0.3	60.3±1.1	**82.3±1.2**	**76.3±2.4**	77.2±1.0	83.1±0.9	**73.8**

Code: https://github.com/THUDM/GraphMAE

GraphMAE: Masked Graph Pre-Training



Hou et al. "GraphMAE: Self-supervised masked graph autoencoders." KDD’22

However…

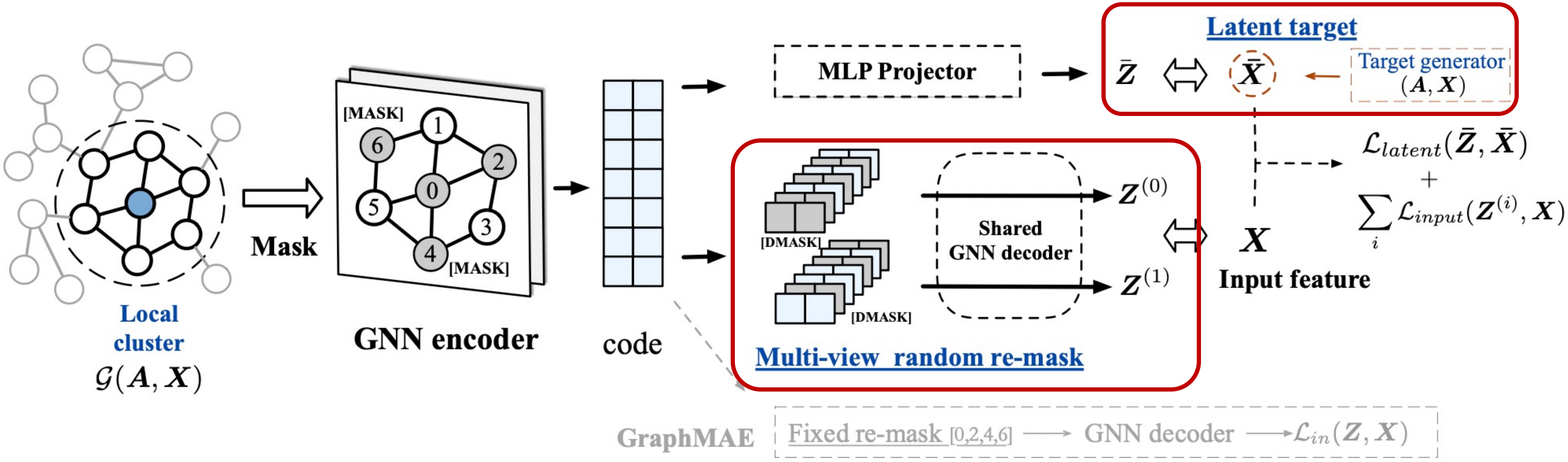
- Problems in masked-feature-prediction

  – ***more sensitive to the discriminability of input features***.

	Cora raw → *w/ PCA*	PubMed raw → *w/ PCA*
Supervised	83.0 → 82.3 (↓ 0.7)	78.0 → 77.0 (↓ 1.0)
GraphMAE	84.2 → 82.6 (↓ 1.6)	81.1 → 78.9 (↓ 2.2)
GraphMAE2	84.5 → 83.5 (↓ 1.0)	81.4 → 80.1 (↓ 1.3)

- *raw* : the original node features
- *w/ PCA* : the input features are reduced to 50-dimensional vectors using PCA

GraphMAE2



- Multi-view random re-mask decoding
- Latent representation prediction
- Scaling to large-scale graphs with local clustering

Hou et al. GraphMAE2: A Decoding-enhanced Masked Self-supervised Graph Learner. WWW’23.

Linear Probing

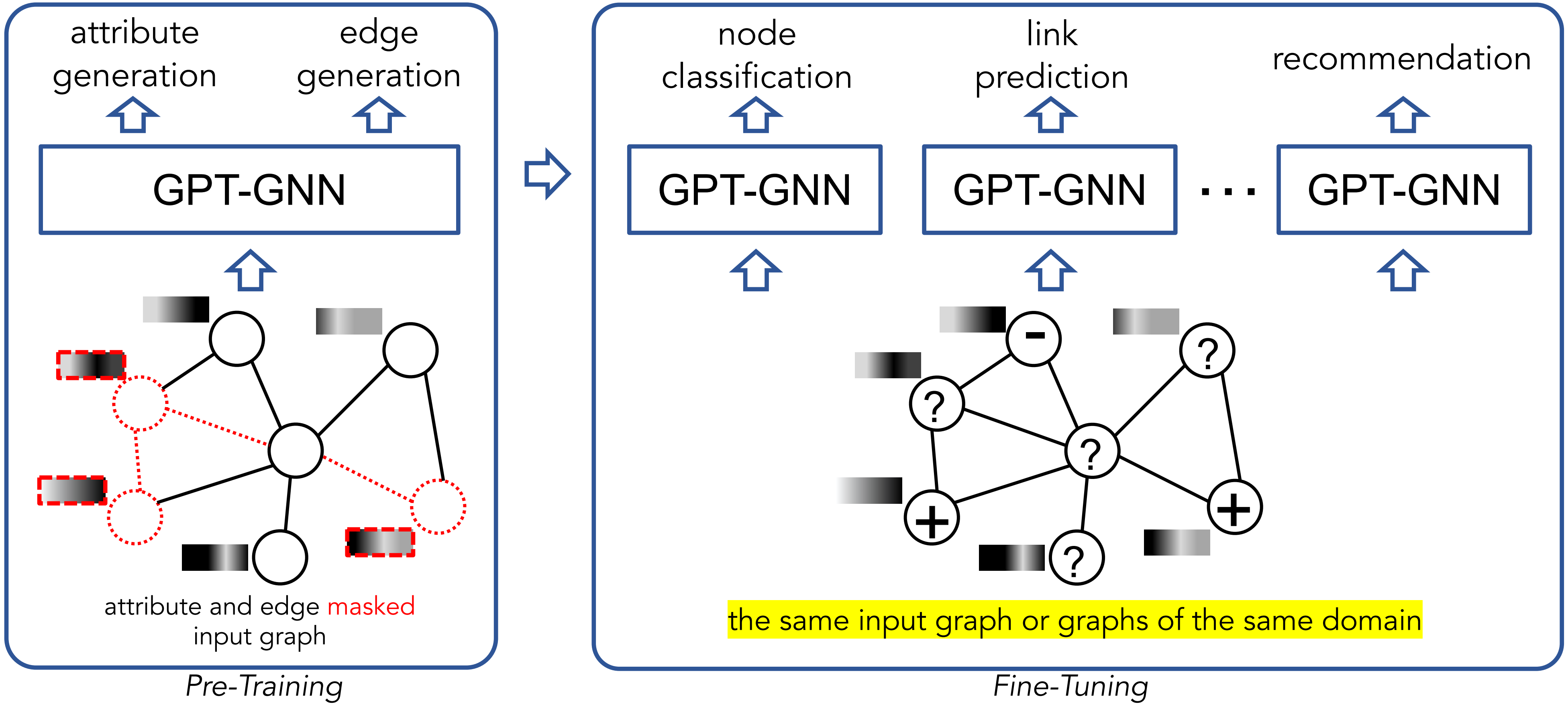
- **Setting: training a linear classifier**
- GraphMAE2 consistently outperforms all baselines
  - Improves GraphMAE on OGB benchmarks

	Arxiv	Products	MAG	Papers100M
MLP	55.50±0.23	61.06±0.08	39.11±0.21	47.24±0.31
SGC	66.92±0.08	74.87±0.25	54.68±0.23	63.29±0.19
Random-Init	68.14±0.02	74.04±0.06	56.57±0.03	61.55±0.12
CCA-SSG	68.57±0.02	75.27±0.05	51.55±0.03	55.67±0.15
GRACE	69.34±0.01	79.47±0.59	57.39±0.02	61.21±0.12
BGRL	70.51±0.03	78.59±0.02	57.57±0.01	62.18±0.15
GGD[1]	-	75.70±0.40	-	63.50±0.50
GraphMAE	71.03±0.02	78.89±0.01	58.75±0.03	62.54±0.09
GraphMAE2	**71.89**±0.03	**81.59**±0.02	**59.24**±0.01	**64.89**±0.04

OGB benchmarks

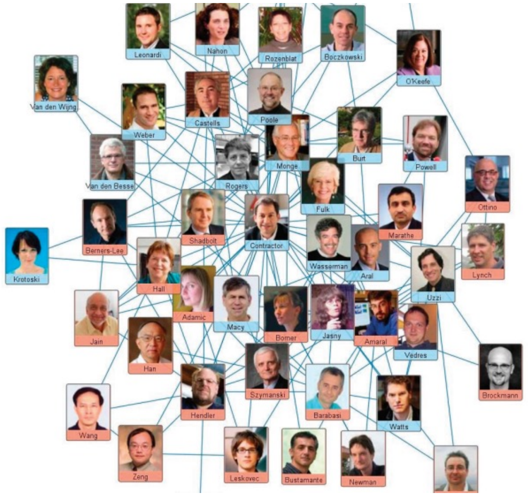
Code: https://github.com/THUDM/GraphMAE2

GNN Pre-Training on the "Same" Networks

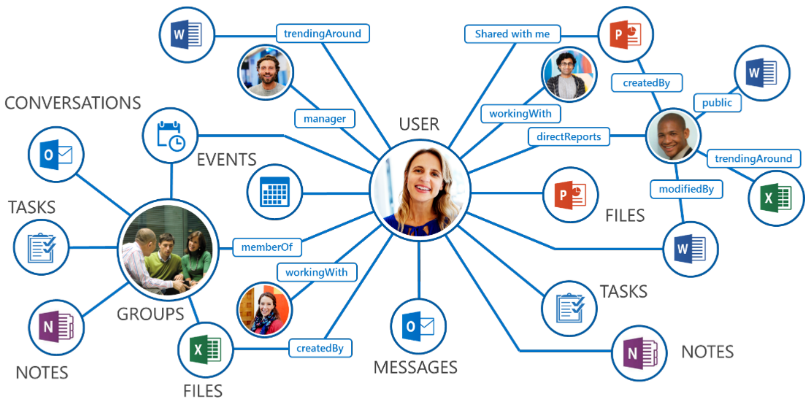


1.Ziniu Hu et al. GPT-GNN: Generative Pre-Training of Graph Neural Networks. **KDD 2020**.
2.Zhenyu Hou et al. GraphMAE: Self-supervised graph autoencoders. **KDD 2022**.
3.Zhenyu Hou et al. GraphMAE2: A Decoding-enhanced Masked Self-supervised Graph Learner. **WWW'23.**

