Scaling Up Graph Neural Network

PRESENTER: Ningyi Liao

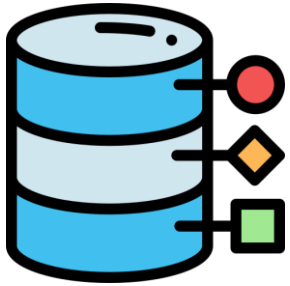
SUPERVISOR: Asst. Prof. Siqiang Luo

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Outline

Introduction

Design Scalable Algorithms



Deploy Scalable Data Structures

Evaluate Scalability Performance

Conclusion and Future Directions

Outline

Introduction

- Graph Data and Representation
- Graph Neural Netwok
- Challenge and Objectives

Algorithms

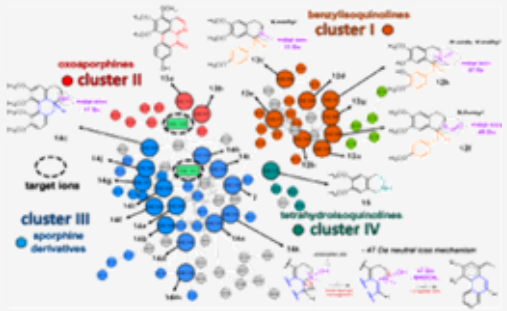
Data Structures

Evaluations

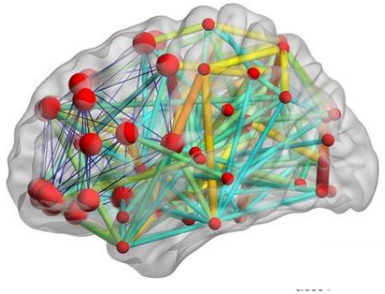
Conclusion and Future Directions

Graph: A Ubiquitous Data Structure

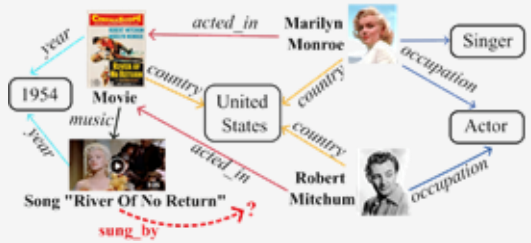
Social Networks



Molecular Networks

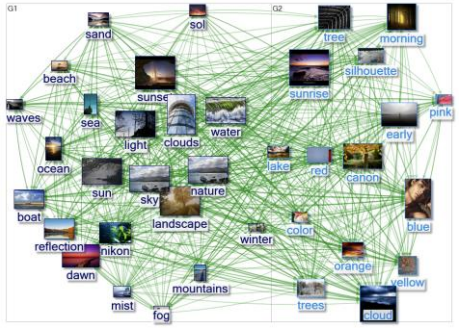


Brain Networks

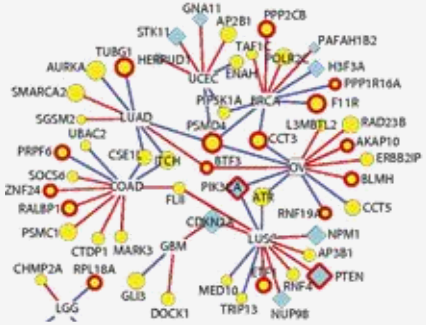


Knowledge Graphs

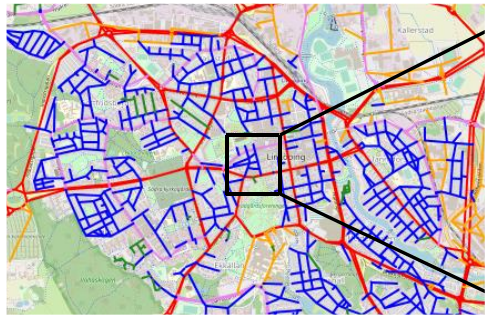
Transportation Networks



Tag Networks



Biological Networks

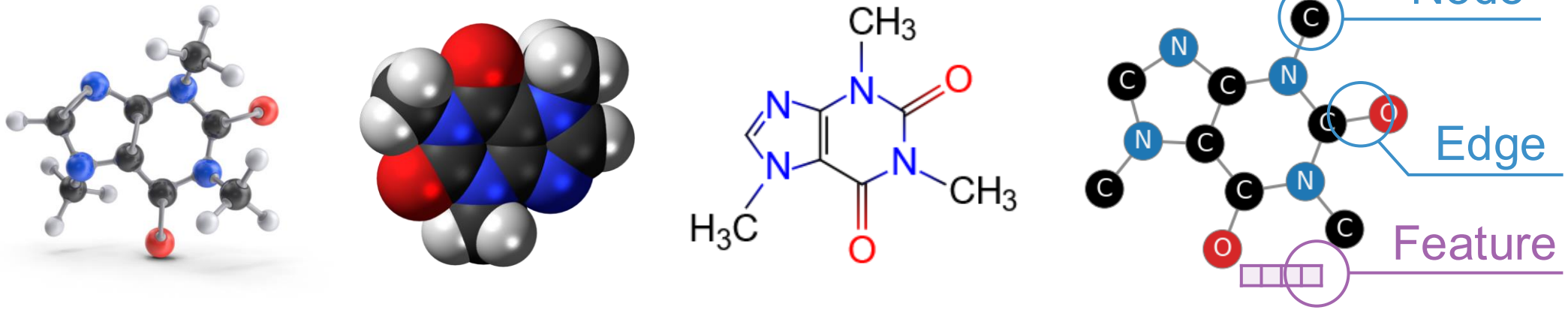


Road Networks

Image credits: pixelsquid.com, flickr.com, straitstimes.com.

Graph Representation

- Graph: entities (nodes) & relationships (edges)
- Attributed Graph: & feature vectors

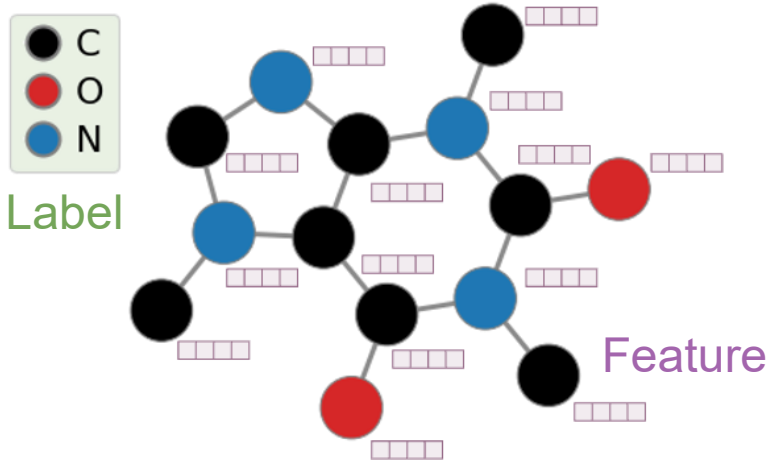


Different models of the caffeine molecule

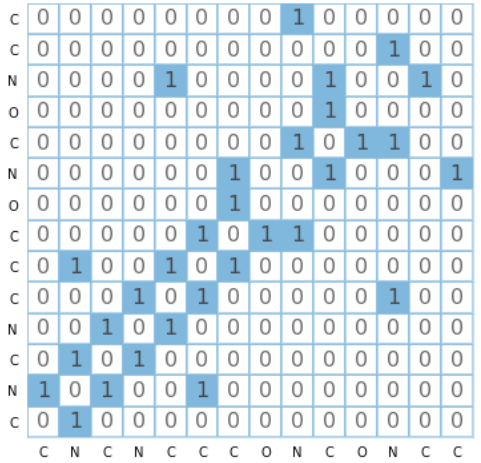
Image credits: wikipedia.org.

Graph Representation

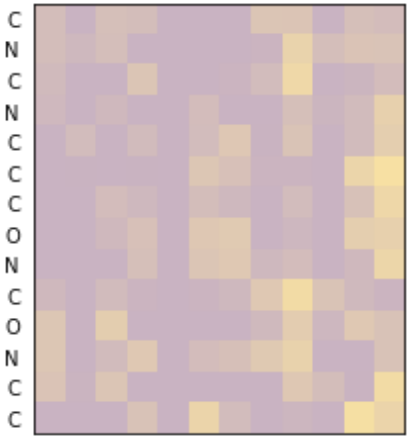
- Node Level: Feature x , Label y
- Graph Level: Adjacency matrix A , Feature matrix X



Attributed Graph



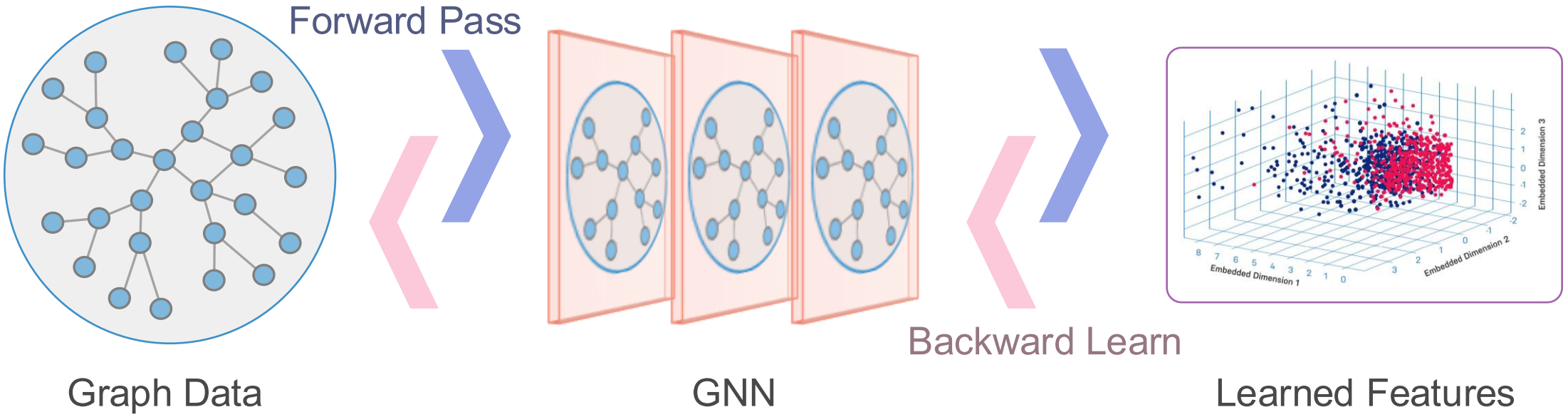
Adjacency matrix



Feature matrix

GNN: Graph Neural Network

- Design: graph data + neural network
- Data: graph data (irregular) → features (structured)



GNN: Graph Neural Network

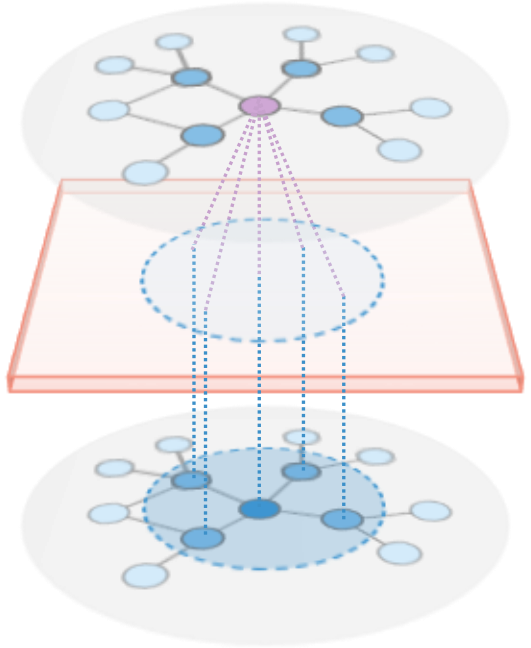
- Graph Convolution:

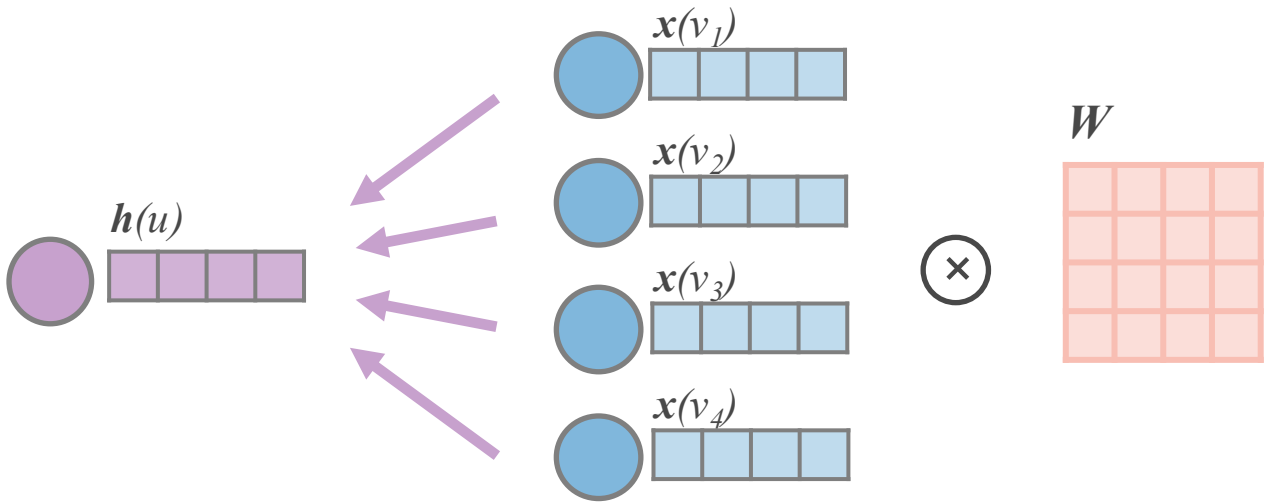
$$\boldsymbol{h}(u) = \sum_{v \in N(u)} \boldsymbol{x}(v) \cdot \boldsymbol{W}$$

Node Representation

Aggregate Neighbors

Learnable Weights



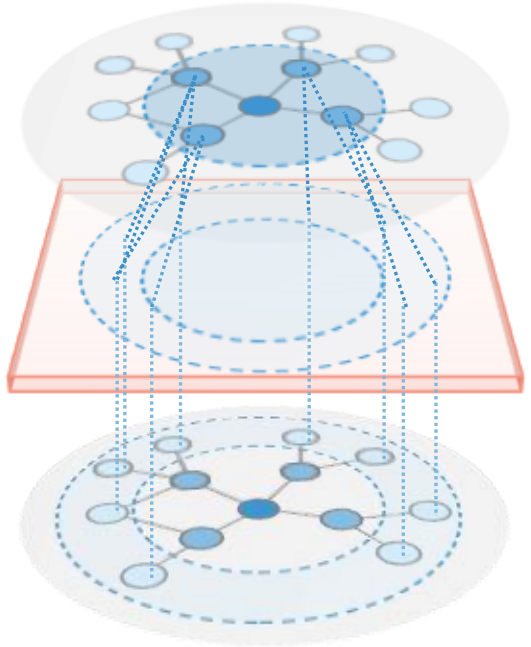


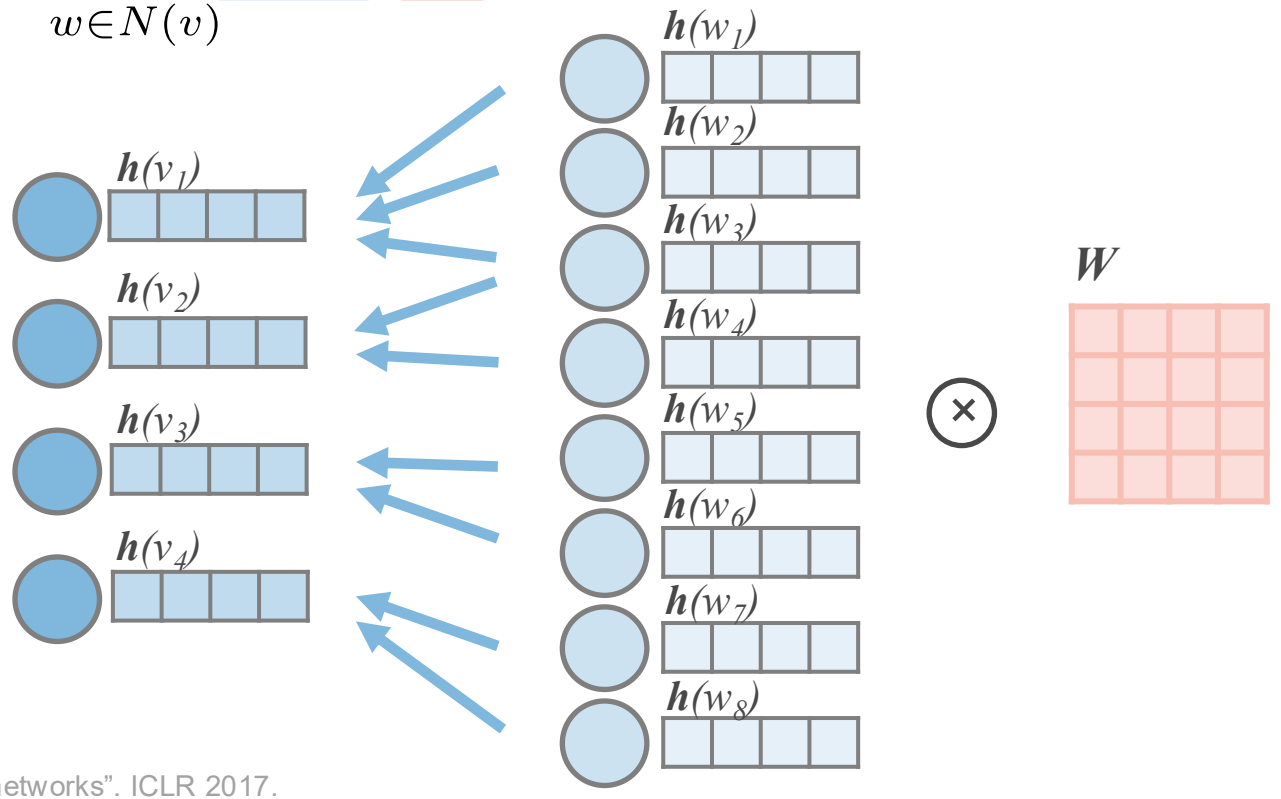
T Kipf & M Welling. “Semi-supervised classification with graph convolutional networks”. ICLR 2017.