So Many Graphs

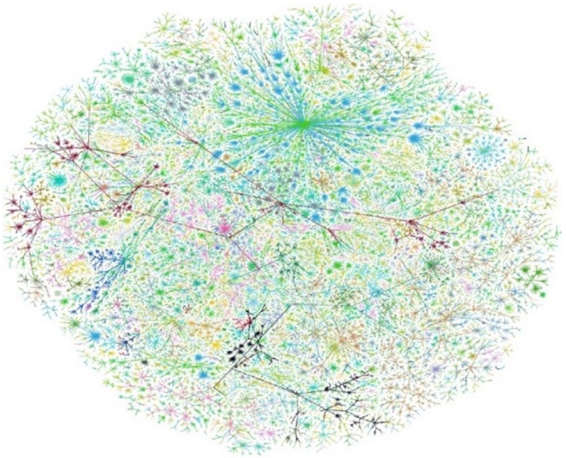


Academic Graph

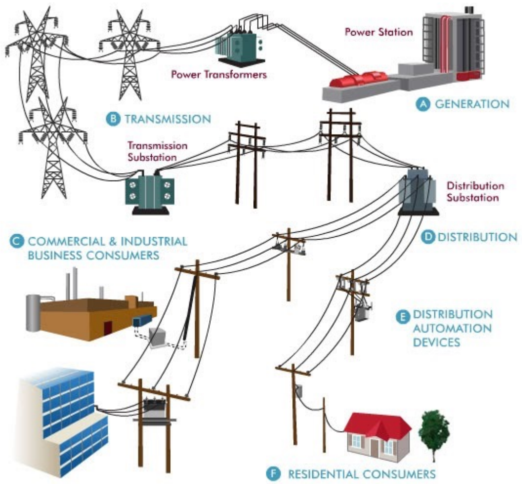


Social & Office Graph

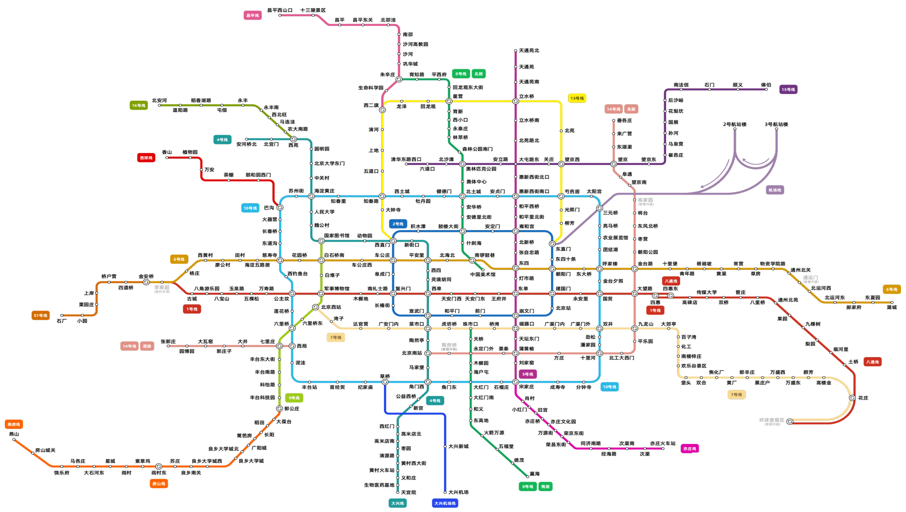
Knowledge Graph



Internet



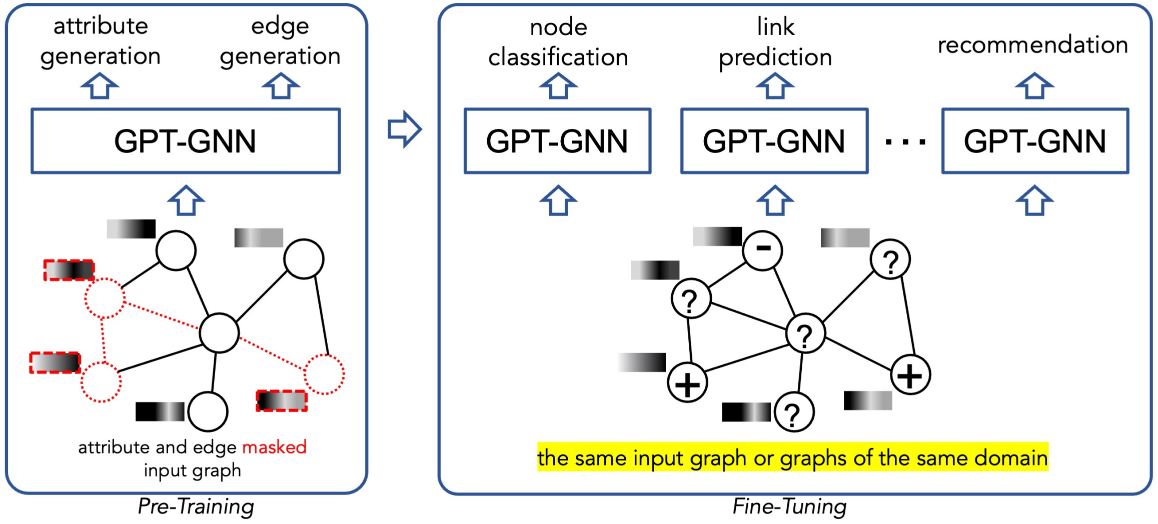
Electrical Grid Network

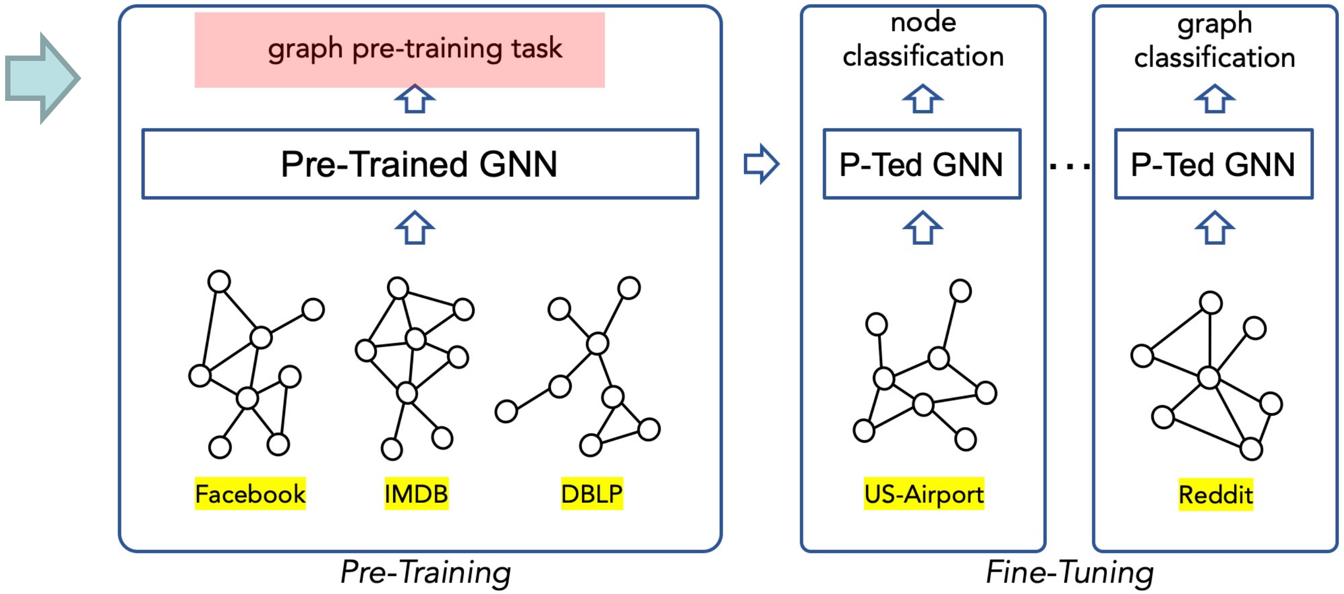


Transportation

figure credit: Web

GNN Pre-Training





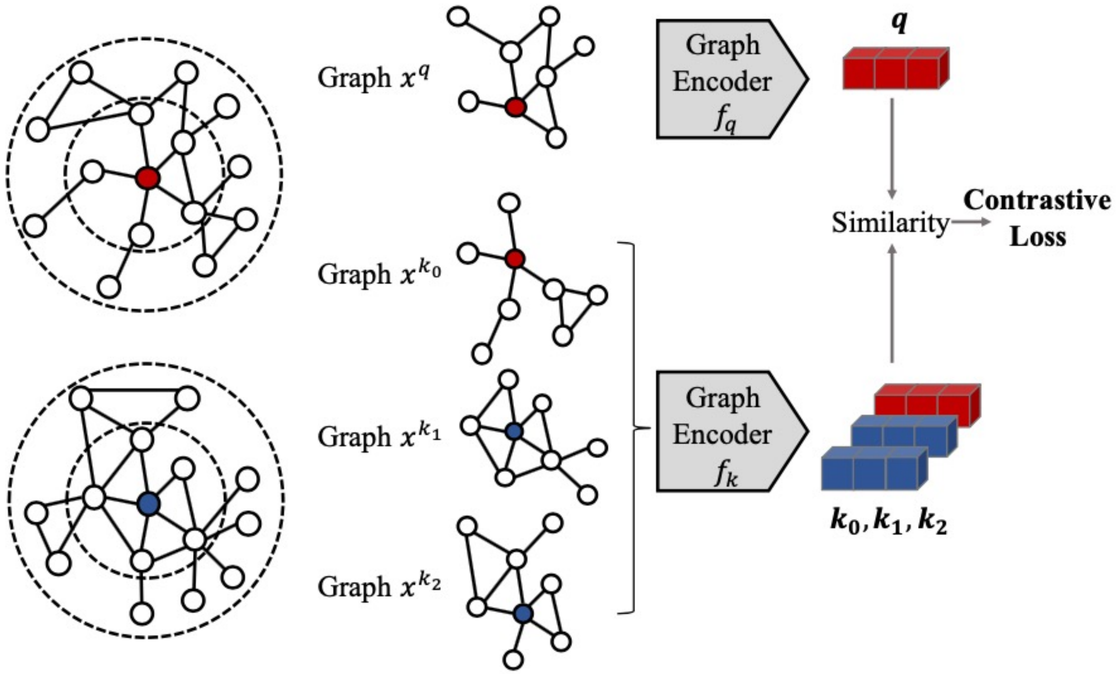
Graph Contrastive Coding (GCC)

Contrastive learning across different graphs

- Q1: How to define instances in graphs?
- Q2: How to define (dis) similar instance pairs in and across graphs?
- Q3: What are the proper graph encoders?

$$\mathcal{L} = -\log \frac{\exp\left(\boldsymbol{q}^\top \boldsymbol{k}_+ / \tau\right)}{\sum_{i=0}^{K} \exp\left(\boldsymbol{q}^\top \boldsymbol{k}_i / \tau\right)}$$

Subgraph instance discrimination



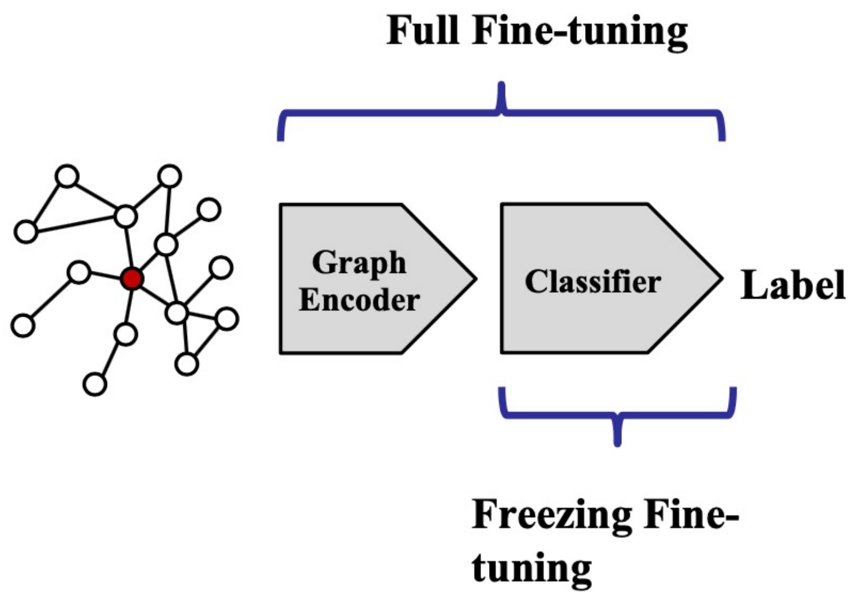
1.Jiezhong Qiu et al. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. **KDD** 2020.
2.Code & Data for GCC: https://github.com/THUDM/GCC

GCC Pre-Training / Fine-Tuning

- Pre-train on six graphs

Dataset	Academia	DBLP (SNAP)	DBLP (NetRep)	IMDB	Facebook	LiveJournal
$\|V\|$	137,969	317,080	540,486	896,305	3,097,165	4,843,953
$\|E\|$	739,384	2,099,732	30,491,458	7,564,894	47,334,788	85,691,368

- Fine-tune on **different** graphs
  - US-Airport & AMiner academic graph
    - Node classification
  - COLLAB, RDT-B, RDT-M, & IMDB-B, IMDB-M
    - Graph classification
  - AMiner academic graph
    - Similarity search
- The base GNN
  - Graph Isomorphism Network (GIN)



1.Jiezhong Qiu et al. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. **KDD** 2020.
2.Code & Data for GCC: https://github.com/THUDM/GCC

Results

Node Classification

Datasets	US-Airport	H-index
$\|V\|$	1,190	5,000
$\|E\|$	13,599	44,020
ProNE	62.3	69.1
GraphWave	60.2	70.3
Struc2vec	**66.2**	> 1 Day
GCC (E2E, freeze)	64.8	**78.3**
GCC (MoCo, freeze)	65.6	75.2
GCC (rand, full)	64.2	76.9
GCC (E2E, full)	**68.3**	80.5
GCC (MoCo, full)	67.2	**80.6**

Similarity Search

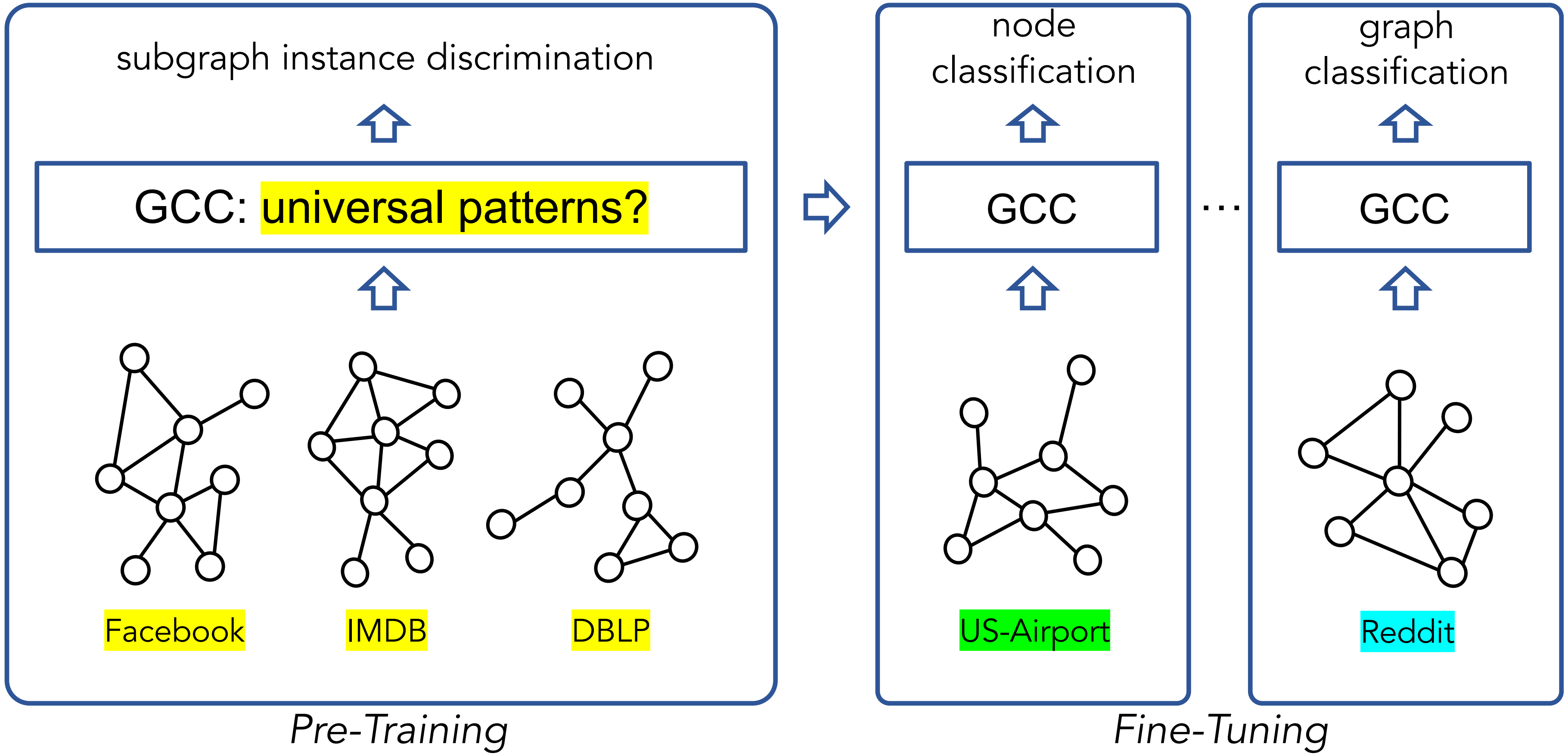
	KDD-ICDM		SIGIR-CIKM		SIGMOD-ICDE	
$\|V\|$	2,867	2,607	2,851	3,548	2,616	2,559
$\|E\|$	7,637	4,774	6,354	7,076	8,304	6,668
# groud truth		697		874		898
k	20	40	20	40	20	40
Random	0.0198	0.0566	0.0223	0.0447	0.0221	0.0521
RolX	0.0779	0.1288	0.0548	0.0984	0.0776	0.1309
Panther++	0.0892	0.1558	**0.0782**	0.1185	0.0921	0.1320
GraphWave	0.0846	**0.1693**	0.0549	0.0995	**0.0947**	**0.1470**
GCC (E2E)	**0.1047**	0.1564	0.0549	**0.1247**	0.0835	0.1336
GCC (MoCo)	0.0904	0.1521	0.0652	0.1178	0.0846	0.1425

Graph Classification

Datasets	IMDB-B	IMDB-M	COLLAB	RDT-B	RDT-M
# graphs	1,000	1,500	5,000	2,000	5,000
# classes	2	3	3	2	5
Avg. # nodes	19.8	13.0	74.5	429.6	508.5
DGK	67.0	44.6	73.1	78.0	41.3
graph2vec	71.1	50.4	–	75.8	47.9
InfoGraph	**73.0**	**49.7**	–	82.5	53.5
GCC (E2E, freeze)	71.7	49.3	74.7	87.5	52.6
GCC (MoCo, freeze)	72.0	49.4	**78.9**	**89.8**	**53.7**
DGCNN	70.0	47.8	73.7	–	–
GIN	**75.6**	**51.5**	80.2	**89.4**	**54.5**
GCC (rand, full)	**75.6**	50.9	79.4	87.8	52.1
GCC (E2E, full)	70.8	48.5	79.0	86.4	47.4
GCC (MoCo, full)	73.8	50.3	**81.1**	87.6	53.0

1. Jiezhong Qiu et al. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. **KDD** 2020.
2. Code & Data for GCC: https://github.com/THUDM/GCC

Does the Pre-Training of GNNs Learn the **Universal Structural Patterns** across Networks?



1.Jiezhong Qiu et al. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. **KDD** 2020.
2.Code & Data for GCC: https://github.com/THUDM/GCC

Does the Pre-Training of GNNs Learn the **Universal Structural Patterns** across Networks?

2014~2022
- Network Embedding [Perozzi et al.]
- Graph Conv. Networks [Kips & Welling]
- Graph Conv. N [Niepert, Defferrard et al.]

2010~2013
- Info. vs. Social (Twitter) [Kwak et al.]
- Signed Networks [Leskovec et al.]
- Semantic Social Networks [Tang et al.]
- 4 Deg. Of Separation [Backstrom et al.]
- Structural Diversity [Ugander et al.]
- Computational Social Science [Watts]

2005~2009
- Graph Evolution [Leskovec et al.]
- Social Influence Analysis [Tang et al.]
- Network Heterogeneity [Sun & Han]
- Network Embedding [Tang & Liu]
- Computer Social Science [Lazer et al.]

2000~2004
- Inf. Max'n [Domingos & Kempe et al.]
- Comm. Detection [Girvan & Newman]
- Network Motifs [Milo et al.]
- Link Pred. [Liben-Nowell & Kleinberg]

1998/9
- Small Worlds [Watts & Strogatz]
- Scale Free [Barabasi & Albert]
- Power Law [Faloutsos × 3]

1997
- HITS [Kleinberg]
- **PageRank [Page & Brin]**
- Hyperlink Vector Voting [Li]

1992
- Structural Hole [Burt]
- Dunbar's Number [Dunbar]

1970s
- The Strength Of Weak Tie [Granovetter]

1960s
- **Small Worlds [Migram]**

1950s
- Homophily [Lazarsfeld & Merton]
- Balance Theory [Heider et al.]
- Random Graph [Erdos, Renyi, Gilbert]
- Degree Sequence [Tuttle, Havel, Hakami]

1930s
- Sociogram [Moreno]

Graph Pre-Training

HGT (WWW'20) → GPT-GNN (KDD'20) → GraphMAE (KDD'22) → GraphMAE2 (WWW'23) → GCC (KDD'20)

HGT (WWW'20)
- billion-scale GNN
- most cited WWW'20
- used in MSFT

GPT-GNN (KDD'20)
- autoreg pre-train
- used in MSFT
- used in FB

GraphMAE (KDD'22)
- masked pre-train
- most cited in KDD'22

GraphMAE2 (WWW'23)
- masked pre-train
- +4% age pred in gaming

GCC (KDD'20)
- contrastive pre-train
- top cited in KDD'20

Structural Embedding

metapath2vec (KDD'17) → NetMF (WSDM'18) → NetSMF (WWW'19) → ProNE (IJCAI'19) → SketchNE (TKDE'23)

metapath2vec (KDD'17)
- hetero. NE
- most cited in KDD'18

NetMF (WSDM'18)
- NE theory of DeepW, LINE, node2vec…
- top cited WSDM'18

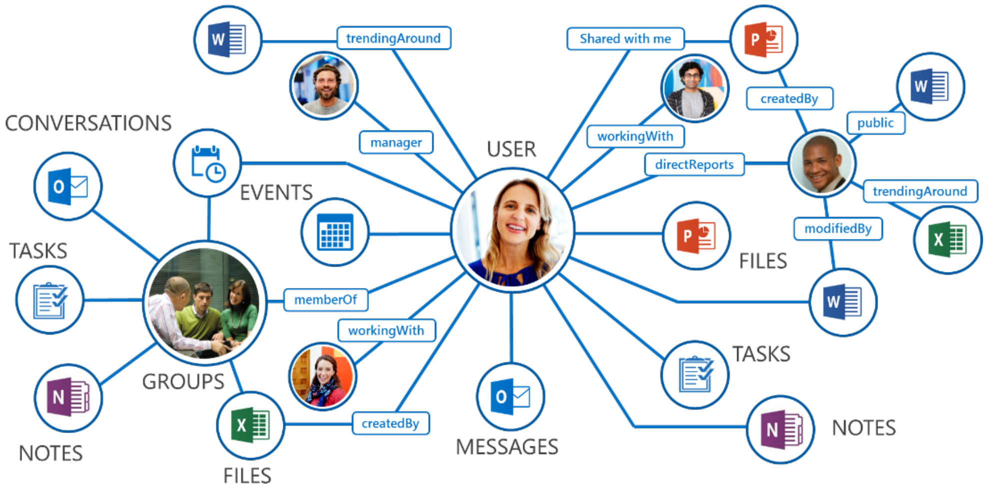
NetSMF (WWW'19)
- 100M nodes
- best paper candidate
- used in MSFT