GNN: Graph Neural Network

- Stack multi-hop graph convolutions as layers

$$\boldsymbol{h}(v) = \sum_{w \in N(v)} \boldsymbol{h}(w) \cdot \boldsymbol{W}$$

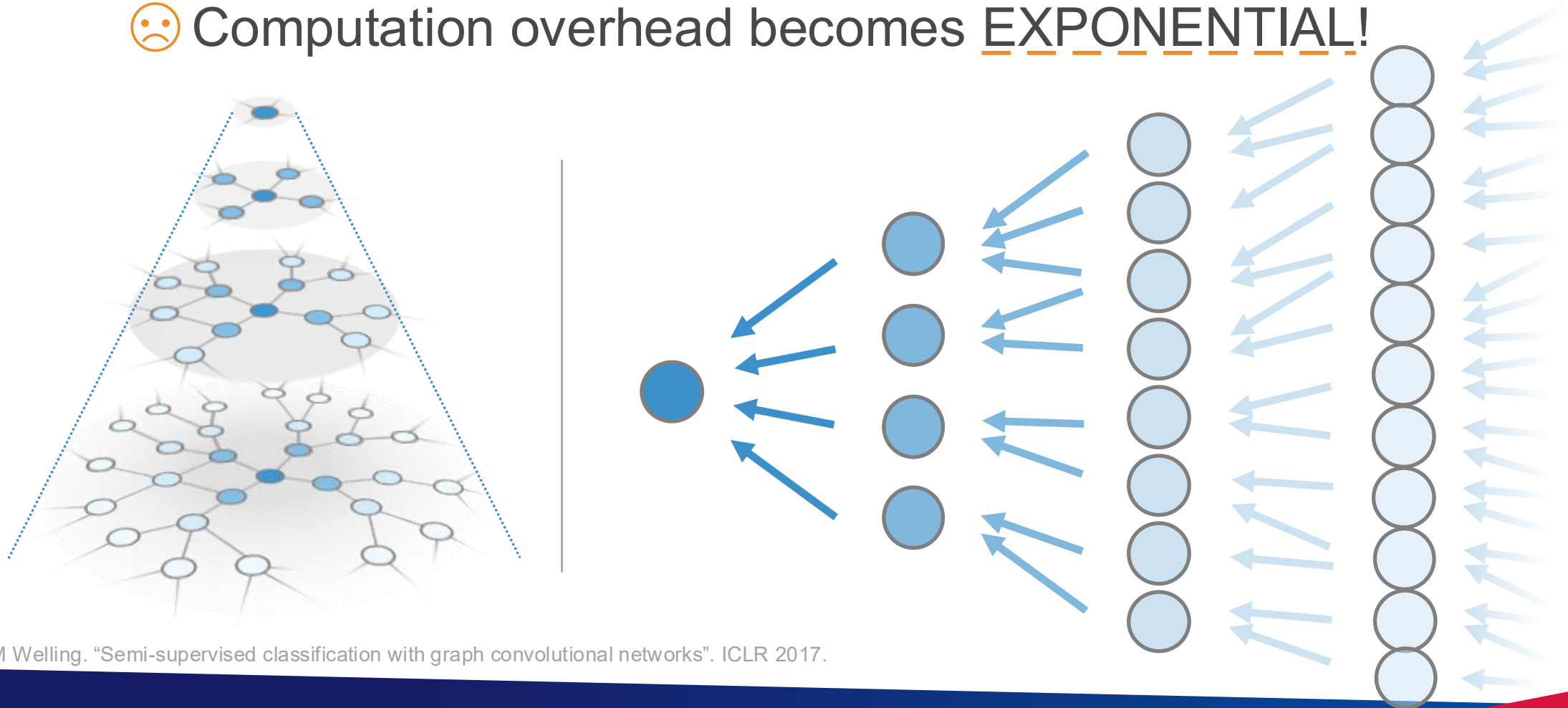




T Kipf & M Welling. “Semi-supervised classification with graph convolutional networks”. ICLR 2017.

Challenge: GNN Scalability Issue

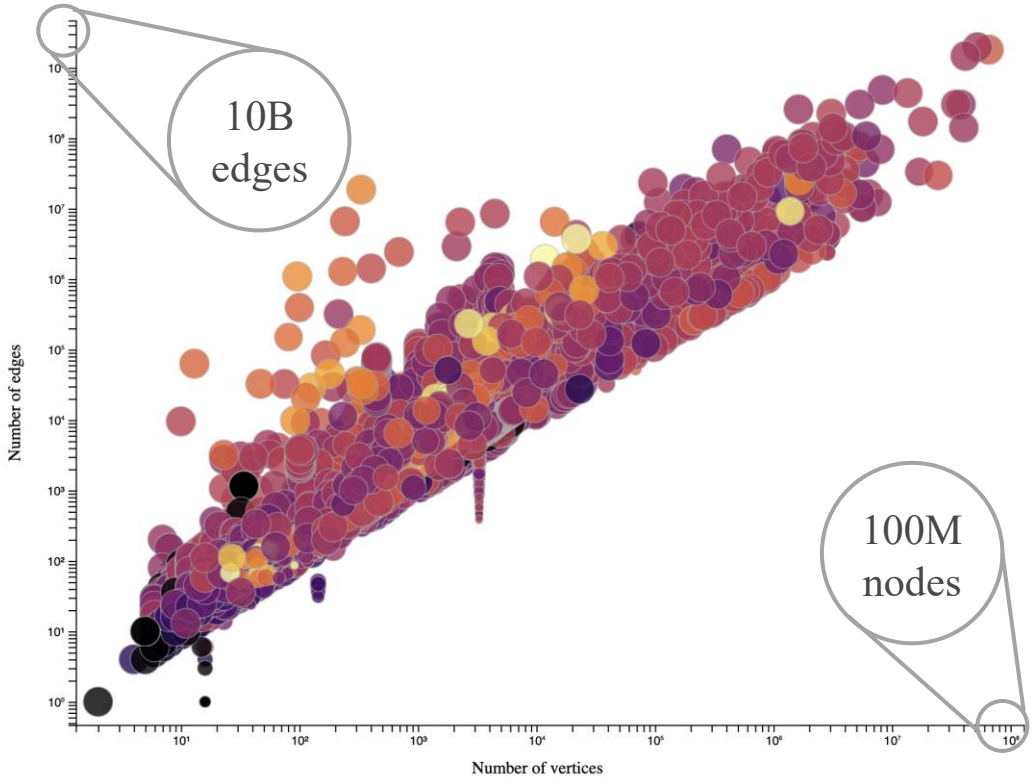
- If keep stacking more layers …
  - ☹ Computation overhead becomes EXPONENTIAL!



T Kipf & M Welling. “Semi-supervised classification with graph convolutional networks”. ICLR 2017.

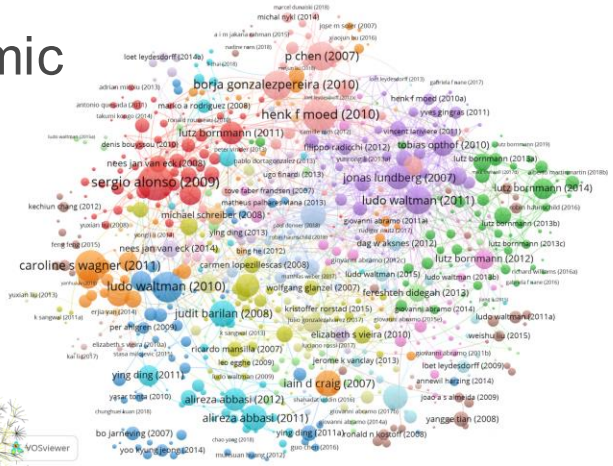
Challenge: GNN Scalability Issue

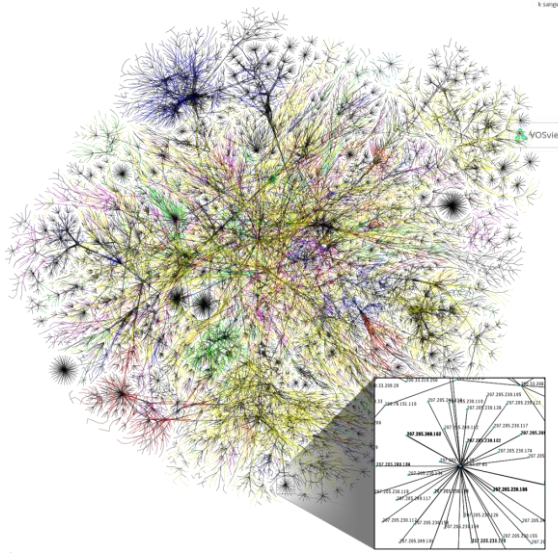
- Modern real-world graphs are on the scale of millions or billions



Scales of graph datasets

Microsoft Academic
240M nodes



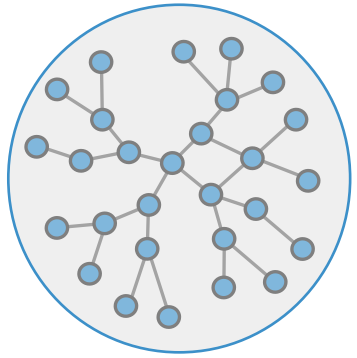


Internet IP
4B nodes

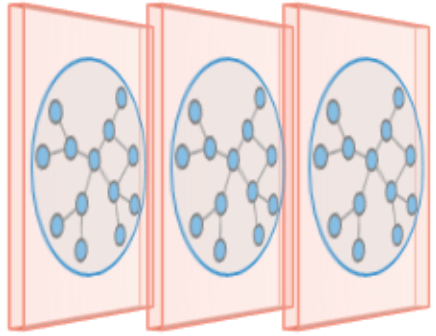
- T Peixoto, "The Netzschleuder network catalogue and repository", 2020. https://networks.skewed.de/
- JM Díaz-Nafría & T Guarda. “Is the Internet-of-Things a Burden or a Leverage for the Human Condition?” MDPI Proceedings. 2017.

Objective: Scaling Up GNN

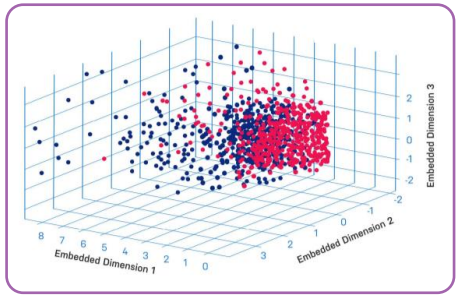
Graph Data



GNN



Learned Features



Deploy Scalable Data Structures

- Transformer model?
- Subgraph model?

Evaluate Performance

- Approximation theory?
- Efficiency benchmark?

Design Scalable Algorithms

- Scalability bottlenecks?
- Long-range convolution?

Outline

Introduction

Data Structures

Evaluations

Design Scalable Algorithms

SCARA: Feature-dimension Utilization

LD^2: Long-distance Embedding

[VLDB 2022] & [VLDB 2023]

Conclusion and Future Directions

Background: GCN Complexity

Representation

$n \times f$ Dense

Update: $H^{(l+1)} = \sigma\left(\tilde{A}H^{(l)}W^{(l)}\right)$

Learnable Weights

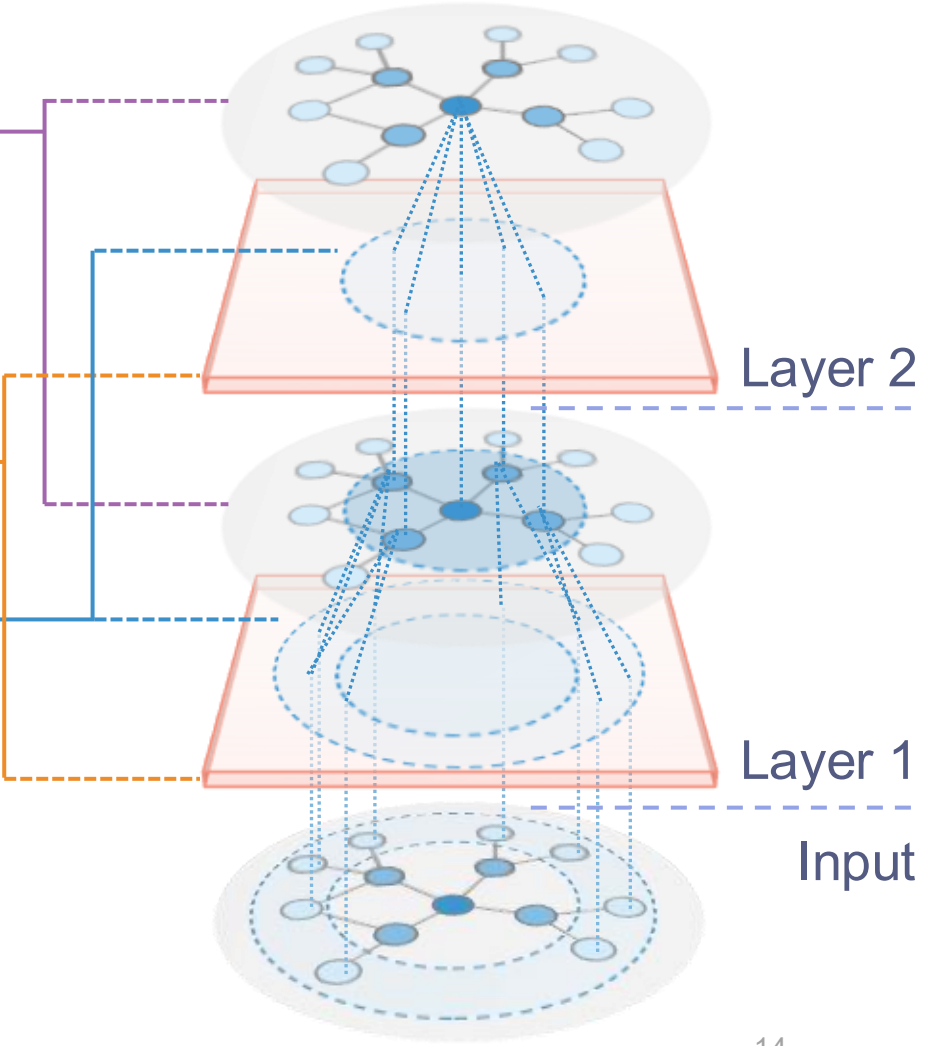
$f \times f$ Dense

Graph Aggregation

$n \times n$ Sparse

Input: $H^{(0)} = X$

Node Attribute



T Kipf & M Welling. “Semi-supervised classification with graph convolutional networks”. ICLR 2017.

Background: GCN Complexity

Each layer: $H^{(l+1)} = \sigma\left(\tilde{A}H^{(l)}W^{(l)}\right)$

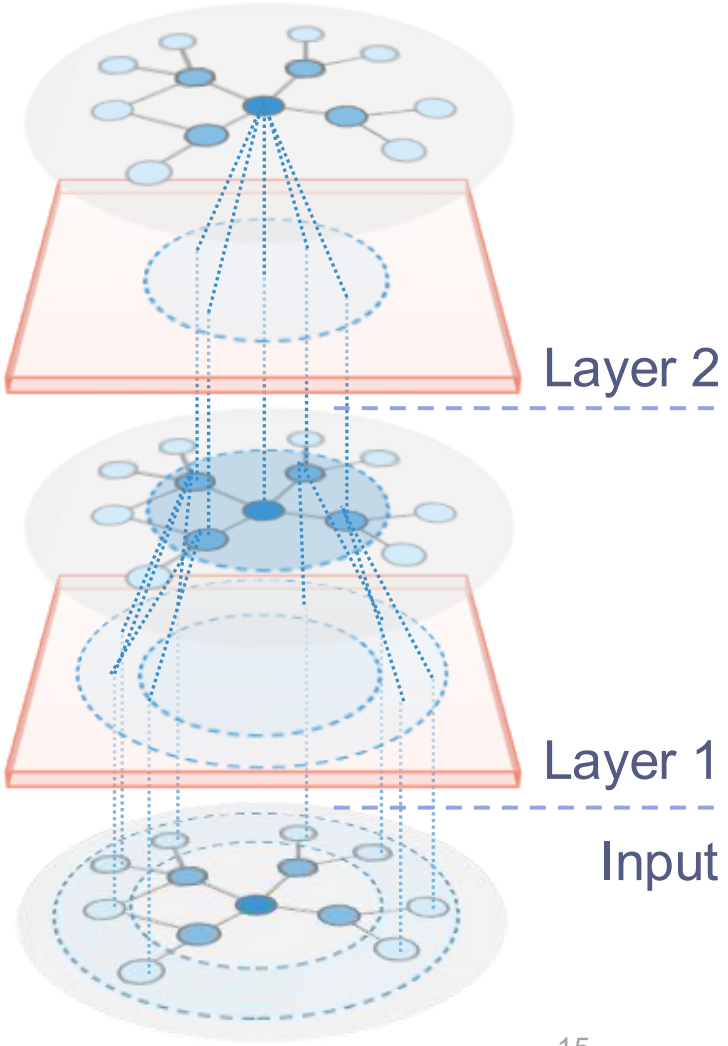
$O(nf^2)$ Weight Transformation

Dense-dense Matrix Multiplication

Graph Aggregation $O(mf)$

Sparse-dense Matrix Multiplication

Total: $O(Lmf + Lnf^2)$



T Kipf & M Welling. “Semi-supervised classification with graph convolutional networks”. ICLR 2017.

SCARA: Iterative → Decoupled Computation

- GCN:

Iterative: $L\times$

$$\boldsymbol{H}^{(l+1)} = \sigma\left(\tilde{\boldsymbol{A}}\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)}\right)$$

- SCARA (ours):

Feature

Normalize: $1\times$

$$\tilde{\boldsymbol{A}}^{l}\cdot\boldsymbol{X} = \boldsymbol{D}^{r-1}\left(\boldsymbol{A}\boldsymbol{D}^{-1}\right)^{l}\boldsymbol{D}^{1-r}\boldsymbol{X}$$

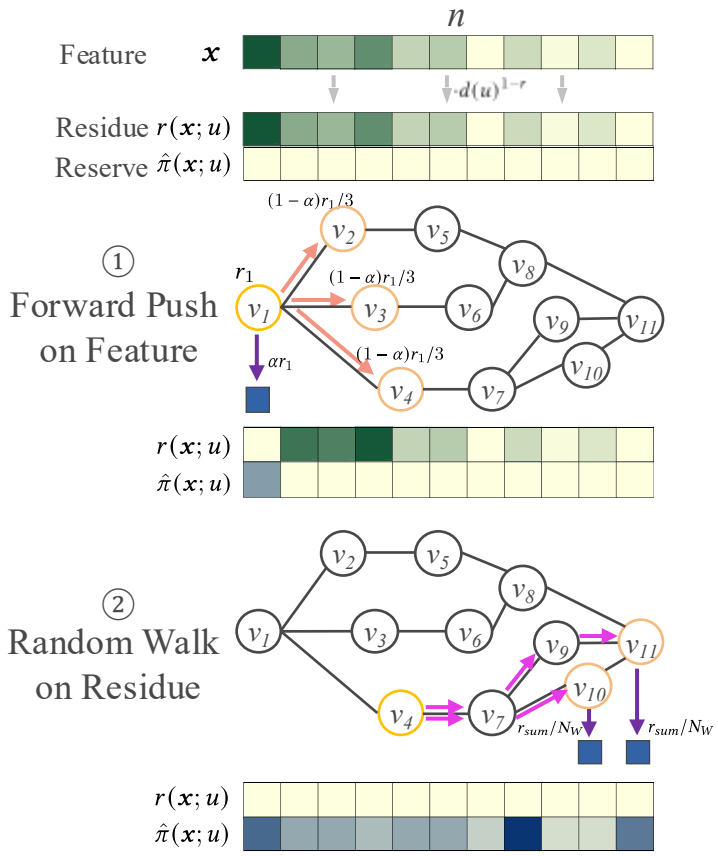
Personalized PageRank

Propagate: $1\times$

$$\boldsymbol{H}^{(0)} = \sum_{0}^{\infty}\alpha(1-\alpha)^{l}\tilde{\boldsymbol{A}}^{l}\cdot\boldsymbol{X}$$

Transform: $L\times$

$$\boldsymbol{H}^{(l+1)} = \sigma\left(\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)}\right)$$



[1] N Liao*, D Mo*, S Luo, X Li, P Yin. "SCARA: Scalable Graph Neural Networks with Feature-Oriented Optimization". **VLDB 2022**.

SCARA: Iterative → Decoupled Computation

- GCN:

Iterative: $O(LmF + LnF^2)$

☹ *Coupled graph & NN computation*

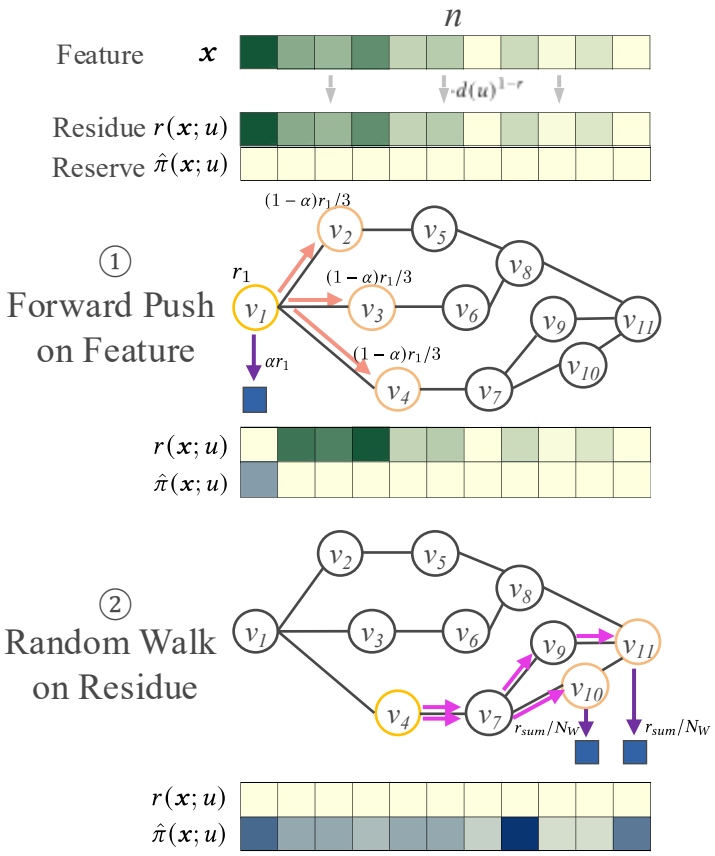
- SCARA (ours):

Propagate: $O(F\sqrt{m \log n}/\lambda)$

😀 *Sublinear CPU precomputation*

Transform: $O(LnF^2)$

😀 *Efficient GPU training*



[1] N Liao*, D Mo*, S Luo, X Li, P Yin. "SCARA: Scalable Graph Neural Networks with Feature-Oriented Optimization". **VLDB 2022**.

SCARA: Node → Feature Dimension

- GCN:

Node-wise: $\tilde{A} \cdot H = \sum_{u=1}^{n} \tilde{A}[u] \cdot H[u]$

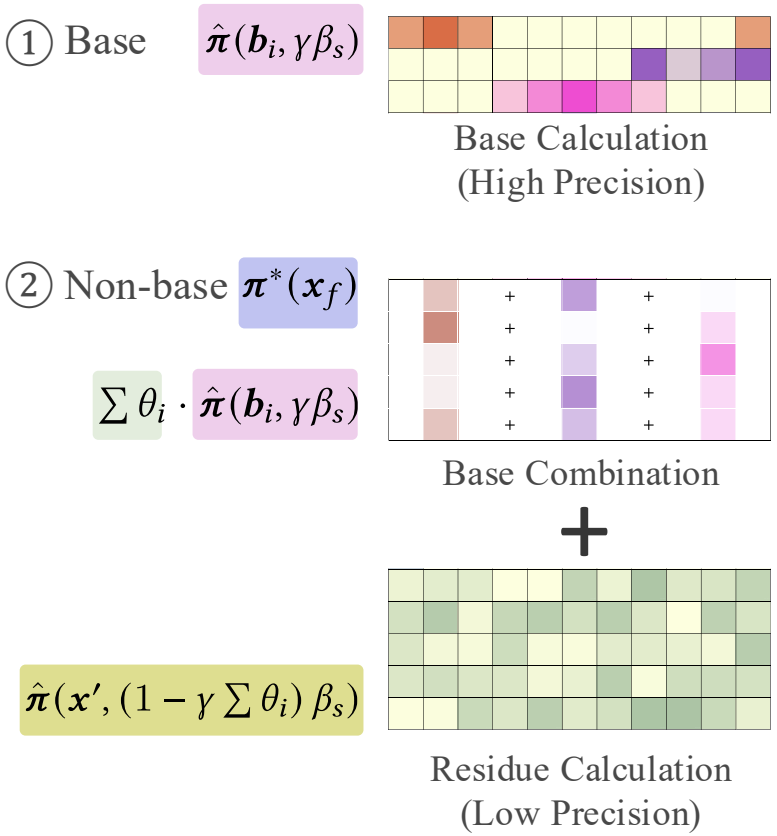
☹ *Limited memory locality*

- SCARA:

Decompose: $\min \operatorname{rank}(X_B\Theta) + \eta \|Z\|_1,$
$\text{s.t. } X_B\Theta + Z = X.$

Feature reuse: $\check{P} = \hat{P}_B \cdot \Theta + \hat{P}_Z$

☺ *Reduced compute redundancy*



[2] N Liao, D Mo, S Luo, X Li, P Yin. "Scalable Decoupling Graph Neural Network with Feature-Oriented Optimization". **VLDBJ 2023**.

SCARA: Complexity Results

- Time: sub-linear precomputation + efficient learning
- Memory: efficient GPU usage

☹ Not suitable for minibatch

☹ Not scalable to large graph m and n

☹ Insufficient memory

Model	Precomp. Time	Training Time	Precomp. Mem.	Training Mem.
GCN [3]	–	$O(ILmF + ILnF^2)$	–	$O(LnF + LF^2)$
Cluster-GCN [22]	$O(m)$	$O(ILmF + ILnF^2)$	$O(n)$	$O(Ln_bF + LF^2)$
GraphSAINT [23]	–	$O(IL_PLnF^2)$	–	$O(L_PLn_bF + LF^2)$
GAS [24]	$O(m + LnF)$	$O(ILmF + ILnF^2)$	$O(LnF)$	$O(Ldn_bF + LF^2)$
APPNP [25]	$O(m)$	$O(IL_PmF + ILnF^2)$	$O(m)$	$O(Ln_bF + LF^2 + nn_b)$
PPRGo [26]	$O(m/\delta)$	$O(IKnF + ILnF^2)$	$O(n/\delta)$	$O(Ln_bF + LF^2 + Kn_b)$
SGC [27]	$O(L_PmF)$	$O(ILnF^2)$	$O(m)$	$O(Ln_bF + LF^2)$
GBP [11]	$O(L_PF\sqrt{L_Pm\log(L_Pn)}/\epsilon)$	$O(ILnF^2)$	$O(nF)$	$O(Ln_bF + LF^2)$
SCARA (ours)	$O(F\sqrt{m\log n}/\lambda)$	$O(ILnF^2)$	$O(nF)$	$O(Ln_bF + LF^2)$

☺ Sub-linear precomputation

☺ Decoupled simple batching

[2] N Liao, D Mo, S Luo, X Li, P Yin. "Scalable Decoupling Graph Neural Network with Feature-Oriented Optimization". **VLDBJ 2023**.

SCARA: Experimental Results

- Time: 30-800× faster parallel precomputation
- Memory: Paper100M with 72GB without OOM
- Effectiveness: similar or better F1-score, fast convergence

Dataset	Nodes n	Edges m	Features F	Dataset	Nodes n	Edges m	Features F
Reddit [15]	232, 965	114, 615, 892	602	Papers100M [16]	111, 059, 956	1, 615, 685, 872	128

Transductive	Reddit				Papers100M			
	Learn (Pre. + Train)	Infer	Mem.	F1	Learn (Pre. + Train)	Infer	Mem.	F1
GraphSAINT	14.4 (– 14.4)	166.2	13.7	41.6 ±4.8	– – –	–	OOM	–
GAS	1151 (– 1151)	**2.2**	14.0	38.2 ±0.3	– – –	–	OOM	–
PPRGo	79.4 (62.3 + 17.1)	29.1	9.4	41.5 ±2.3	– – –	–	OOM	–
GBP	138 (124 + 13.7)	13.5	7.9	38.8 ±0.3	– – –	–	OOM	–
SCARA (ours)	**13.9** (0.07 + 13.8)	10.7	**5.6**	**44.1 ±0.4**	**1346** (12.7 + 1333)	**4.7**	**71.4**	**35.7 ±0.9**

[2] N Liao, D Mo, S Luo, X Li, P Yin. "Scalable Decoupling Graph Neural Network with Feature-Oriented Optimization". VLDBJ 2023.

Outline

Introduction

Data Structures

Evaluations

Design Scalable Algorithms

SCARA: Feature-dimension Utilization

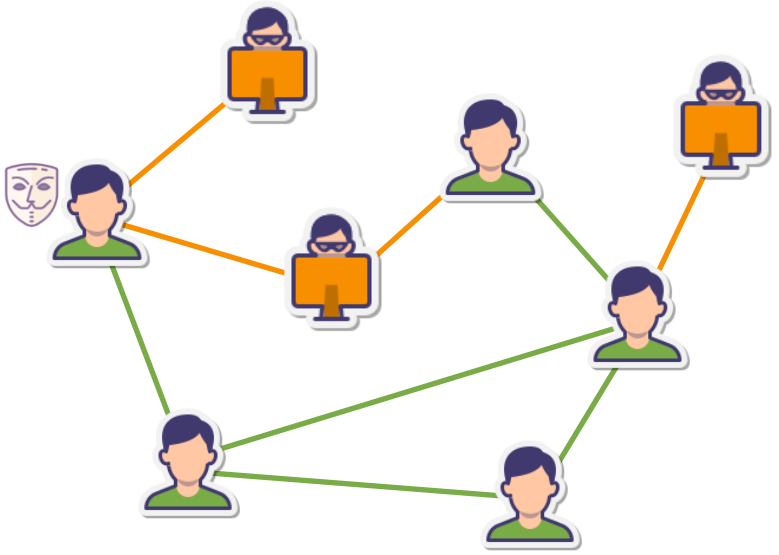
LD^2: Long-distance Embedding

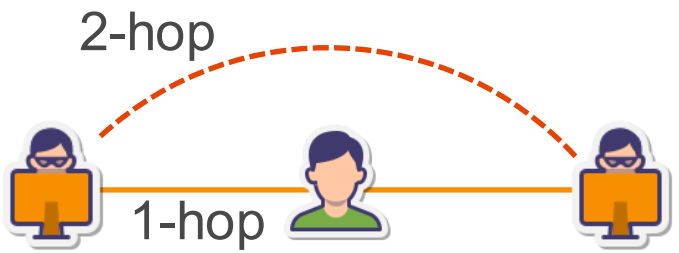
[NeurIPS 2023]

Conclusion and Future Directions

Background: Graph Heterophily

- Heterophily: connected nodes tend to be of dissimilar classes
- Graph: financial transaction network with benign or fraud users
- Task: identify fraud users
- Fraudsters use benign accounts as spring-boards to reach other fraudsters