ProNE (IJCAI'19)
- 100M nodes
- 10-400X speedup

SketchNE (TKDE'23)
- **3.5B nodes in 1 hour w/ accuracy superiority**

Pre-Train Graphs with ***Language/Image/Knowledge***

Microsoft/AMiner Academic Graph



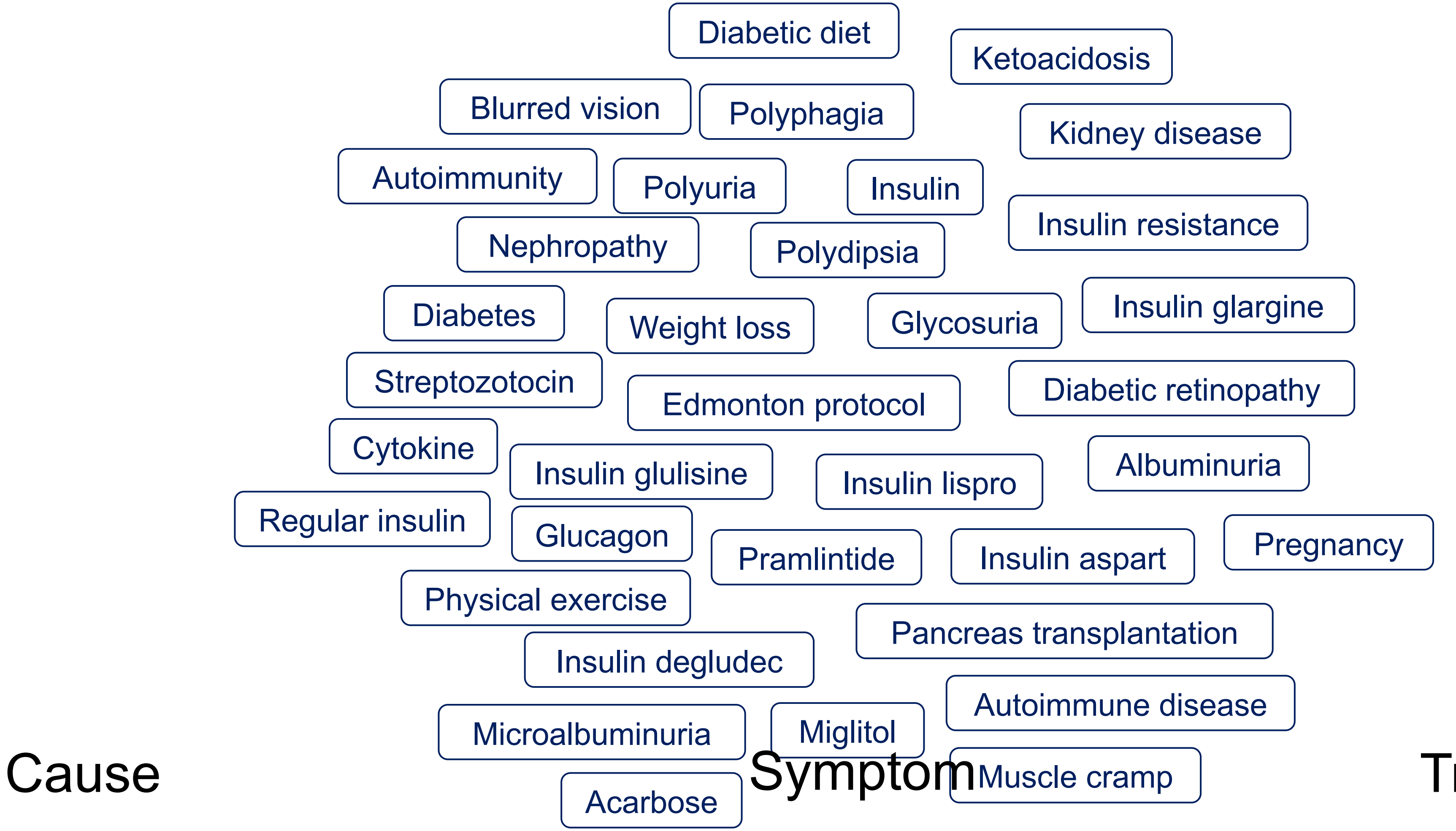
Microsoft Office Graph

LinkedIn Economic Graph

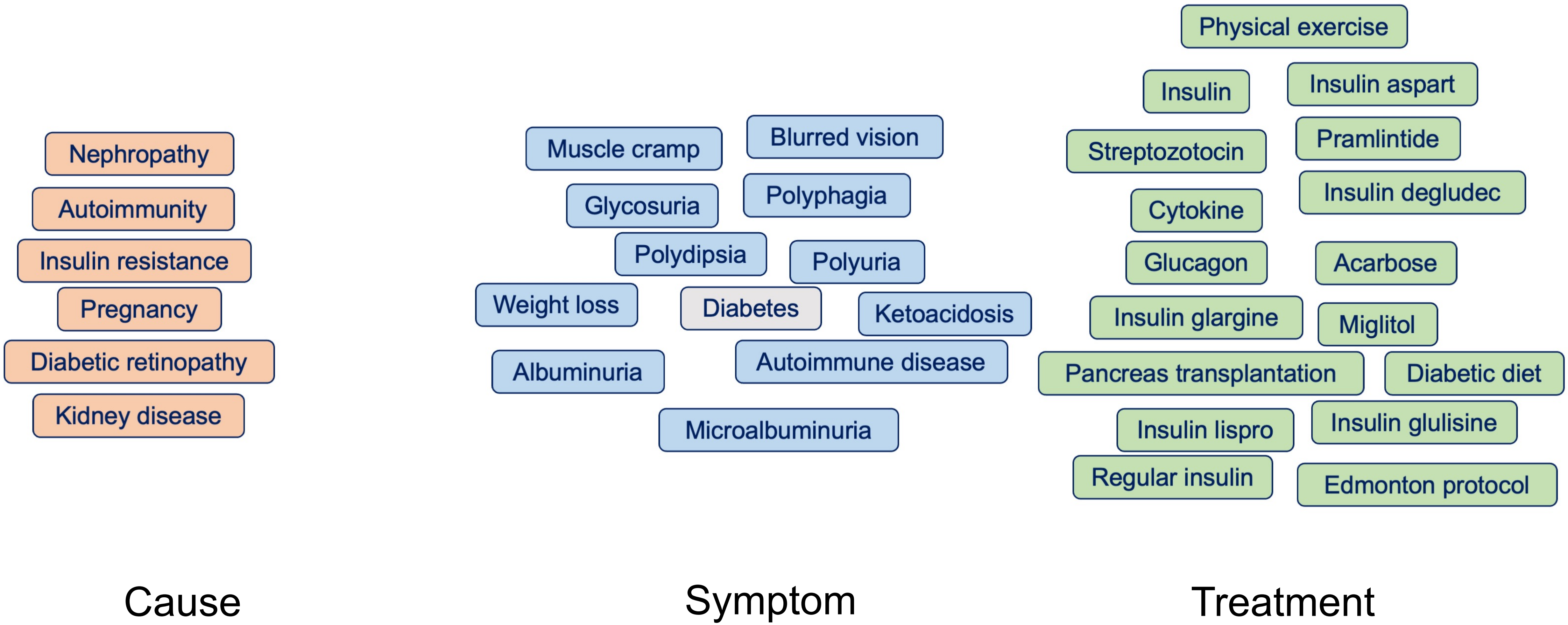
Facebook Entity Graph

Figure Credit: Microsoft/LinkedIn/Facebook

Neural Symbolic Reasoning



Neural Symbolic Reasoning



Open Research

ChatGLM2-6B Public

ChatGLM2-6B: An Open Bilingual Chat LLM | 开源双语对话语言模型

Python ☆ 9.8k Forks 1.6k

ChatGLM-6B Public

ChatGLM-6B: An Open Bilingual Dialogue Language M
开源双语对话语言模型

Python ☆ 32.4k Forks 4.3k

VisualGLM-6B Public

Chinese and English multimodal conversational language model | 多模态中英双语对话语言模型

Python ☆ 3k Forks 305

WebGLM Public

WebGLM: An Efficient Web-enhanced Question Answering System (KDD 2023)

Python ☆ 1.2k Forks 113

GLM-130B Public

GLM-130B: An Open Bilingual Pre-Trained Model (ICLR 2023)

Python ☆ 6.8k Forks 537

CodeGeeX Public

CodeGeeX: An Open Multilingual Code Generation Model (KDD 2023)

Python ☆ 6.2k Forks 457

https://github.com/THUDM

References

1. Zhenyu Hou, et al. *GraphMAE: Self-Supervised Masked Graph Autoencoders*. KDD 2022.
2. Xiao Liu, et al. *Mask and Reason: Pre-Training Knowledge Graph Transformers for Complex Logical Queries*. KDD 2022.
3. Xiao Liu, et al. *SelfKG: Self-Supervised Entity Alignment in Knowledge Graphs*. WWW 2022. **Best Paper Candidate.**
4. Wenzheng Feng, et al. *GRAND+: Scalable Graph Random Neural Networks*. WWW 2022.
5. Yukuo Cen, et al. *CogDL: A Unified Library for Graph Neural Networks*. https://cogdl.ai/.
6. Chenhui Zhang, et al. *SCR: Training Graph Neural Networks with Consistency Regularization*. arXiv.
7. Tinglin Huang, et al. *MixGCF: An Improved Training Method for Graph Neural Network-based Recommender Systems*. KDD 2021.
8. Xu Zou, et al. *TDGIA: Effective Injection Attacks on Graph Neural Networks*. KDD 2021.
9. Wenzheng Feng, et al. *Graph Random Neural Networks for Semi-Supervised Learning on Graphs*. NeurIPS 2020.
10. Weihua Hu et al. *Open Graph Benchmark: Datasets for Machine Learning on Graphs*. NeurIPS 2020.
11. Ziniu Hu et al. *GPT-GNN: Generative Pre-Training of Graph Neural Networks*. KDD 2020. *Top cited in KDD'20*
12. Jiezhong Qiu et al. *GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training*. KDD 2020. *Most cited in KDD'20*
13. Ziniu Hu et al. *Heterogeneous Graph Transformer*. WWW 2020. *Most cited in WWW'20*
14. Yuxiao Dong et al. *Heterogeneous Network Representation Learning*. IJCAI 2020.
15. Jie Zhang et al. *ProNE: Fast and Scalable Network Representation Learning*. IJCAI 2019.
16. Jiezhong Qiu et al. *NetSMF: Large-Scale Network Embedding as Sparse Matrix Factorization*. WWW 2019. **Best Paper Candidate**
17. Jiezhong Qiu et al. *Network Embedding as Matrix Factorization: Unifying DeepWalk, LINE, PTE, and node2vec.* WSDM 2018. *2nd Most cited in WSDM'18*
18. Yuxiao Dong et al. *metapath2vec: Scalable Representation Learning for Heterogeneous Networks*. KDD 2017. *Most cited in KDD'17*

Papers & code & data at https://keg.cs.tsinghua.edu.cn/yuxiao/

Thank You!