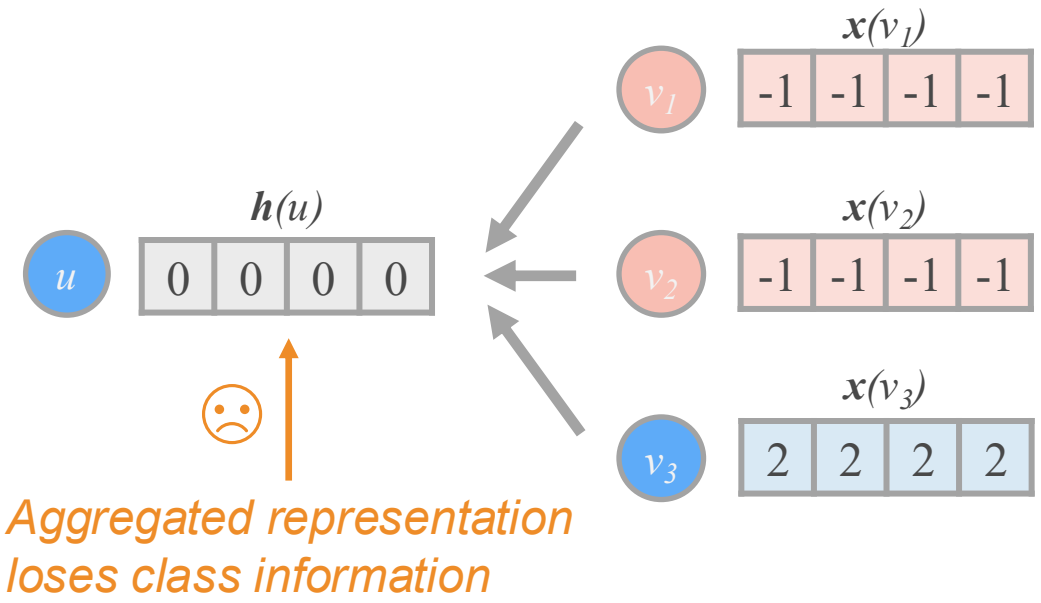
☹ *1-hop connection is not enough!*

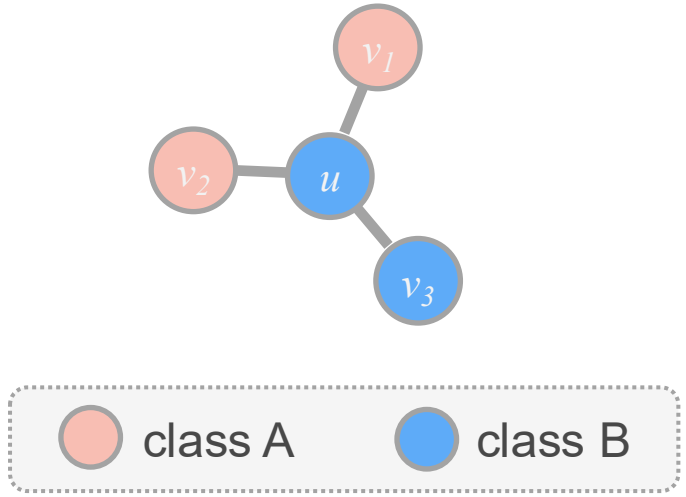
D Lim et al. “Large Scale Learning on Non-Homophilous Graphs: New Benchmarks and Strong Simple Methods”. NeurIPS 2021.

Background: Graph Heterophily

- Heterophily: 1-hop graph convolution fails

How to design decoupled representation under heterophily?





F Wu et al. "Simplifying graph convolutional networks". ICML 2019.

LD²: Representing Heterophily

- SGC:

$$\boldsymbol{P} = \tilde{\boldsymbol{A}}^{L} \cdot \boldsymbol{X}$$

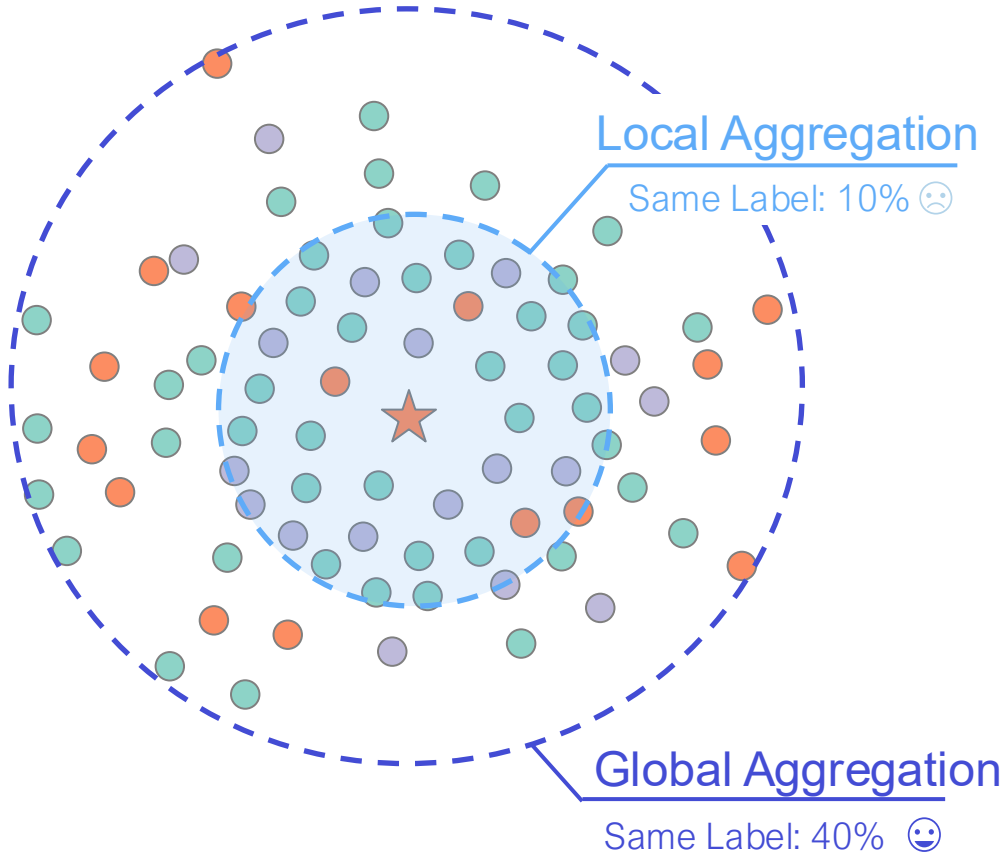
Local Aggregation

- LD² (ours):

2-hop Adjacency: $$\boldsymbol{P}_{A} = \phi\left(\boldsymbol{A}^{2L} \cdot \boldsymbol{\mathcal{N}}\right)$$

Constant Feature: $$\boldsymbol{P}_{X,L2} = \sum_{l=1}^{L} \bar{\boldsymbol{A}}^{2l} \cdot \boldsymbol{X}$$

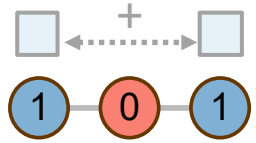
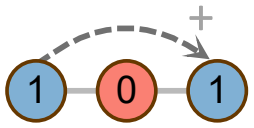
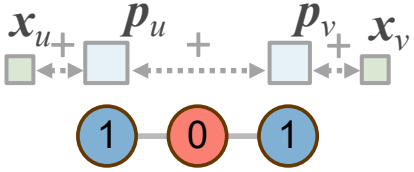
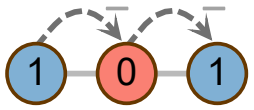
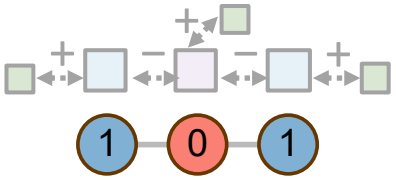
Inverse Feature: $$\boldsymbol{P}_{X,H} = \sum_{l=1}^{L} \tilde{\boldsymbol{L}}^{l} \cdot \boldsymbol{X}$$



[3] N Liao, S Luo, X Li, J Shi. "LD²: Scalable Heterophilous Graph Neural Network with Decoupled Embedding". NeurIPS 2023.

LD^2: Representing Heterophily

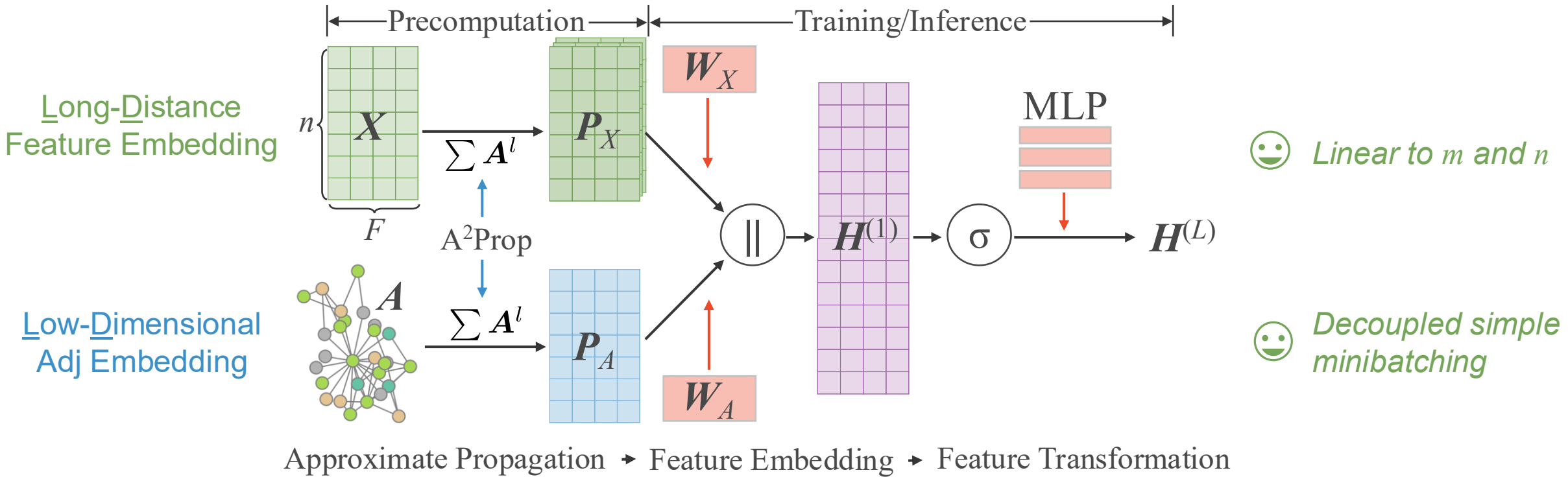
- LD^2 (ours):

	Spatial	Spectral
2-hop Adjacency: $\boldsymbol{P}_A = \phi\left(\boldsymbol{A}^{2L} \cdot \boldsymbol{\mathcal{N}}\right)$	$\boldsymbol{P}_A \cdot \boldsymbol{P}_A^{\top} \approx \boldsymbol{A}^2$ Low-rank approximation	 Structure smoothing
Constant Feature: $\boldsymbol{P}_{X,L2} = \sum_{l=1}^{L} \bar{\boldsymbol{A}}^{2l} \cdot \boldsymbol{X}$	 Long-range similarity	 Feature smoothing
Inverse Feature: $\boldsymbol{P}_{X,H} = \sum_{l=1}^{L} \tilde{\boldsymbol{L}}^{l} \cdot \boldsymbol{X}$	 Fine-grained details	 Structure difference

[3] N Liao, S Luo, X Li, J Shi. "LD^2: Scalable Heterophilous Graph Neural Network with Decoupled Embedding". **NeurIPS 2023**.

LD²: Decoupled Architecture

- Spectral precomputation + simple feature transformation



[3] N Liao, S Luo, X Li, J Shi. "LD²: Scalable Heterophilous Graph Neural Network with Decoupled Embedding". NeurIPS 2023.

LD^2: Complexity Results

- Minibatching without graph-scale overhead

Model	Precomp. Time	Training Time	Inference Time
GPRGNN [39]	$O(m)$	$O(ILPmF + ILnF^2)$	$O(LPmF + LnF^2)$
GCNJK [40]	–	$O(ILmF + ILnF^2)$	$O(LmF + LnF^2)$
MixHop [41]	–	$O(ILPLmF + ILnF^2)$	$O(LPLmF + LnF^2)$
LINKX [42]	–	$O(ImF + ILnF^2)$	$O(mF + LnF^2)$
LD^2 (ours)	$O(LPmF)$	$O(ILnF^2)$	$O(LnF^2)$

☹ *Not scalable to large graph m and n*

😀 *Linear to m and n*

Model	Precomp. Mem.	Training Mem.	Inference Mem.
GPRGNN [39]	$O(m)$	$O(LnF + LF^2 + m)$	$O(LnF + LF^2 + m)$
GCNJK [40]	–	$O(LCnF + LCF^2)$	$O(LCnF + LCF^2)$
MixHop [41]	–	$O(CLnF + CLF^2)$	$O(CLnF + CLF^2)$
LINKX [42]	–	$O(LCnbF + LCF^2 + nF)$	$O(LCnbF + LCF^2 + nF)$
LD^2 (ours)	$O(CnF)$	$O(LCnbF + LCF^2)$	$O(LCnbF + LCF^2)$

☹ *Not suitable for minibatch*

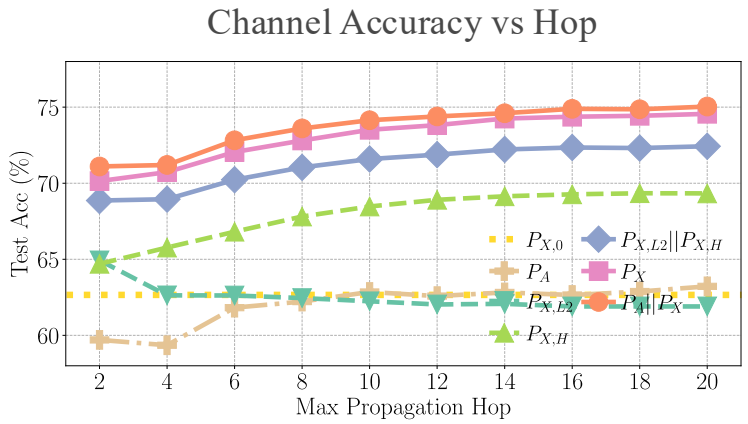
😀 *Decoupled simple batching*

[3] N Liao, S Luo, X Li, J Shi. "LD^2: Scalable Heterophilous Graph Neural Network with Decoupled Embedding". **NeurIPS 2023**.

LD^2: Experimental Results

- Efficiency: 3-15× faster, 5× lower memory
- Effectiveness: No accuracy drop for minibatch

Dataset	twitch-gamers			pokec			snap-patents			wiki		
	Learn	Infer	Mem.	Learn	Infer	Mem.	Learn	Infer	Mem.	Learn	Infer	Mem.
MLP	6.36	0.02	0.61	47.86	0.11	13.77	27.39	0.28	9.33	133.55	0.62	18.15
PPRGo	10.46+15.88	0.41	9.64	121.95+56.11	2.69	3.82	(>12h)			(>12h)		
SGC	0.09+0.74	0.01	0.28	1.05+8.08	0.01	0.28	4.94+23.54	0.01	0.42	12.66+7.98	0.01	0.52
GCNJK-GS	71.48	0.02*	7.33	27.33	0.09*	9.03	19.02	0.23*	9.21	95.52	0.69*	16.36
MixHop-GS	52.12	0.01*	1.49	71.35	0.03*	12.91	45.24	0.16*	19.58	84.22	0.23*	16.28
LINKX	10.99	0.19	2.35	28.77	0.33	9.03	39.80	0.22	21.53	180.71	1.14	14.53
LD^2 (ours)	0.85+**1.96**	**0.01**	**1.44**	17.95+**6.18**	**0.01**	**3.82**	31.32+**6.96**	**0.02**	**3.96**	28.12+**6.50**	**0.01**	**4.47**



[3] N Liao, S Luo, X Li, J Shi. "LD^2: Scalable Heterophilous Graph Neural Network with Decoupled Embedding". **NeurIPS 2023**.

Takeaway

SCARA: Feature-dimension Utilization

- Design: Decouple graph convolution as PPR
- Algorithm: Reuse features during graph computation
- Experiment: 20 mins for billion-edge graphs

LD^2: Long-distance Embedding

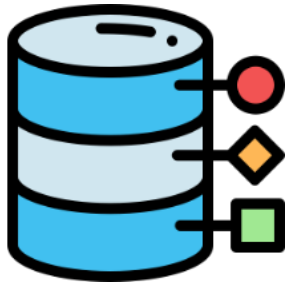
- Algorithm: Adapt decoupled propagation under heterophily
- Theory: Long-range spectral aggregation
- Experiment: Minibatching on large-scale heterophilous graphs

Outline

Introduction

Algorithms

Evaluations



Deploy Scalable Data Structures

HubGT: Hierarchical Hub Labeling

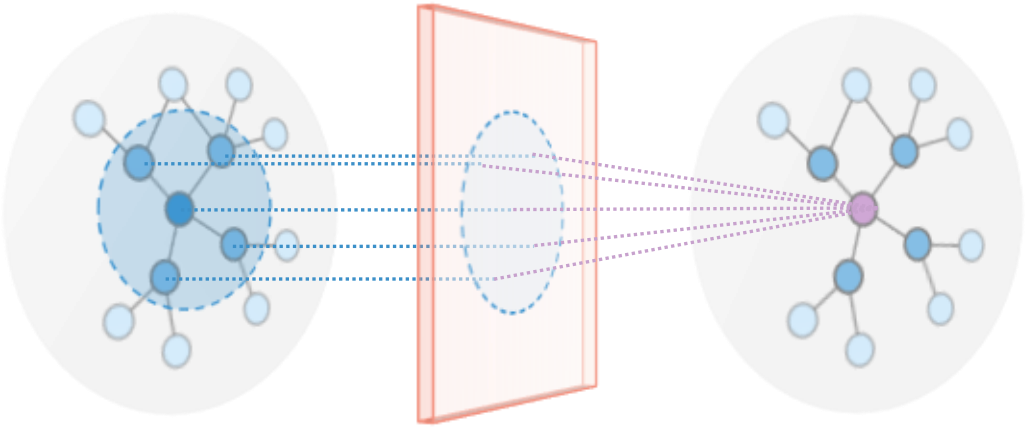
GENTI: Subgraph 2023 Streaming

Conclusion and Future Directions

Background: Subgraph GNN

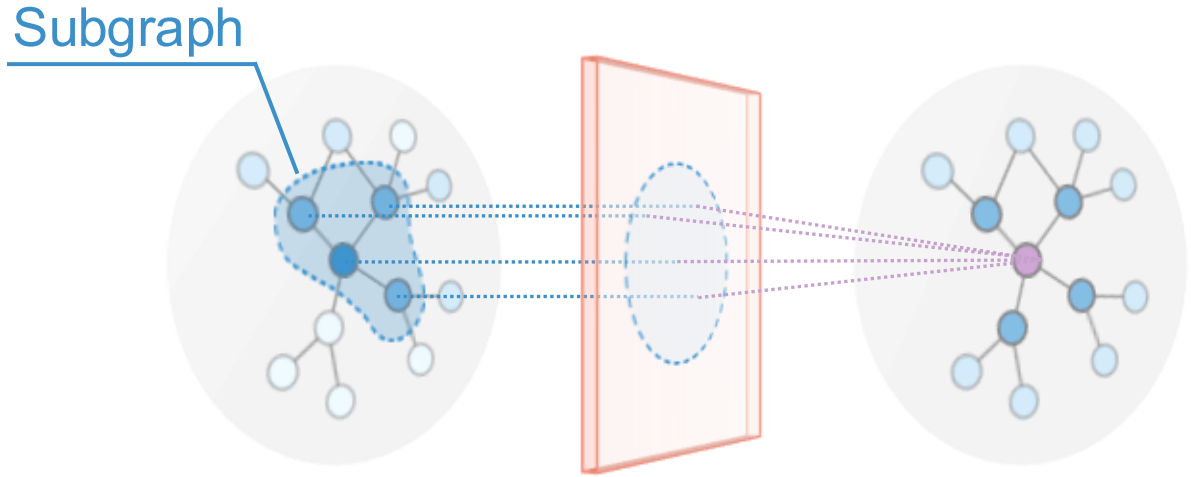
- Full-graph learning:

$$\boldsymbol{h}(u) = \sum_{v \in N(u)} \boldsymbol{x}(v) \cdot \boldsymbol{W}$$



- Subgraph learning:

$$\boldsymbol{h}(u) = \sum_{v \in S(u)} \boldsymbol{x}(v) \cdot \boldsymbol{W}$$

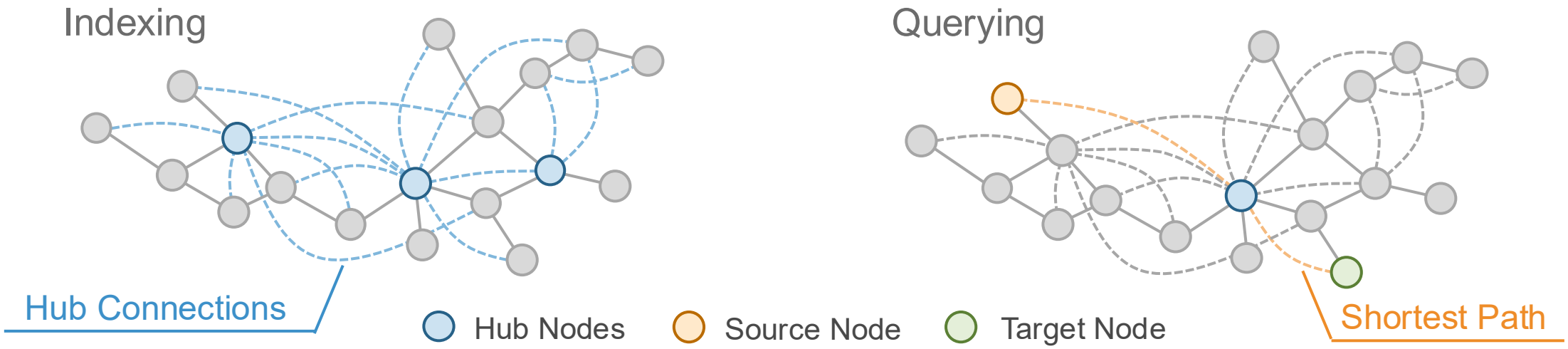


☺ *Smaller graph data*

W Hamilton et al. “Inductive Representation Learning in Large Attributed Graphs”. NeurIPS 2017.

HubGT: Hub Labeling

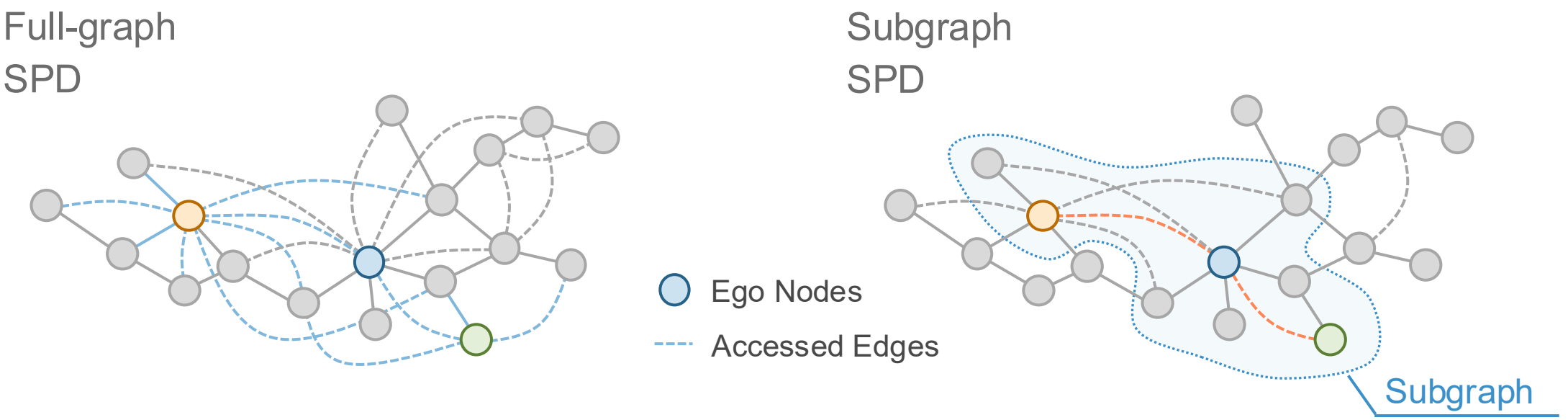
- Hub Labeling: adding links to selected hub nodes → distant node pairs can be easily reached through hubs
- Shortest Path Distance (SPD) can be computed through hubs



[4] N Liao*, Z Yu*, S Luo. "HubGT: Fast Graph Transformer with Decoupled Hierarchy Labeling". NeurIPS 2025.

HubGT: Hub Labeling

- Subgraph SPD (ours): querying any node pairs within a subgraph

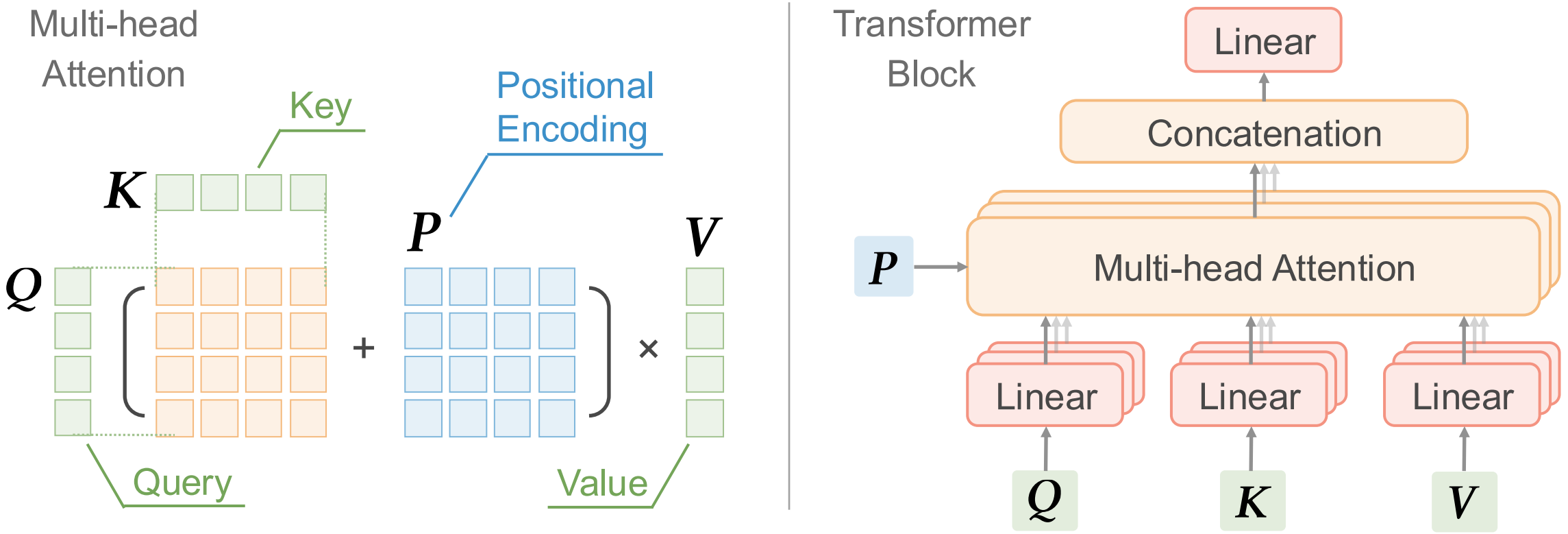


Query Time	$O(s)$	$O(1)$
Index Size	$O(ns^2)$	$O(ns^2)$

[4] N Liao*, Z Yu*, S Luo. "HubGT: Fast Graph Transformer with Decoupled Hierarchy Labeling". **NeurIPS 2025**.

Background: Transformer PE

- Positional Encoding (PE): relation as bias in attention



A Vaswani et al. “Attention Is All You Need”. NeurIPS 2017.

HubGT: Graph Transformer PE

- Positional Encoding: emphasize important pair-wise relation

Positional Encoding

$$\tilde{\boldsymbol{H}}_i = \text{softmax}\left(\frac{\boldsymbol{Q}_i \boldsymbol{K}_i^\top}{\sqrt{F_K}} + \boldsymbol{P}\right)\boldsymbol{V}_i,$$

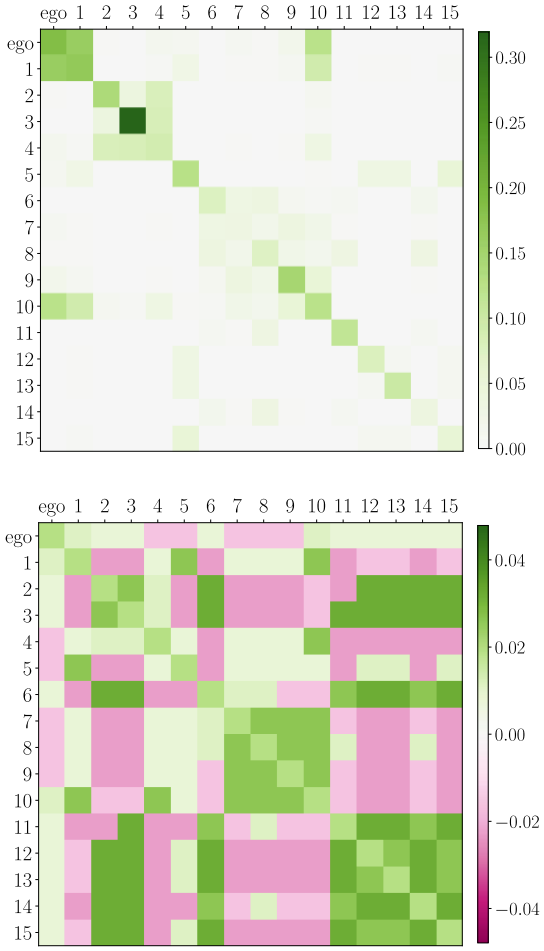
Node Feature

Transformer Attention

- Complexity:

$O(n^2)$ Full graph

$O(s^2)$ Subgraph



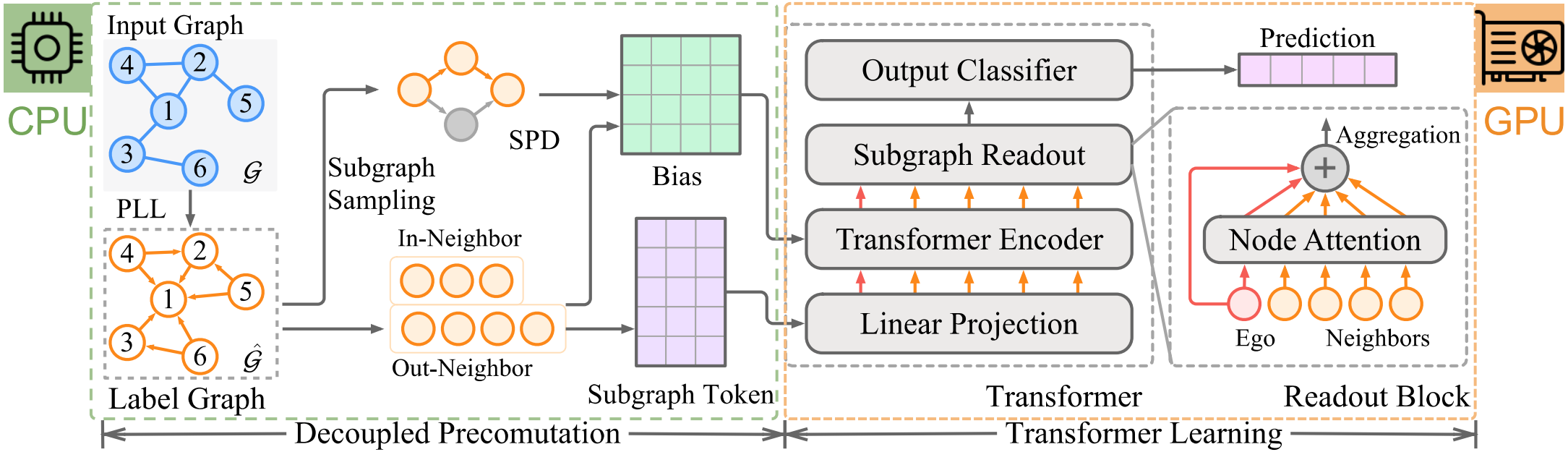
Adjacency PE

SPD Encoding

[4] N Liao*, Z Yu*, S Luo. "HubGT: Fast Graph Transformer with Decoupled Hierarchy Labeling". **NeurIPS 2025**.

HubGT: Graph Transformer PE

- Hub node → hierarchical subgraph → input token
- SPD value → positional encoding → attention bias



[4] N Liao*, Z Yu*, S Luo. "HubGT: Fast Graph Transformer with Decoupled Hierarchy Labeling". NeurIPS 2025.