Jiezhong Qiu, Ziniu Hu, Zhenyu Hou, Wenzheng Feng, Xiao Liu

Yukuo Cen, Weihua Hu, Jie Zhang, Chenhui Zhang, Yuyang Xie

Hao Ma, Wenjian Yu, Yizhou Sun, Jure Leskovec, Jie Tang

… …

Jiezhong Qiu

Ziniu Hu

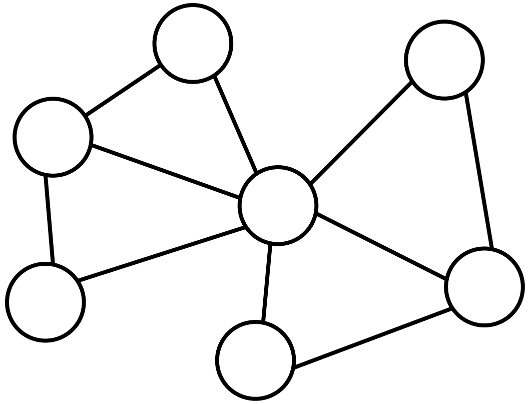
Zhenyu Hou

Papers & code & data at https://keg.cs.tsinghua.edu.cn/yuxiao/

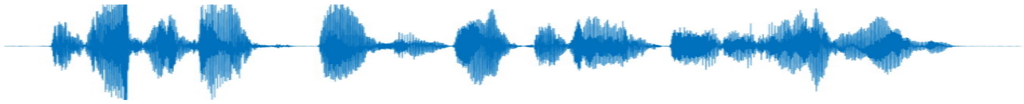
Appendix

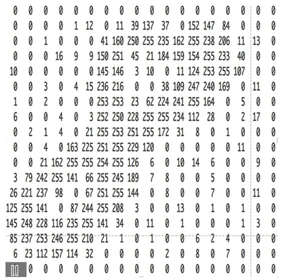
How to Encode Graph Structures?

“**Graph**, a structure made of vertices and edges”



vs.





Network Embedding: Random Walk + Skip Gram

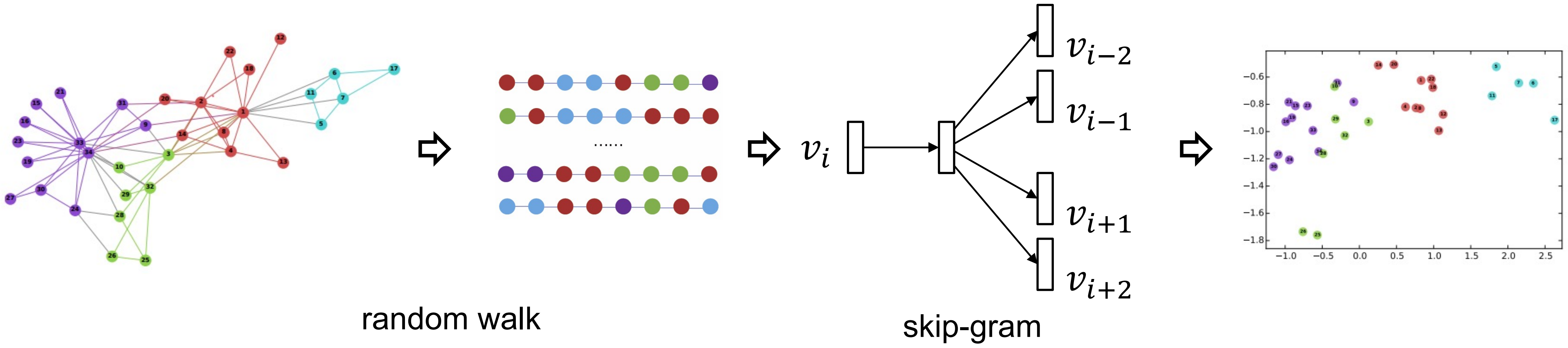
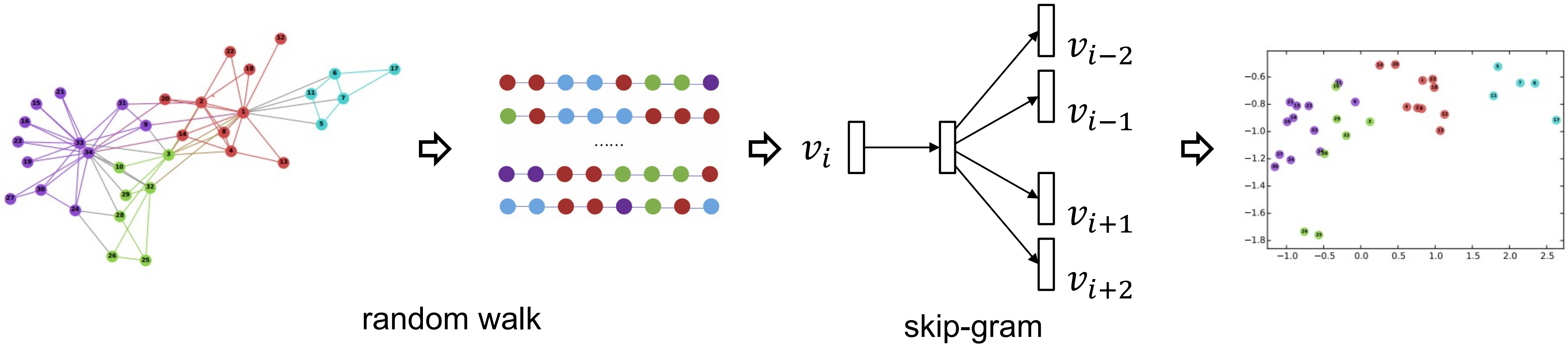


Image Credit: Perozzi et al. DeepWalk: Online learning of social representations. In *KDD' 14*, pp. 701–710.

Network Embedding: Random Walk + Skip Gram

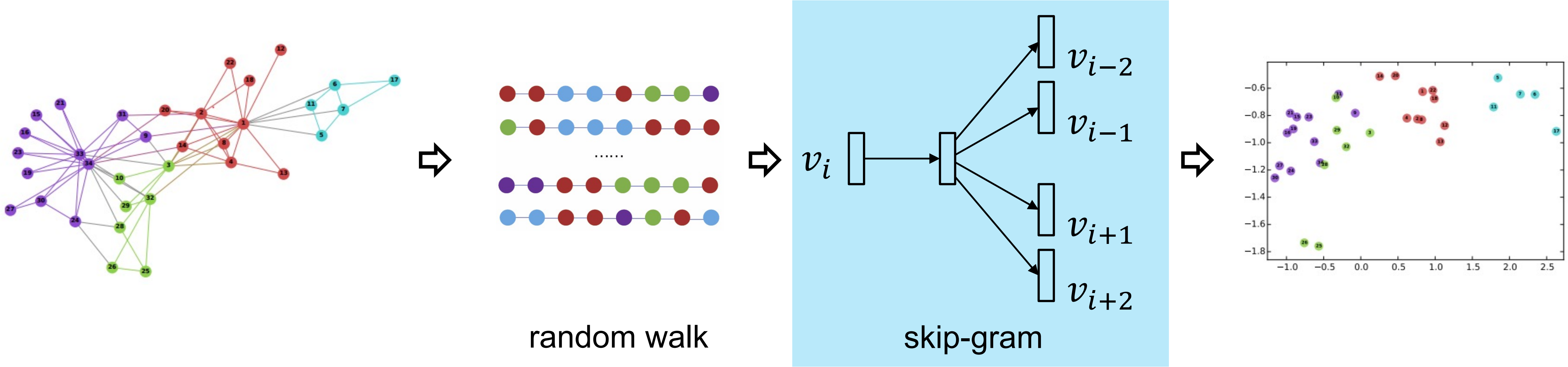


Random Walk Strategies:

- DeepWalk (walk length > 1)
- LINE (walk length = 1)
- PTE (walk length = 1)
- node2vec (biased random walk)

1. Perozzi et al. DeepWalk: Online learning of social representations. In *KDD' 14*. **Most Cited Paper in KDD'14.**
2. Tang et al. LINE: Large scale information network embedding. In *WWW'15*. **Most Cited Paper in WWW'15.**
3. Grover and Leskovec. node2vec: Scalable feature learning for networks. *In KDD'16.* **2nd Most Cited Paper in KDD'16.**

Understanding Random Walk + Skip Gram



Graph Language

- G: graph
- $\boldsymbol{A}$**:** adjacency matrix
- $\boldsymbol{D}$:degree matrix
- $vol(G)$: volume of G

?

$$\log\left(\frac{\#(\boldsymbol{w},\boldsymbol{c})|\mathcal{D}|}{b\#(w)\#(c)}\right)$$

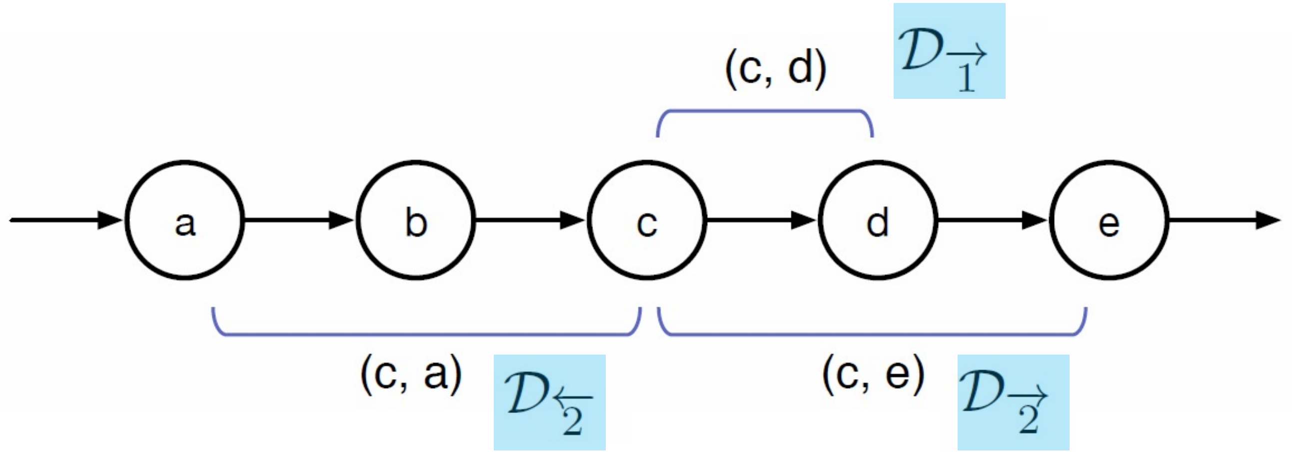
NLP Language

- #(w,c): co-occurrence of w & c
- #(w): occurrence of word w
- #(c): occurrence of context c
- $\mathcal{D}$: word−context pair (w, c) multi−set
- $|\mathcal{D}|$: number of word-context pairs

Levy and Goldberg. Neural word embeddings as implicit matrix factorization. In *NIPS 2014*

Understanding Random Walk + Skip Gram

$$\log\left(\frac{\#(w,c)\,|\mathcal{D}|}{b\#(w)\cdot\#(c)}\right)=\log\left(\frac{\frac{\#(w,c)}{|\mathcal{D}|}}{b\frac{\#(w)}{|\mathcal{D}|}\frac{\#(c)}{|\mathcal{D}|}}\right)$$