HubGT: Complexity Results

- Only linear time & memory complexity

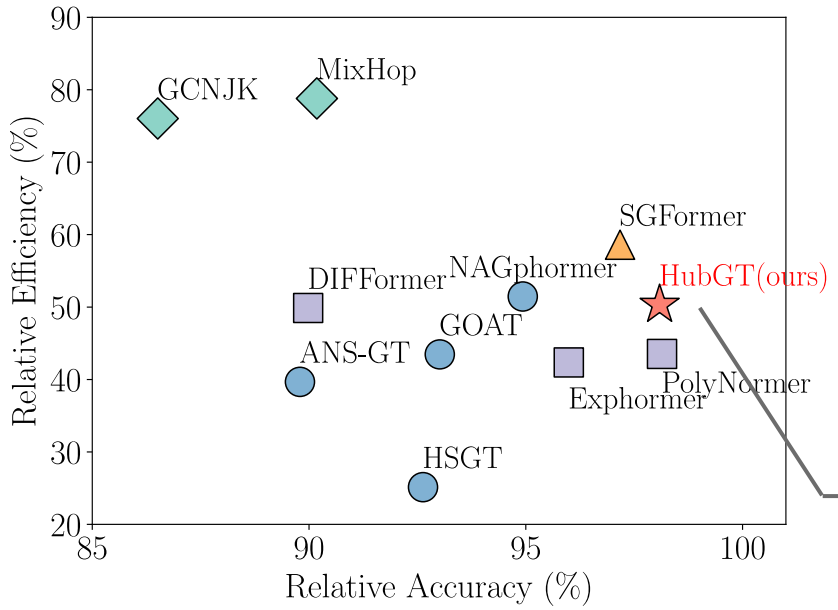
☹ *Not scalable to large m and n* ☹ *Insufficient memory*

Taxonomy	Model	Precompute Time	Train Time	RAM Mem.	GPU Mem.	Scale
Vanilla (FB)	Graphormer [2]	$O(n^3)$	$O(Ln^2F)$	$O(n^2)$	$O(Ln^2F)$	0.3K/0.6K
	GRPE [18]	$O(n^2)$	$O(Ln^2F)$	$O(n^2)$	$O(Ln^2F)$	0.3K/0.6K
Kernel-based (NS)	GraphGPS [7]	$O(n^3)$	$O(LnF^2 + LmF)$	$O(nF + n^2)$	$O(Ln_bF + m)$	1.0K/3.0K
	Exphormer [19]	$O(dn)$	$O(LnF^2 + LmF)$	$O(nF + m)$	$O(Ln_bF + m)$	0.2M/1.2M
	NodeFormer [6]	–	$O(LnF^2 + LmF)$	$O(nF + m)$	$O(Ln_bF + m)$	2.4M/60M
	DIFFormer [8]	–	$O(LnF^2 + LmF)$	$O(nF + m)$	$O(Ln_bF + m)$	1.6M/40M
	PolyNormer [12]	–	$O(LnF^2 + LmF)$	$O(nF + m)$	$O(Ln_bF + m)$	2.4M/0.1B
Hierar-chical (RS)	NAGphormer [9]	$O(LmF_0)$	$O(LnF^2)$	$O(LnF)$	$O(Ln_bF^2)$	2.4M/60M
	PolyFormer [11]	$O(LmF_0)$	$O(LnF^2)$	$O(LnF)$	$O(Ln_bF^2)$	0.2M/30M
	ANS-GT [13]	$O(ns^2 + md^L)$	$O(LnF^2 + Lns^2F + Lm)$	$O(nF + ns^2 + md^L)$	$O(Ln_bF + n_bs^2)$	20K/0.2M
	GOAT [10]	$O(nF)$	$O(LnF^2 + LmF)$	$O(nF + m)$	$O(Ln_b^2 + Ln_bF + m)$	2.9M/0.1B
	HSGT [14]	$O(n + md^L)$	$O(LnF^2 + LmF)$	$O(nF + md^L)$	$O(Ln_b^2 + Ln_bF + Lm)$	2.4M/0.1B
	HubGT (ours)	$O(ns^3 + ms)$	$O(LnF^2 + Lns^2F)$	$O(nF + ns^2)$	$O(Ln_bF + n_bs^2)$	1.6M/0.1B

[4] N Liao*, Z Yu*, S Luo. "HubGT: Fast Graph Transformer with Decoupled Hierarchy Labeling". **NeurIPS 2025**.

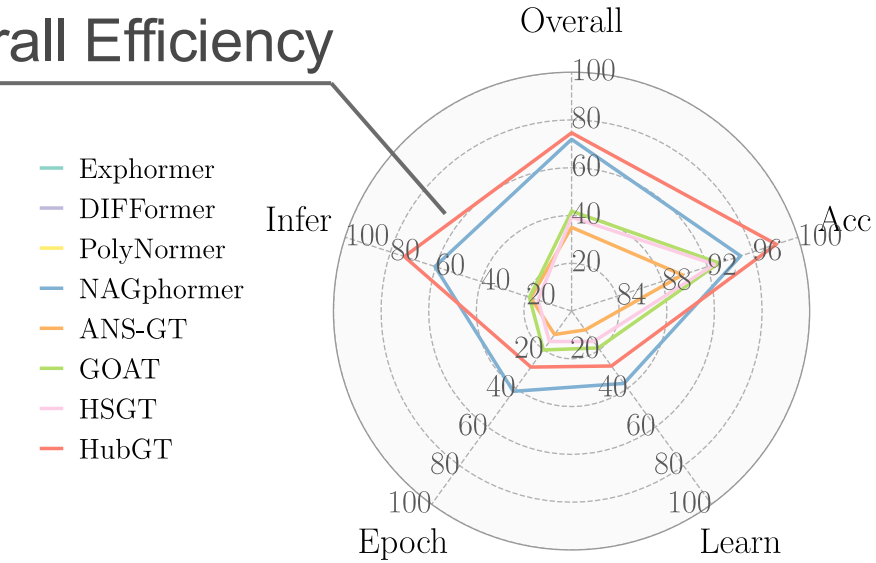
HubGT: Experimental Results

- Efficiency: fast & scalable minibatch
- Effectiveness: higher accuracy under heterophily



Fast + top-tier Accuracy

Overall Efficiency



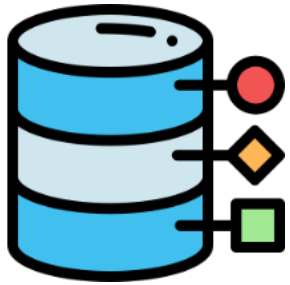
[4] N Liao*, Z Yu*, S Luo. "HubGT: Fast Graph Transformer with Decoupled Hierarchy Labeling". **NeurIPS 2025**.

Outline

Introduction

Algorithms

Evaluations



Deploy Scalable Data Structures

HubGT: Hierarchical Hub Labeling

GENTI: Subgraph Streaming

[VLDB 2024]

Conclusion and Future Directions

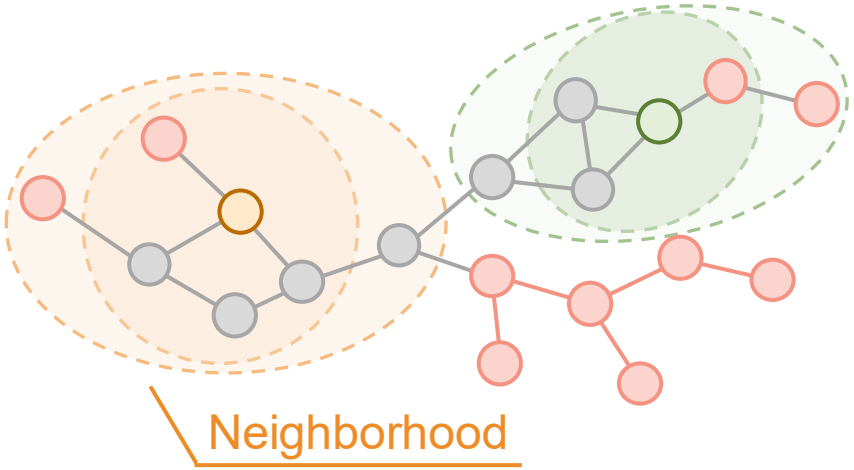
Background: Walk-based GNN

- Subgraph by convolution:

$$P[u,v] = H[\mathcal{N}(u)] + H[\mathcal{N}(v)]$$

Link Feature

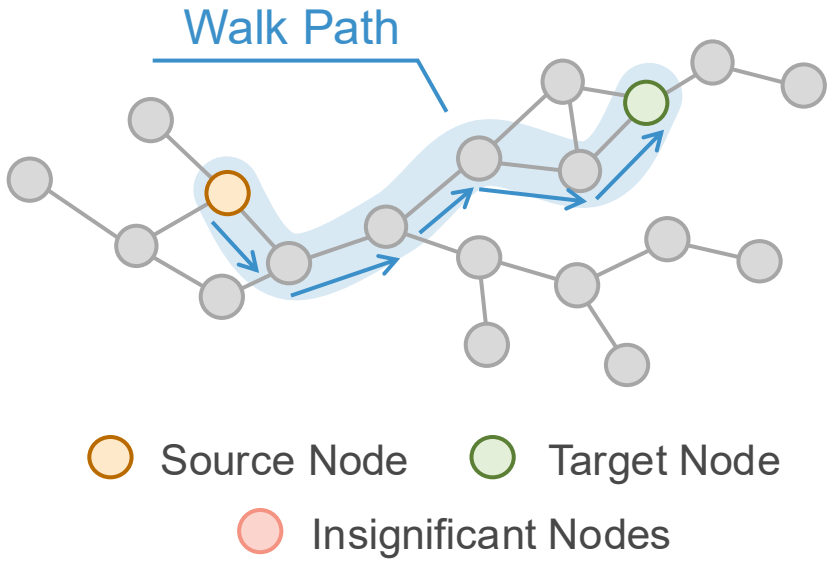
☹ *Include redundant nodes*



- Subgraph by random walks:

$$P[u,v] = H[\{u, t_1, t_2, \cdots, v\}]$$

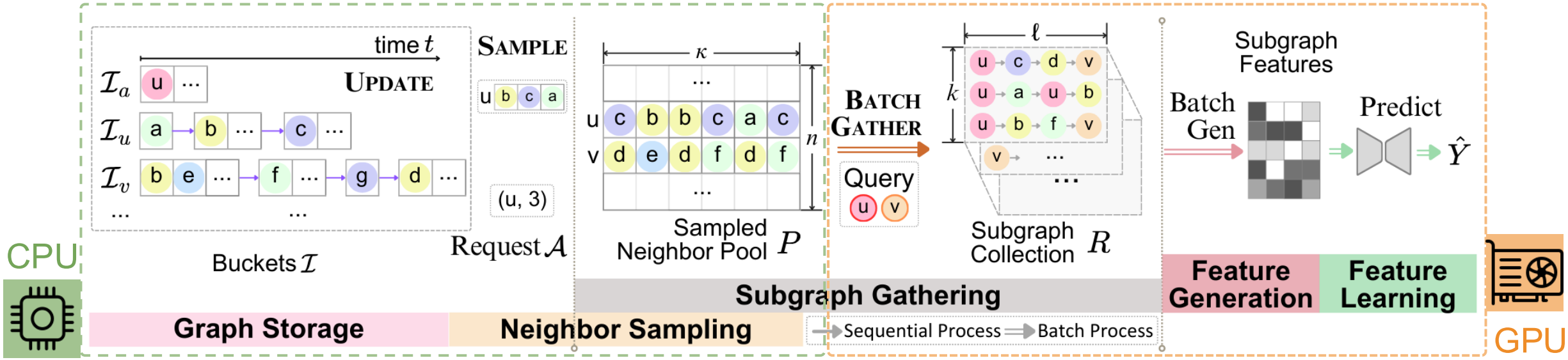
☺ *More relevant nodes*



M Zhang & Y Chen. “Link prediction based on graph neural networks”. NeurIPS 2018.

GENTI: Subgraph GNN Operators

- Sampling: linked buckets arranged in temporal order
- Gathering: pipelining sample (CPU) and gather (GPU)

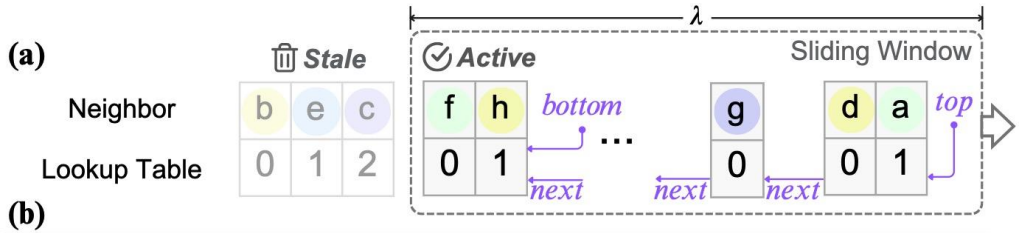


[5] Z Yu*, N Liao*, S Luo. "GENTI: GPU-powered Walk-based Subgraph Extraction for Scalable Representation Learning on Dynamic Graphs". VLDB 2024.

GENTI: Streaming in Subgraph GNN

Time-based Storage

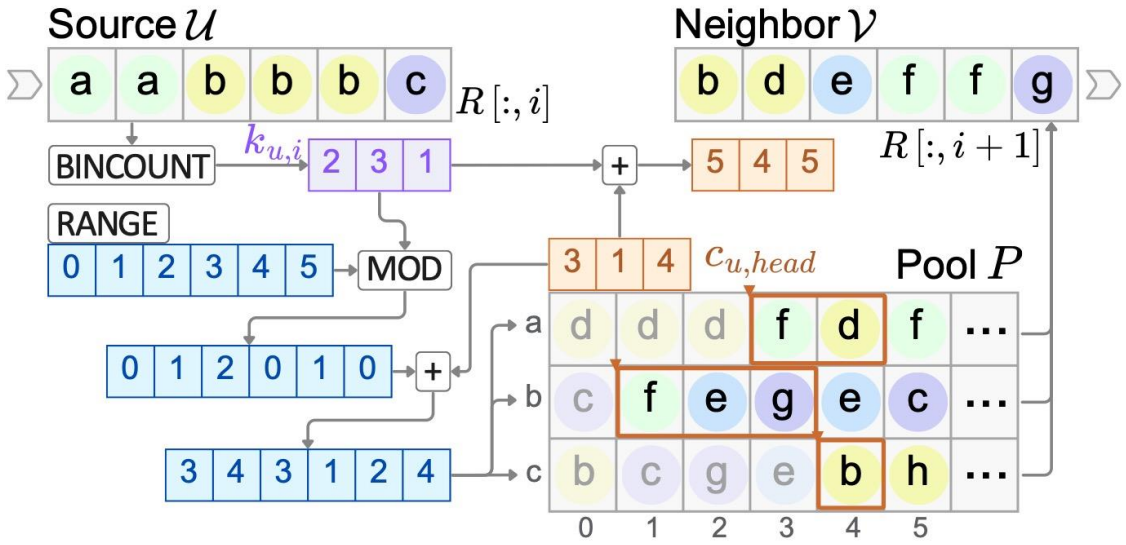
- Fast sampling across buckets
- Constrain time by sliding window



Bucket	$\mathcal{I}_{u,0}$	$\mathcal{I}_{u,1}$	⋯	$\mathcal{I}_{u,r-1}$	$\mathcal{I}_{u,r}$
Timestamp	$[0, \alpha)$	$[\alpha, 2\alpha)$	⋯	$[(r-1)\alpha, r\alpha)$	$[r\alpha, (r+1)\alpha)$
Sample Weight	$[0, e^1)$	$[e^1, e^2)$	⋯	$[e^{r-1}, e^r)$	$[e^r, e^{r+1})$
Random Number	$\varnothing$	$\{z_4\}$	⋯	$\{z_3\}$	$\{z_1, z_2\}$
Fetch Amount	0	1	⋯	1	2
Sampled Neighbor	$\varnothing$	f	⋯	g	a d

Batch Join GPU Operator

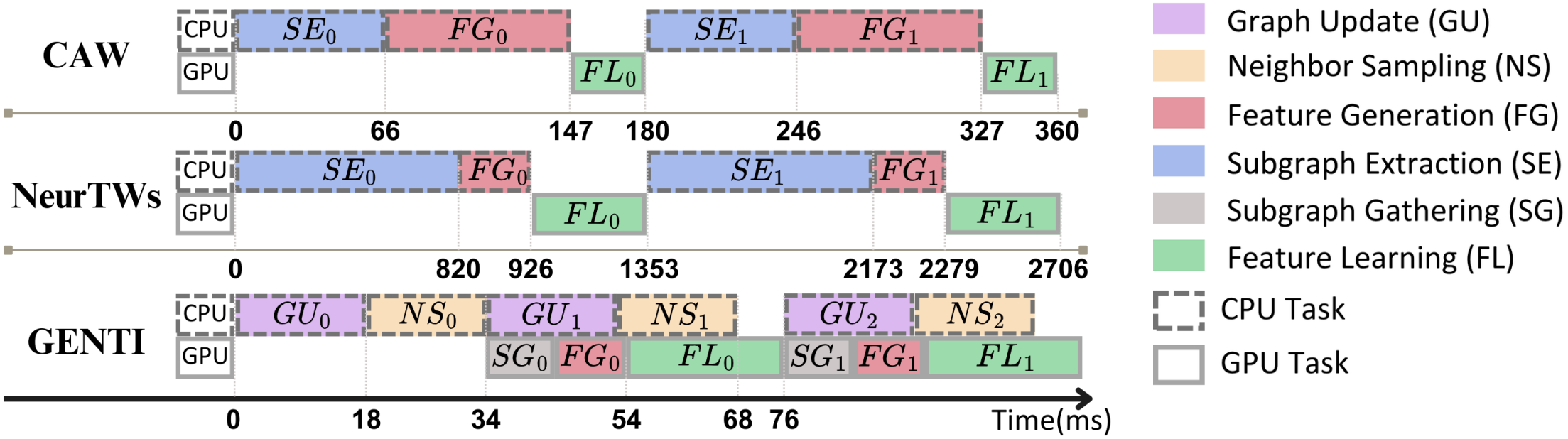
- Dense operations on GPU
- Complexity: reduce from L to $\sqrt{L}$



[5] Z Yu*, N Liao*, S Luo. "GENTI: GPU-powered Walk-based Subgraph Extraction for Scalable Representation Learning on Dynamic Graphs". VLDB 2024.

GENTI: Experimental Results

- Pipeline: high device utilization, balanced workload
- Efficiency: 3-26× speedup, fast extraction & gathering



[5] Z Yu*, N Liao*, S Luo. "GENTI: GPU-powered Walk-based Subgraph Extraction for Scalable Representation Learning on Dynamic Graphs". VLDB 2024.

Takeaway

HubGT: Hub Labeling for Graph Transformer

- Design: Decouple positional encoding
- Algorithm: Hierarchical hub labeling
- Experiment: Minibatch & improved accuracy

GENTI: Streaming for Subgraph GNN

- Design: Decouple subgraph extraction
- Algorithm: Batch operators & data structure on GPU
- Experiment: Tight pipeline, faster subgraph extraction

Outline

Introduction

Algorithms

Data Structures

Evaluate Scalability Performance

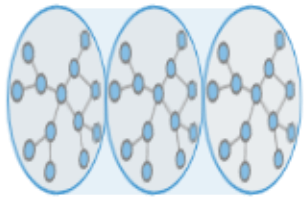
UNIFEWS: Spectral Sparsification

pyg_spectral GNN Benchmark

Conclusion and Future Directions

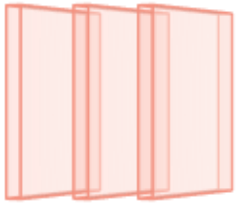
Background: Spectral GNN

- Spectral domain: related to graph eigen-decomposition



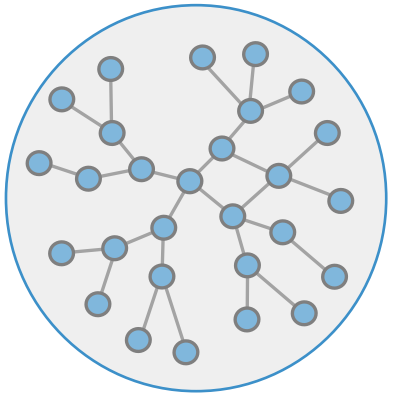
Graph Computation

One-time Precomputation

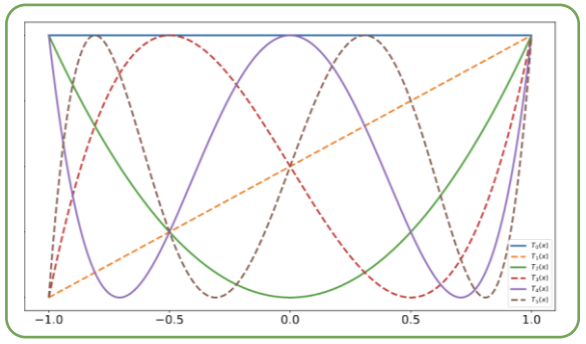


Feature Transformation

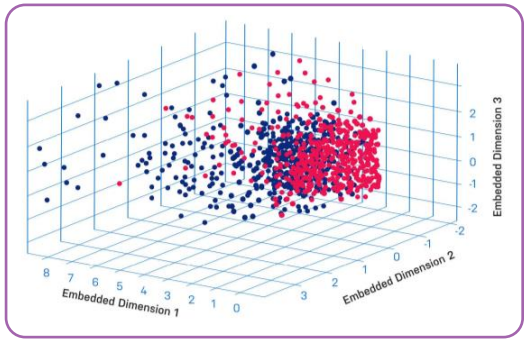
Iterative training



(Spatial) Graph



Spectral Features



Learned Features

M He et al. “Convolutional Neural Networks on Graphs with Chebyshev Approximation, Revisited”. NeurIPS 2022.

Background: Spectral GNN

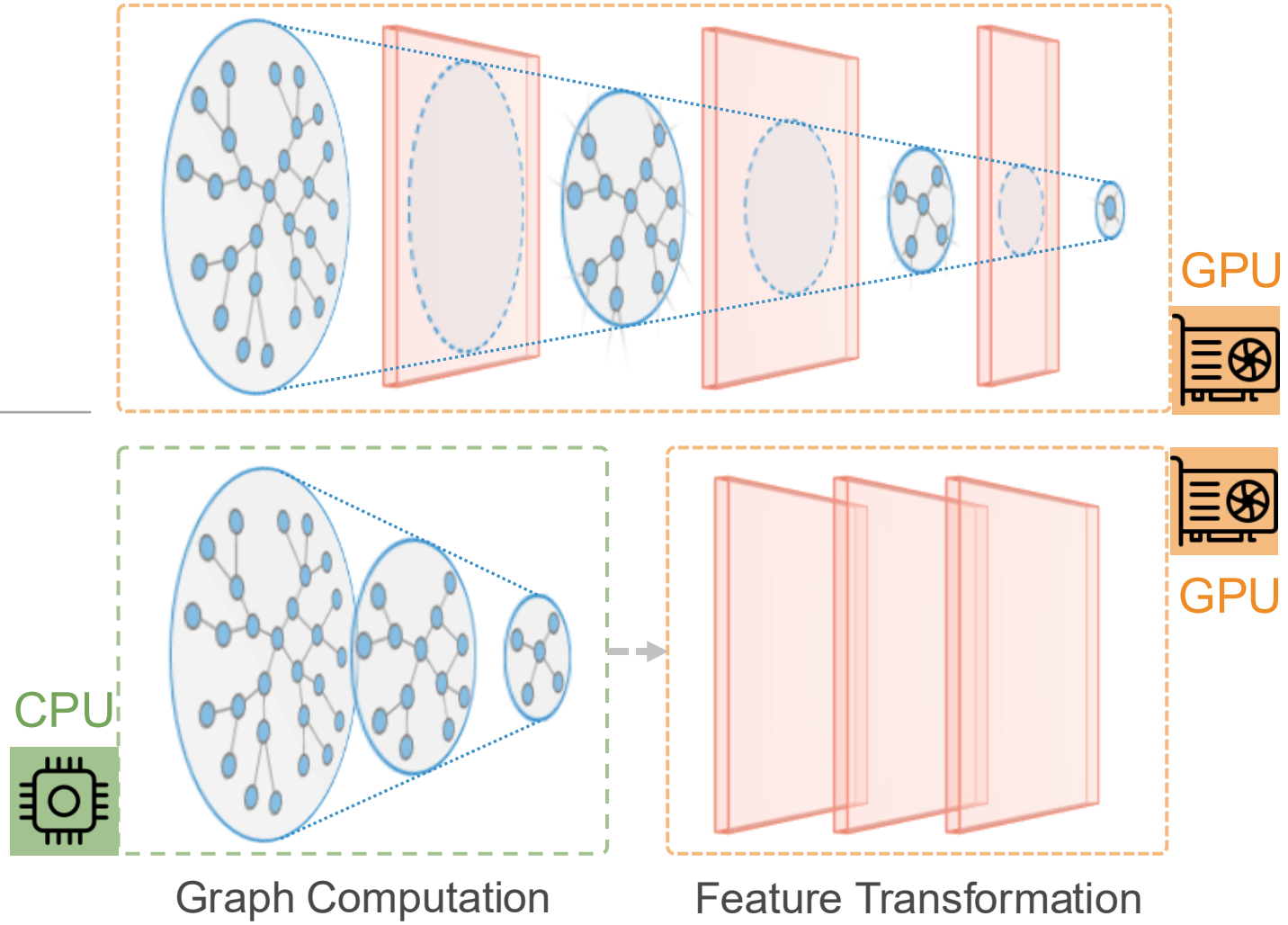
- Spatial GNN:

  Coupled & iterative graph computation

- Spectral GNN:

  Explicit & decoupled spectral features

☺ *Save memory & time*



UNIFEWS: Spectral Sparsification

- Sparsification: prune insignificant values to zero

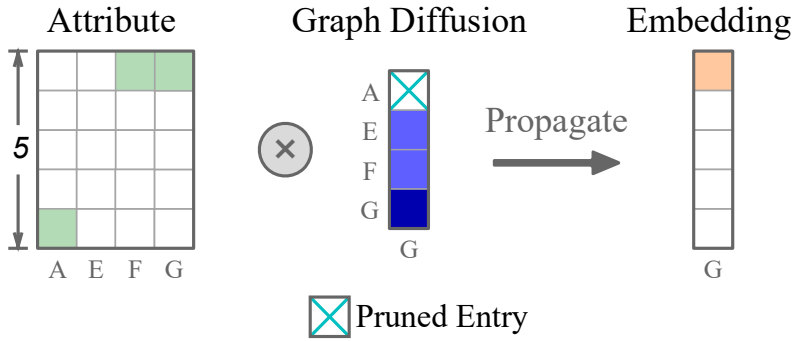
Graph Propagation

$$\boldsymbol{p}_{(l+1)}[u] = \sum_{v \in \mathcal{N}(u)} \boldsymbol{T}_{(l)}[u, v] \cdot \boldsymbol{p}_{(l)}[v]$$

Embedding

Graph Diffusion

Subject to prune



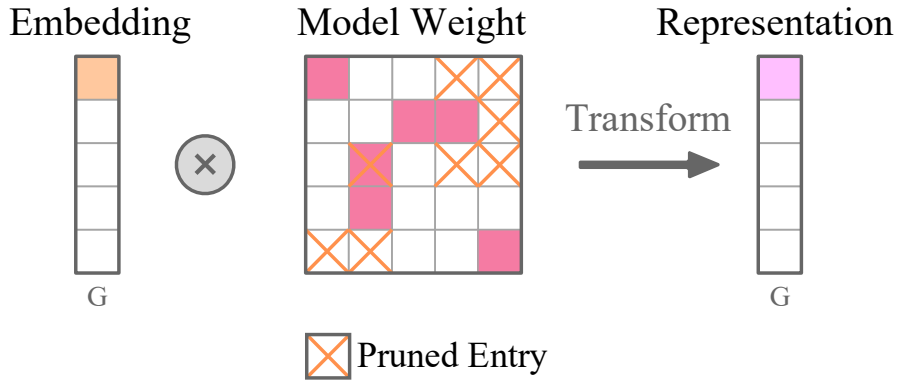
Feature Transformation

$$\boldsymbol{H}_{(l+1)}[:, i] = \sum_{j=1}^{f} \boldsymbol{W}_{(l)}[j, i] \cdot \boldsymbol{P}_{(l)}[:, j]$$

Representation

Weight

Subject to prune



[6] N Liao, Z Yu, R Zeng, S Luo. "UNIFEWS: You Need Fewer Operations for Efficient Graph Neural Network". ICML 2025.

UNIFEWS: Approximation Theory

- GNN as Graph Smoothing:

$$\boldsymbol{p}^* = \arg\min_{\boldsymbol{p}} \mathcal{L}, \quad \mathcal{L} = \|\boldsymbol{p} - \boldsymbol{x}\|^2 + c \cdot \boldsymbol{p}^\top \boldsymbol{L}\boldsymbol{p},$$

Embedding; Attribute; Graph

- Spectral Approximation:

$$\|\hat{\boldsymbol{p}}^* - \boldsymbol{p}^*\| \leq c\epsilon \|\boldsymbol{p}^*\|.$$

Approx Bound

- Error Guarantee:

(general)

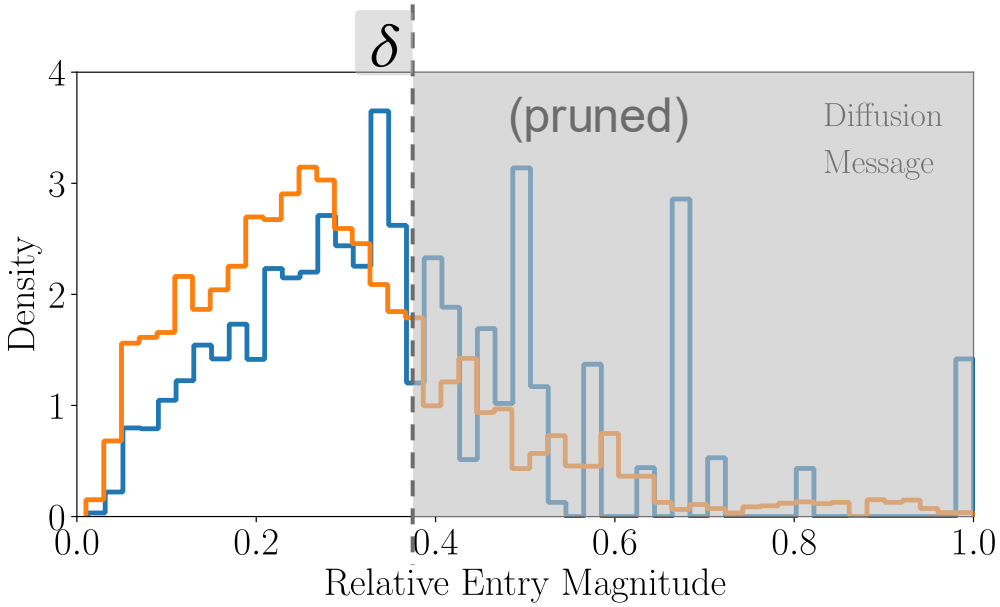
$$\Longrightarrow \quad q_a \delta_a \leq \epsilon \|\boldsymbol{p}\|$$

Sparsity; Prune Threshold

(scale-free graph)

$$\Longrightarrow \quad \epsilon = O\left(\eta_a (1-\eta_a)^{\frac{1}{1-\alpha}}\right)$$

Graph Distribution



[6] N Liao, Z Yu, R Zeng, S Luo. "UNIFEWS: You Need Fewer Operations for Efficient Graph Neural Network". ICML 2025.