Distinguish **direction** and **distance**

NLP Language

- #(w,c): co-occurrence of w & c
- #(w): occurrence of word w
- #(c): occurrence of context c
- $\mathcal{D}$: word−context pair (w, c) multi−set
- $|\mathcal{D}|$: number of word-context pairs

- Formally, for $r = 1, 2, \cdots, T$, we define

$$\mathcal{D}_{\vec{r}} = \{(w,c) : (w,c) \in \mathcal{D}, w = w_j^n, c = w_{j+r}^n\}$$
$$\mathcal{D}_{\overleftarrow{r}} = \{(w,c) : (w,c) \in \mathcal{D}, w = w_{j+r}^n, c = w_j^n\}$$

Qiu, Dong, Ma, Li, Wang, Tang. Network embedding as matrix factorization: unifying deepwalk, line, pte, and node2vec. In *WSDM'18*

Understanding Random Walk + Skip Gram

$$\log\left(\frac{\#(w,c)\,|\mathcal{D}|}{b\#(w)\cdot\#(c)}\right) = \log\left(\frac{\frac{\#(w,c)}{|\mathcal{D}|}}{b\frac{\#(w)}{|\mathcal{D}|}\frac{\#(c)}{|\mathcal{D}|}}\right)$$

$$\frac{\#(w,c)}{|\mathcal{D}|} = \frac{1}{2T}\sum_{r=1}^{T}\left(\frac{\#(w,c)_{\overrightarrow{r}}}{|\mathcal{D}_{\overrightarrow{r}}|} + \frac{\#(w,c)_{\overleftarrow{r}}}{|\mathcal{D}_{\overleftarrow{r}}|}\right)$$

the length of random walk $L \to \infty$ $\quad \boldsymbol{P} = \boldsymbol{D}^{-1}\boldsymbol{A}$

$$\frac{\#(w,c)_{\overrightarrow{r}}}{|\mathcal{D}_{\overrightarrow{r}}|} \xrightarrow{p} \frac{d_w}{\mathrm{vol}(G)}\left(\boldsymbol{P}^r\right)_{w,c} \qquad \frac{\#(w,c)_{\overleftarrow{r}}}{|\mathcal{D}_{\overleftarrow{r}}|} \xrightarrow{p} \frac{d_c}{\mathrm{vol}(G)}\left(\boldsymbol{P}^r\right)_{c,w}$$

$$\frac{\#(w,c)}{|\mathcal{D}|} \xrightarrow{p} \frac{1}{2T}\sum_{r=1}^{T}\left(\frac{d_w}{\mathrm{vol}(G)}\left(\boldsymbol{P}^r\right)_{w,c} + \frac{d_c}{\mathrm{vol}(G)}\left(\boldsymbol{P}^r\right)_{c,w}\right)$$

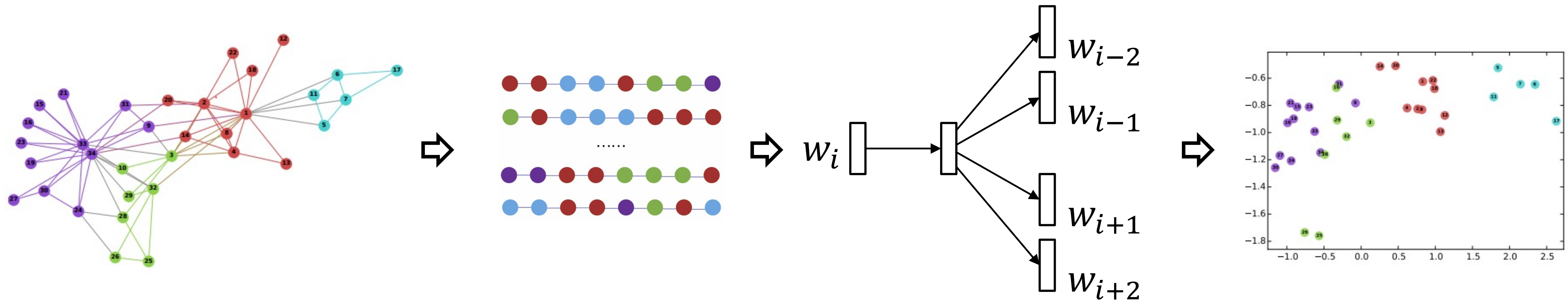
$$\frac{\#(w)}{|\mathcal{D}|} \xrightarrow{p} \frac{d_w}{\mathrm{vol}(G)} \qquad \frac{\#(c)}{|\mathcal{D}|} \xrightarrow{p} \frac{d_c}{\mathrm{vol}(G)}$$

$$\frac{\frac{\#(w,c)}{|\mathcal{D}|}}{\frac{\#(w)}{|\mathcal{D}|}\cdot\frac{\#(c)}{|\mathcal{D}|}} \xrightarrow{p} \frac{\frac{1}{2T}\sum_{r=1}^{T}\left(\frac{d_w}{\mathrm{vol}(G)}\left(\boldsymbol{P}^r\right)_{w,c} + \frac{d_c}{\mathrm{vol}(G)}\left(\boldsymbol{P}^r\right)_{c,w}\right)}{\frac{d_w}{\mathrm{vol}(G)}\cdot\frac{d_c}{\mathrm{vol}(G)}}$$

$$= \mathrm{vol}(G)\left(\frac{1}{T}\sum_{r=1}^{T}\boldsymbol{P}^r\right)\boldsymbol{D}^{-1}.$$

Qiu, Dong, Ma, Li, Wang, Tang. Network embedding as matrix factorization: unifying deepwalk, line, pte, and node2vec. In *WSDM'18*

Understanding Random Walk + Skip Gram



DeepWalk is *asymptotically and implicitly* factorizing

$$\log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right)$$

$\boldsymbol{A}$ Adjacency matrix

$\boldsymbol{D}$ Degree matrix

$vol(G) = \sum_i \sum_j A_{ij}$

b: #negative samples

T: context window size

Qiu, Dong, Ma, Li, Wang, Tang. Network embedding as matrix factorization: unifying deepwalk, line, pte, and node2vec. In *WSDM'18*

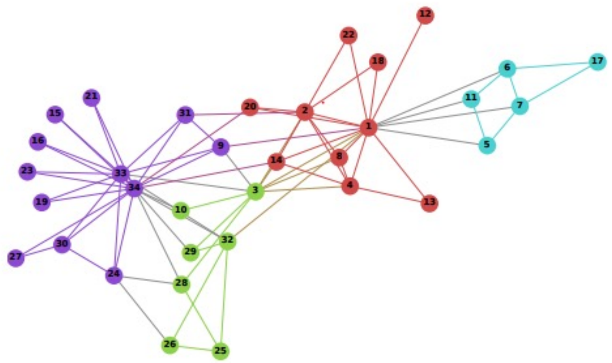
Unifying DeepWalk, LINE, PTE, & node2vec as Matrix Factorization

- DeepWalk $\log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right)$
- LINE $\log\left(\frac{\mathrm{vol}(G)}{b}\boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D}^{-1}\right)$ $T = 1$
- PTE $\log\left(\begin{bmatrix}\alpha\,\mathrm{vol}(G_{\mathrm{ww}})(\boldsymbol{D}_{\mathrm{row}}^{\mathrm{ww}})^{-1}\boldsymbol{A}_{\mathrm{ww}}(\boldsymbol{D}_{\mathrm{col}}^{\mathrm{ww}})^{-1}\\ \beta\,\mathrm{vol}(G_{\mathrm{dw}})(\boldsymbol{D}_{\mathrm{row}}^{\mathrm{dw}})^{-1}\boldsymbol{A}_{\mathrm{dw}}(\boldsymbol{D}_{\mathrm{col}}^{\mathrm{dw}})^{-1}\\ \gamma\,\mathrm{vol}(G_{\mathrm{lw}})(\boldsymbol{D}_{\mathrm{row}}^{\mathrm{lw}})^{-1}\boldsymbol{A}_{\mathrm{lw}}(\boldsymbol{D}_{\mathrm{col}}^{\mathrm{lw}})^{-1}\end{bmatrix}\right) - \log b$
- node2vec $\log\left(\frac{\frac{1}{2T}\sum_{r=1}^{T}\left(\sum_u \boldsymbol{X}_{w,u}\underline{\boldsymbol{P}}^{r}_{c,w,u} + \sum_u \boldsymbol{X}_{c,u}\underline{\boldsymbol{P}}^{r}_{w,c,u}\right)}{b\left(\sum_u \boldsymbol{X}_{w,u}\right)\left(\sum_u \boldsymbol{X}_{c,u}\right)}\right)$

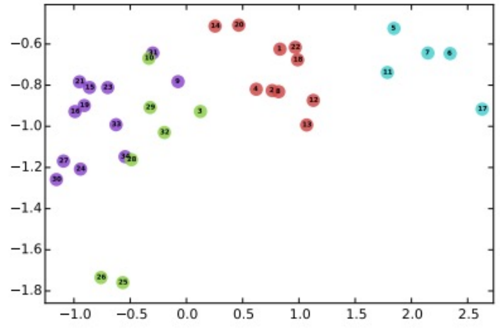
$\boldsymbol{A}$ Adjacency matrix
$\boldsymbol{D}$ Degree matrix
$vol(G) = \sum_i \sum_j A_{ij}$
b: #negative samples
T: context window size

1. Perozzi et al. DeepWalk: Online learning of social representations. In *KDD' 14*. **Most Cited Paper in KDD'14.**
2. Tang et al. LINE: Large scale information network embedding. In *WWW'15*. **Most Cited Paper in WWW'15.**
3. Grover and Leskovec. node2vec: Scalable feature learning for networks. *In KDD'16*. **2nd Most Cited Paper in KDD'16.**

NetMF: **Explicitly** Factorizing the Matrix



Matrix Factorization (NetMF)



DeepWalk is asymptotically and ***implicitly*** factorizing

$$\log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right)$$

$\boldsymbol{A}$ Adjacency matrix
$\boldsymbol{D}$ Degree matrix
$vol(G) = \sum_i \sum_j A_{ij}$
b: #negative samples
T: context window size

1. Qiu et al. Network embedding as matrix factorization: unifying deepwalk, line, pte, and node2vec. In *WSDM'18*

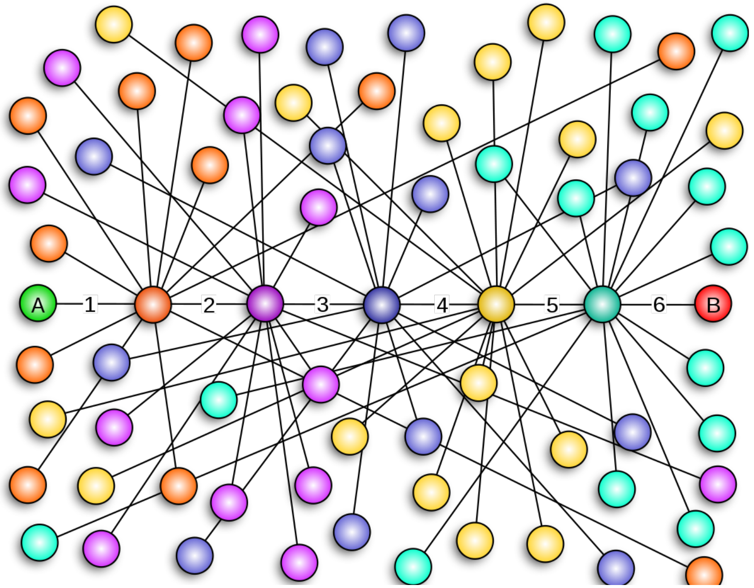
NetMF

1. Construction of $\boldsymbol{S}$
2. Factorization of $\boldsymbol{S}$

$$\boldsymbol{S} = \log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right)$$

Qiu, Dong, Ma, Li, Wang, Tang. Network embedding as matrix factorization: unifying deepwalk, line, pte, and node2vec. In *WSDM'18*

Challenge?



six (four) degrees of separation

$$\Rightarrow \quad \boldsymbol{S} = \log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right)$$

n^2 non-zeros
Dense!!

Time complexity
$O(n^3)$

How to Solve it?

NetMF

1. Construction of $\boldsymbol{S}$
2. Factorization of $\boldsymbol{S}$

NetSMF—Sparse

1. **Sparse** Construction of $\boldsymbol{S}$
2. **Sparse** Factorization of $\boldsymbol{S}$

$$\boldsymbol{S} = \log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right)$$

Qiu, Dong, Ma, Li, Wang, Wang, Tang. NetSMF: Network embedding as sparse matrix factorization. In **WWW 2019**.

Sparsify $\boldsymbol{S}$

For random-walk matrix polynomial $\boldsymbol{L} = \boldsymbol{D} - \sum_{r=1}^{T} \alpha_r \boldsymbol{D} \left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^r$

where $\sum_{r=1}^{T} \alpha_r = 1$ and α_r non-negative

One can construct a $(1+\epsilon)$-spectral sparsifier $\tilde{\boldsymbol{L}}$ with $O(n \log n \epsilon^{-2})$ non-zeros

in time $O(T^2 m \epsilon^{-2} \log n)$ for undirected graphs

$$\alpha_1 = \cdots = \alpha_T = \frac{1}{T} \quad \Rightarrow \quad \begin{aligned} \boldsymbol{S} &= \log^\circ \left(\frac{\mathrm{vol}(G)}{b} \left(\frac{1}{T} \sum_{r=1}^{T} \left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^r \right) \boldsymbol{D}^{-1} \right) \\ &= \log^\circ \left(\frac{\mathrm{vol}(G)}{b} \boldsymbol{D}^{-1} (\boldsymbol{D} - \boldsymbol{L}) \boldsymbol{D}^{-1} \right) \\ &\approx \log^\circ \left(\frac{\mathrm{vol}(G)}{b} \boldsymbol{D}^{-1} (\boldsymbol{D} - \widetilde{\boldsymbol{L}}) \boldsymbol{D}^{-1} \right) \end{aligned}$$

1. Dehua Cheng, Yu Cheng, Yan Liu, Richard Peng, and Shang-Hua Teng. Spectral sparsification of random-walk matrix polynomials. arXiv:1502.03496. 2015.

An Bounded Approximation Error

$$
\begin{aligned}
&\log^\circ\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^r\right)\boldsymbol{D}^{-1}\right)\\
&=\log^\circ\left(\frac{\mathrm{vol}(G)}{b}\boldsymbol{D}^{-1}(\boldsymbol{D}-\boldsymbol{L})\boldsymbol{D}^{-1}\right) \longrightarrow \boldsymbol{M}\\
&\approx\log^\circ\left(\frac{\mathrm{vol}(G)}{b}\boldsymbol{D}^{-1}(\boldsymbol{D}-\widetilde{\boldsymbol{L}})\boldsymbol{D}^{-1}\right) \longrightarrow \widetilde{\boldsymbol{M}}
\end{aligned}
$$

Theorem

The singular value of $\widetilde{\boldsymbol{M}}-\boldsymbol{M}$ *satisfies*

$$\sigma_i(\widetilde{\boldsymbol{M}}-\boldsymbol{M}) \le \frac{4\epsilon}{\sqrt{d_i d_{\min}}}, \forall i \in [n].$$

Theorem

Let $\|\cdot\|_F$ *be the matrix Frobenius norm. Then*

$$\left\|\mathrm{trunc\_log}^\circ\left(\frac{\mathrm{vol}(G)}{b}\widetilde{\boldsymbol{M}}\right)-\mathrm{trunc\_log}^\circ\left(\frac{\mathrm{vol}(G)}{b}\boldsymbol{M}\right)\right\|_F \le \frac{4\epsilon\,\mathrm{vol}(G)}{b\sqrt{d_{\min}}}\sqrt{\sum_{i=1}^{n}\frac{1}{d_i}}.$$

NetSMF

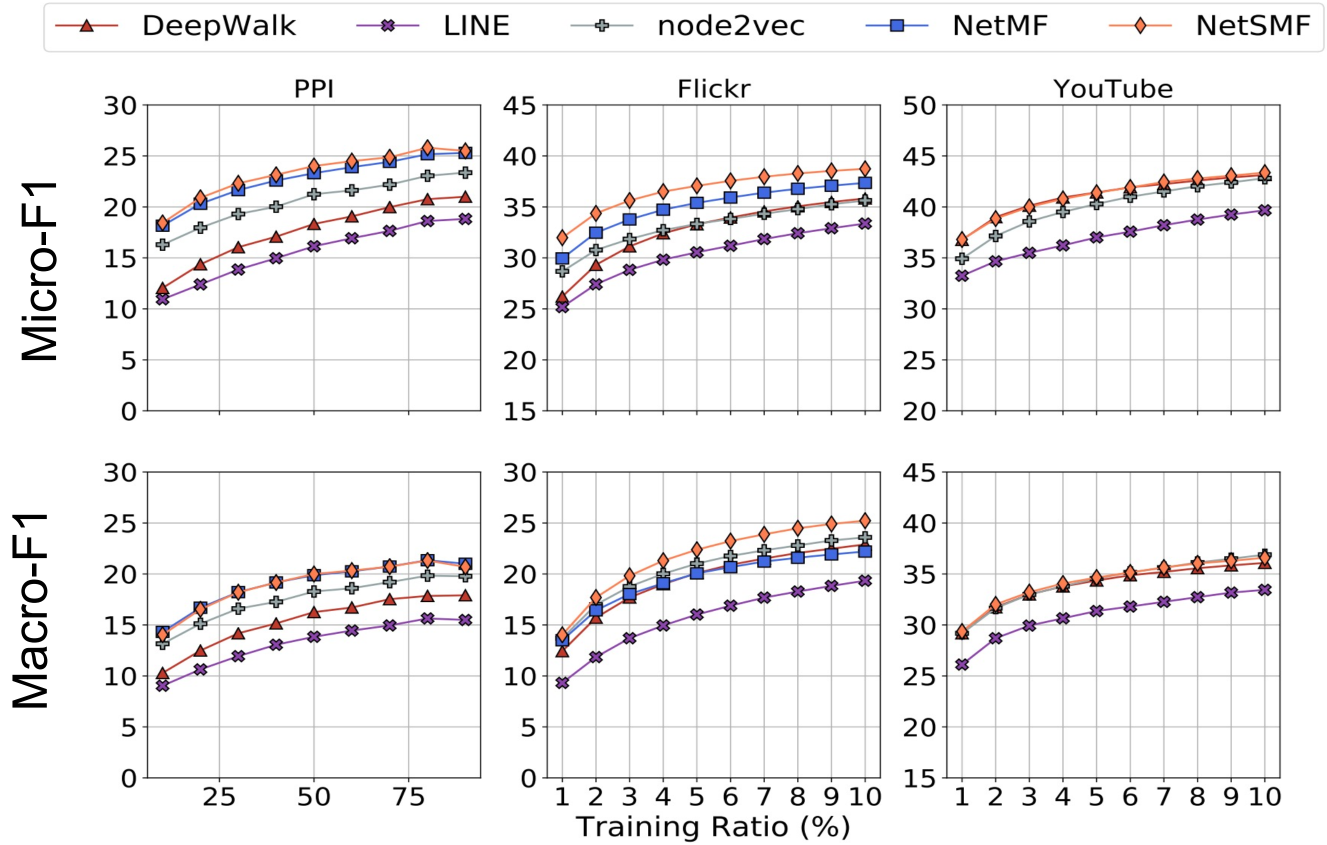
- Construct a random walk matrix polynomial sparsifier, $\widetilde{\boldsymbol{L}}$
- Construct a NetMF matrix sparsifier.

$$\operatorname{trunc\_log}^{\circ}\left(\frac{\operatorname{vol}(G)}{b}\boldsymbol{D}^{-1}(\boldsymbol{D}-\widetilde{\boldsymbol{L}})\boldsymbol{D}^{-1}\right)$$

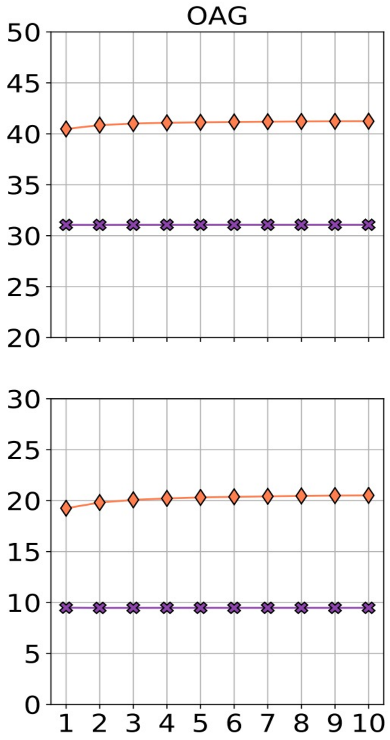
- Factorize the constructed matrix

Qiu, Dong, Ma, Li, Wang, Wang, Tang. NetSMF: Network embedding as sparse matrix factorization. In **WWW 2019**.

Results



~100 million nodes
~1 billion edges



30% improvements over LINE (Micro-F1)

100% improvements over LINE (Macro-F1)

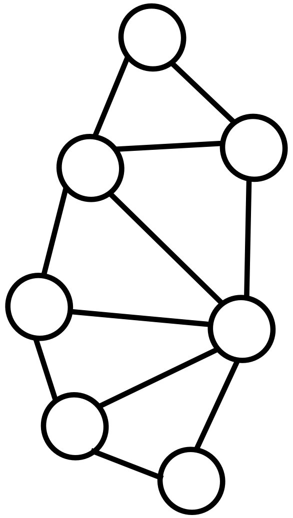
- **Effectiveness:** NetMF (explicit MF) **≈** NetSMF (sparse MF) **>** DeepWalk/LINE (implicit MF)
- **Scalability:** NetSMF can handle billion-scale network embedding

Yuyang Xie
Tsinghua

SketchNE: Embedding 3.5B nodes (225B edges) in **1 hour**

$$\log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right)$$

$$\boldsymbol{L}=\frac{\mathrm{vol}(G)}{bT}\boldsymbol{D}^{-1/2}\boldsymbol{U}_k,\ \boldsymbol{R}=\left(\sum_{r=1}^{T}\boldsymbol{\Lambda}_k^{r}\right)\boldsymbol{U}_k^{\top}\boldsymbol{D}^{-1/2}$$



NetSMF

T: $O(T(T+d)m\log n+nd^2)$

S: $O(Tm\log n+m+nd)$

1.Construct *sparse* $f^{o}(\boldsymbol{M})$

2.Factorize *sparse* $f^{o}(\boldsymbol{M})$

NetMF

T: $O(\beta mk+n^2k)$

S: $O(m+n^2)$

1. Eigen-decomp. to get low rank $\boldsymbol{M}\approx\boldsymbol{LR}$

2. Construct $f^{o}(\boldsymbol{LR})$

3. Factorize $f^{o}(\boldsymbol{LR})$

Goal:
To factorize matrix $f^{o}(\boldsymbol{M})$

SketchNE

Time: $O(qmk+qnk^2)$

Space: $O(m+nk)$

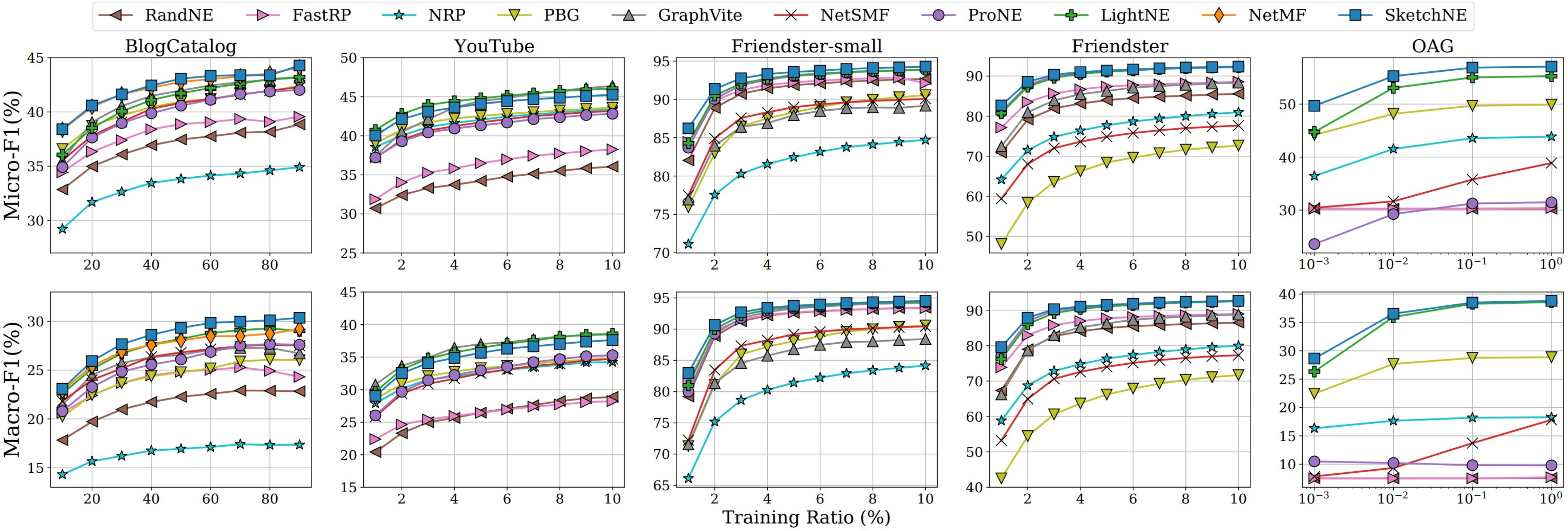
1. ***Fast*** eigen-decomp. to get low rank $\boldsymbol{M}\approx\boldsymbol{LR}$

2. ***Sparse-sign randomized single pass SVD*** to avoid explicit construction & factorization of $f^{o}(\boldsymbol{LR})$

Yuyang Xie et al., SketchNE: Embedding Billion-Scale Networks Accurately in One Hour.

Node Classification

	Multi-label Vertex Classification Task				
	BlogCatalog	YouTube	Friendster-small	Friendster	OAG
$\lvert V \rvert$	10,312	1,138,499	7,944,949	65,608,376	67,768,244
$\lvert E \rvert$	333,983	2,990,443	447,219,610	1,806,067,142	895,368,962



Yuyang Xie et al., SketchNE: Embedding Billion-Scale Networks Accurately in One Hour.

Link Prediction

Link Prediction Task			
Livejournal	ClueWeb	Hyperlink2014	Hyperlink2012
4,847,571	978,408,098	1,724,573,718	3,563,602,789
68,993,773	74,744,358,622	124,141,874,032	225,840,663,232

Datasets	Systems	MR ↓	HITS@10 ↑	HITS@50 ↑	AUC ↑
Livejournal	PBG	4.25	0.929	0.974	0.968
	GraphVite	3.06	0.962	0.982	**0.973**
	NetSMF	3.09	0.954	0.986	0.935
	RandNE	5.19	0.912	0.966	0.957
	FastRP	4.51	0.928	0.973	0.965
	NRP	2.98	0.949	0.993	0.934
	ProNE	3.31	0.950	0.982	0.932
	LightNE	2.13	**0.977**	0.993	0.945
	SketchNE	**2.10**	**0.977**	**0.994**	0.945
ClueWeb	LightNE	105.9	0.753	0.803	0.903
	SketchNE	**33.3**	**0.774**	**0.870**	**0.971**
Hyperlink2014	LightNE	129.7	0.5	0.628	0.874
	SketchNE	**110.3**	**0.593**	**0.693**	**0.890**
Hyperlink2012	LightNE	257.7	0.189	0.348	0.751
	SketchNE	**71.4**	**0.715**	**0.798**	**0.927**

Yuyang Xie et al., SketchNE: Embedding Billion-Scale Networks Accurately in One Hour.

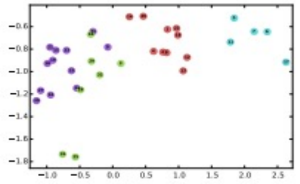
Metric	Datasets	RandNE	FastRP	NRP	PBG	GraphVite	NetSMF	ProNE	LightNE	SketchNE
Time	BlogCatalog	**0.5 s**	1.0 s	3.0 s	174.0 s	4.0 s	11.3 m	79.0 s	152.0 s	2.0 s
	YouTube	**12.0 s**	17.0 s	173.0 s	12.5 m	44.0 s	3.7 h	65.0 s	96.0 s	40.0 s
	Friendster-small	11.8 m	40.0 m	3.5 h	22.7 m	2.8 h	52 m	5.3 m	7.5 m	**5.2 m**
	Friendster	57.8 m	3.5 h	16.1 h	5.3 h	20.3 h	16.5 h	19.5 m	37.6 m	**16.0 m**
	OAG	33.7 m	3.5 h	11.4 h	20 h	1+day	22.4 h	**22.6 m**	1.5 h	1.1 h
	Livejournal	**9.0 m**	18.0 m	4.3 h	7.3 h	29.0 m	2.1 h	12.8 m	16.0 m	12.5 m
	ClueWeb	×	×	×	1+day	1+day	×	×	1.3 h	**37.7 m**
	Hyperlink2014	×	×	×	1+day	1+day	×	×	1.8 h[1]	**1.0 h**
	Hyperlink2012	×	×	×	1+day	1+day	×	×	5.6 h[1]	**1.2 h**
Mem (GB)	BlogCatalog	**0.2**	0.3	0.6	★	★	135	18	273	17
	YouTube	**6.9**	12.5	9.7	★	★	854	28	83	27
	Friendster-small	125	125	105	★	★	85	84	541	**56**
	Friendster	548	583	400	★	★	1144	326	559	**236**
	OAG	429	746	473	★	★	1500	403	1391	**283**
	Livejournal	224	417	282	★	★	140	**131**	532	147
	ClueWeb	×	×	×	★	★	×	×	1493	**612**
	Hyperlink2014	×	×	×	★	★	×	×	1500[1]	**1076**
	Hyperlink2012	×	×	×	★	★	×	×	1500[1]	**1321**

[1] LightNE requires sufficient samples that cost more than 1500GB mem (and more time), so it has to stop before it reaches the mem limit.
"★" indicates that we do not compare the memory cost of the CPU-GPU hybrid system (GraphVite) or distributed memory system (PBG).
"×" indicates that the corresponding algorithm is unable to handle the computation due to excessive space and memory consumption.

Yuyang Xie et al., SketchNE: Embedding Billion-Scale Networks Accurately in One Hour.

A Brief History of Network/Graph Embedding

$$\log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right)$$



2022 **SketchNE** [Xie et al.]

2019 NetSMF [Qiu et al.], ProNE [Zhang et al.]

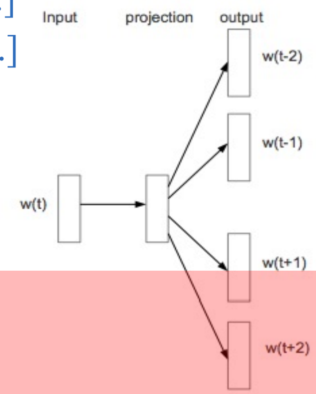
2018 **NetMF** [Qiu et al.]

2017 metapath2vec [Dong et al.]

2016 node2vec [Grover & Leskovec]

2015 LINE & PTE [Tang et al.]

2014 **DeepWalk** [Perozzi et al.]



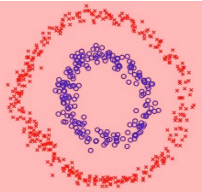
word2vec (skip-gram) [Mikolov et al.] 2013

Graph convolutional network 2009

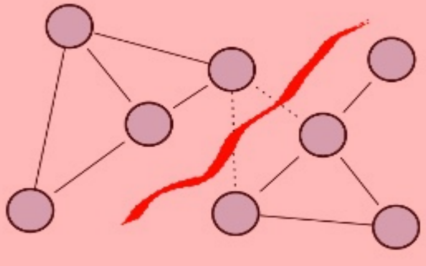
Spectral Clustering v.s. Kernel k-means [Dhillon et al.] 2005

Spectral Clustering [Ng et al.] 2002

Image Segmentation [Shi & Malik] 2000



1996 A large body of literature
[Pothen et al.] [Simon] [Bolla],
[Hagen & Kahng] [Hendrickson & Leland]
[Barnard et al.] [Spielman & Teng]



1973 Fiedler Vector [Fiedler]
Spectral Partitioning [Donath, Hoffman]

Images are extracted from related academic publications

Graph Representation Learning & Pre-Training

large

SketchNE

- embed 3.5B nodes in 1 hour

ProNE

- handle 100M nodes
- 10-400X speedup

NetSMF

- handle 100M nodes
- best candidate in WWW'19
- used in MSFT

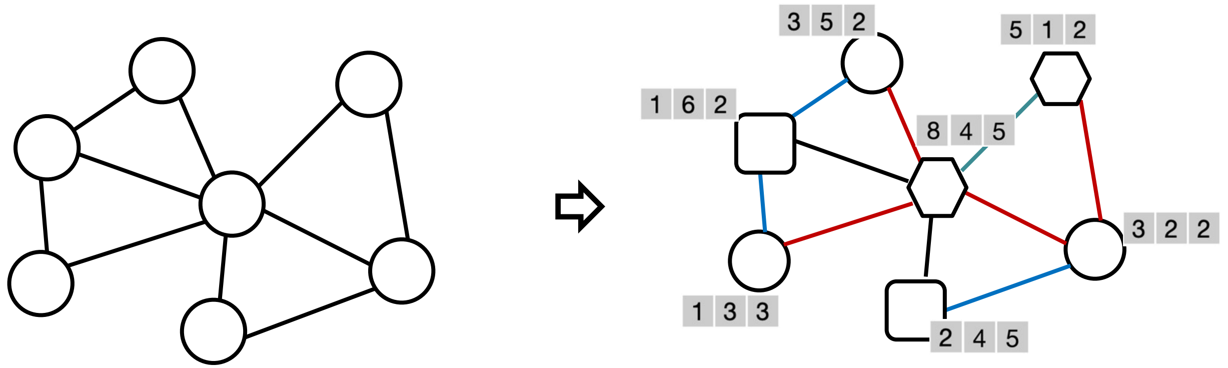
NetMF

- theoretical understanding of NE
- 2^{nd} most cited in WSDM'18

small

simple

complex



Paper & code & data at https://keg.cs.tsinghua.edu.cn/yuxiao/