UNIFEWS: Approximation Theory

Graph Sparsification:

$$\|\boldsymbol{p}_{(l+1)} - \hat{\boldsymbol{p}}_{(l+1)}\| \le O(\epsilon) \cdot \|\boldsymbol{p}_{(l)}\| + bc\epsilon(1 + O(\epsilon))\|\boldsymbol{p}_{(l)}\|$$

Embedding

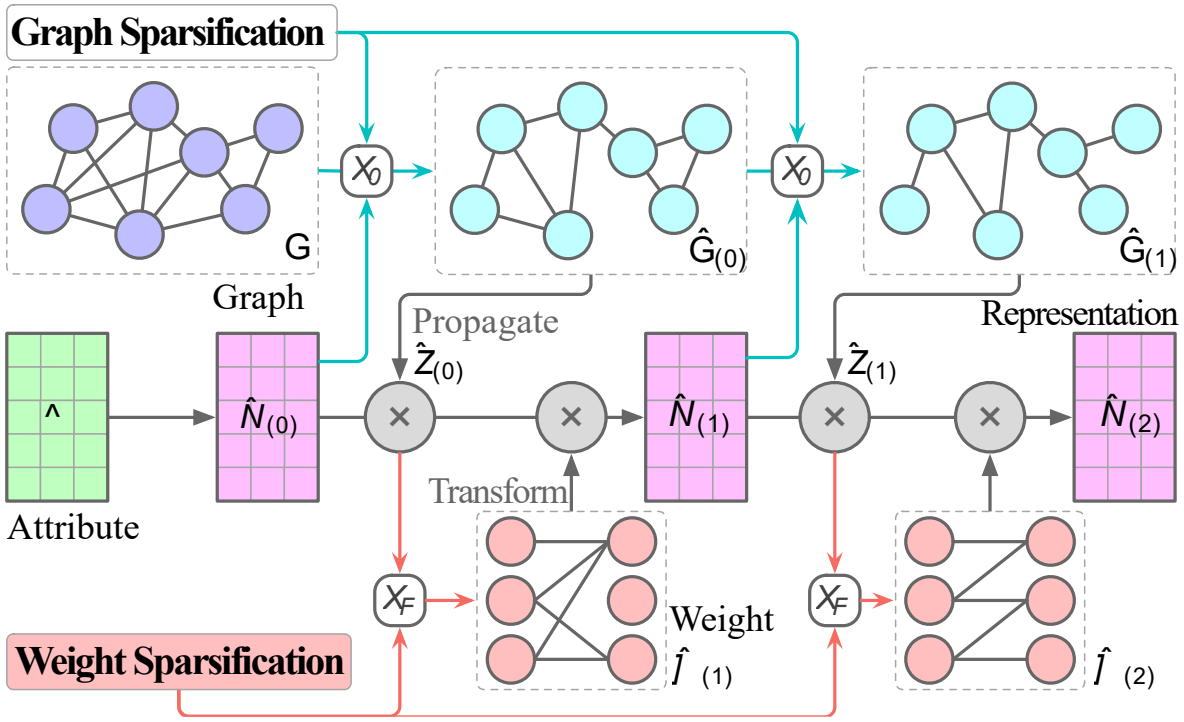
Approx Bound

+ Weight Sparsification:

$$\|\hat{\boldsymbol{H}}_{(l+1)} - \boldsymbol{H}_{(l+1)}\|_F \le \ell_\sigma bc\epsilon\|\boldsymbol{H}_{(l)}\|_F\|\boldsymbol{W}\|_F + \ell_\sigma q_w \delta_w$$

Representation

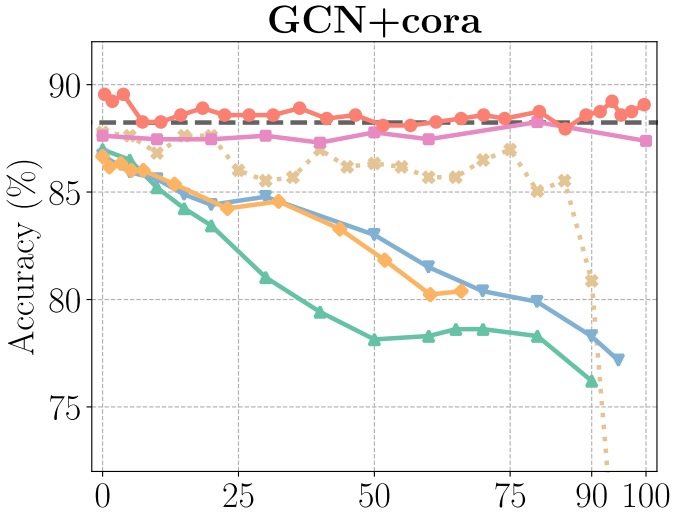
Network Distribution



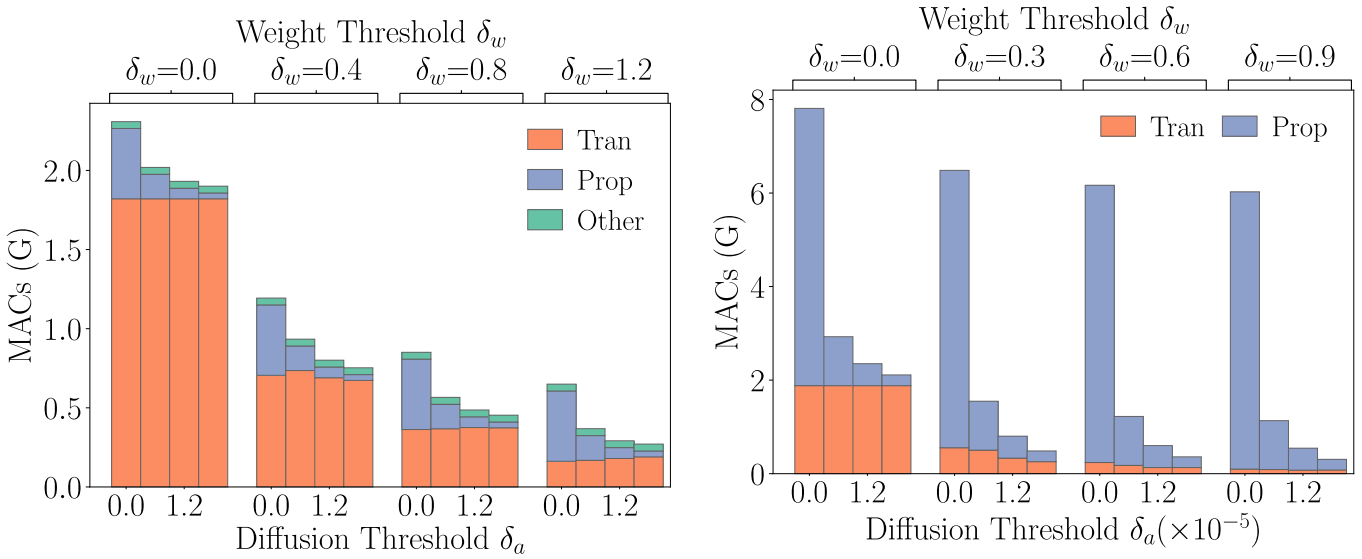
[6] N Liao, Z Yu, R Zeng, S Luo. "UNIFEWS: You Need Fewer Operations for Efficient Graph Neural Network". ICML 2025.

UNIFEWS: Experimental Results

- Accuracy: Accurate at high pruning ratio
- Efficiency: Boost graph & weight computation



Accuracy

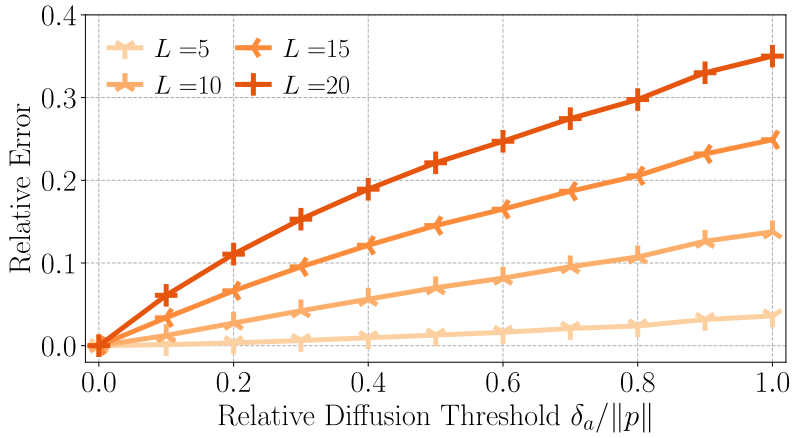


Efficiency

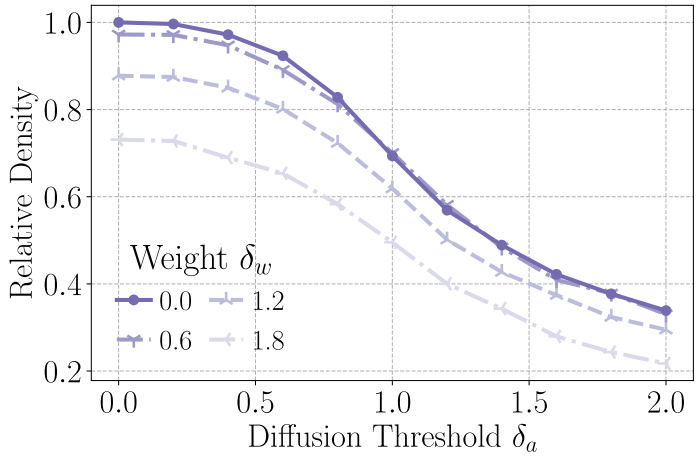
[6] N Liao, Z Yu, R Zeng, S Luo. "UNIFEWS: You Need Fewer Operations for Efficient Graph Neural Network". **ICML 2025**.

UNIFEWS: Experimental Results

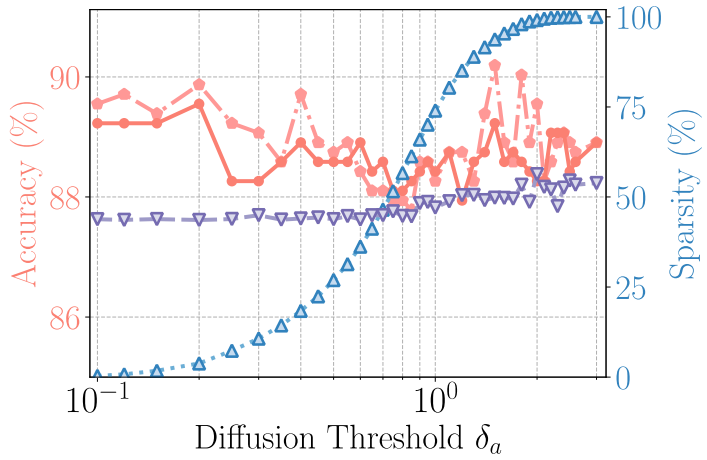
- Approximation: Follow error guarantee theory
- Implementation: Applicable to both iterative & decoupled models



Relative Error



Message Sparsity



Element Sparsity

[6] N Liao, Z Yu, R Zeng, S Luo. "UNIFEWS: You Need Fewer Operations for Efficient Graph Neural Network". **ICML 2025**.

Outline

Introduction

Algorithms

Data Structures

Evaluate Scalability Performance

UNIFEWS: Spectral Sparsification

pyg_spectral: GNN Benchmark

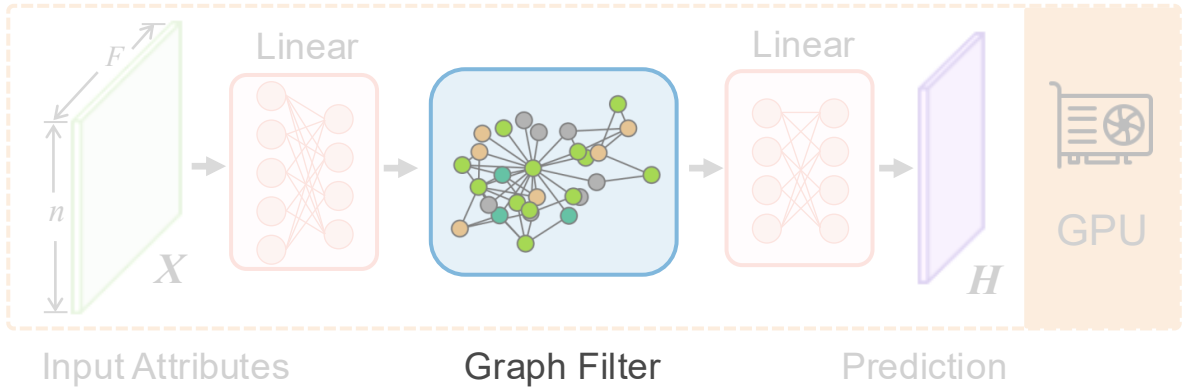
[SIGMOD 2026]

Conclusion and Future Directions

pyg_spectral: Spectral Graph Filter

- Iterative:

$$\boldsymbol{H}^{(L)} = \sigma(\tilde{\boldsymbol{A}}\sigma(\tilde{\boldsymbol{A}}\cdots\boldsymbol{X}\boldsymbol{W}^{(1)}\cdots)\boldsymbol{W}^{(L-1)})$$

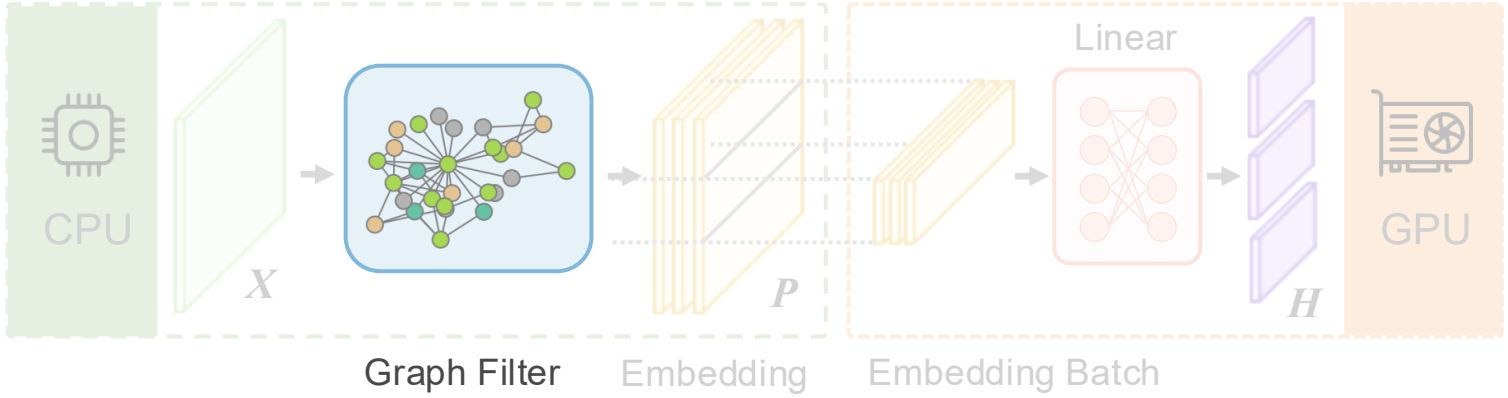


😀 *Graph filter is the same!*

- Decoupled:

$$\boldsymbol{H}^{(0)} = \tilde{\boldsymbol{A}}\cdots\tilde{\boldsymbol{A}}\boldsymbol{X}$$

$$\boldsymbol{H}^{(l+1)} = \sigma(\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)})$$



[7] N Liao, H Liu, Z Zhu, S Luo, L Lakshmanan. "A Comprehensive Benchmark on Spectral GNNs: The Impact on Efficiency, Memory, and Effectiveness". SIGMOD 2026.

pyg_spectral: Graph Filter Taxonomy

SCARA

$$\boldsymbol{H}^{(0)} = \sum_{0}^{\infty} \alpha(1-\alpha)^{l} \tilde{\boldsymbol{A}}^{l} \cdot \boldsymbol{X}$$

LD²

$$\boldsymbol{P}_{A} = \phi\left(\boldsymbol{A}^{2L} \cdot \boldsymbol{\mathcal{N}}\right)$$

$$\boldsymbol{P}_{X,L2} = \sum_{l=1}^{L} \bar{\boldsymbol{A}}^{2l} \cdot \boldsymbol{X}$$

$$\boldsymbol{P}_{X,H} = \sum_{l=1}^{L} \tilde{\boldsymbol{L}}^{l} \cdot \boldsymbol{X}$$

UNIFEWS

$$\boldsymbol{p}_{(l+1)}[u] = \sum_{v \in \mathcal{N}(u)} \boldsymbol{T}_{(l)}[u,v] \cdot \boldsymbol{p}_{(l)}[v]$$

Spectral Filter:

$$g(\tilde{\mathbf{L}}) = \sum_{k=0}^{K} \theta_k T^{(k)}(\tilde{\mathbf{L}})$$

Filter — Graph

[7] N Liao, H Liu, Z Zhu, S Luo, L Lakshmanan. "A Comprehensive Benchmark on Spectral GNNs: The Impact on Efficiency, Memory, and Effectiveness". SIGMOD 2026.

pyg_spectral: Graph Filter Taxonomy

GCN $\quad H = (I + \tilde{A})^K \cdot X$

APPNP $\quad H = \sum_{k=0}^{K} \alpha(1-\alpha)^k \tilde{A}^k \cdot X$

GIN $\quad H = (\theta I + \tilde{A})^K \cdot X$

ChebNet $\quad H = \sum_{k=0}^{K} \theta_k T_{\text{Cheb}}^{(k)}(\tilde{A}) \cdot X$

BernNet $\quad H = \sum_{k=0}^{K} \frac{\theta_k}{2^K} \binom{K}{k} (2I - \tilde{L})^{K-k} \tilde{L}^k \cdot X$

AdaGNN $\quad H = (I + \Theta_k \tilde{A})^K \cdot X$

Spectral Filter:

$$g(\tilde{\mathbf{L}}) = \sum_{k=0}^{K} \theta_k T^{(k)}(\tilde{\mathbf{L}})$$

Filter

Graph

[7] N Liao, H Liu, Z Zhu, S Luo, L Lakshmanan. "A Comprehensive Benchmark on Spectral GNNs: The Impact on Efficiency, Memory, and Effectiveness". SIGMOD 2026.

pyg_spectral: Graph Filter Taxonomy

Fixed Filters	GCN	APPNP
	SGC	G²CN ▪ ▪ ▪
Variable Filters	GIN	ChebNet ▪ ▪ ▪
	BernNet	GPRGNN
Composed Filters	AdaGNN	ACMGNN
	FAGNN	FiGURe ▪ ▪ ▪

Spectral Filter:

$$g(\tilde{\mathbf{L}}) = \sum_{k=0}^{K} \theta_k T^{(k)}(\tilde{\mathbf{L}})$$

35 Covered GNNs

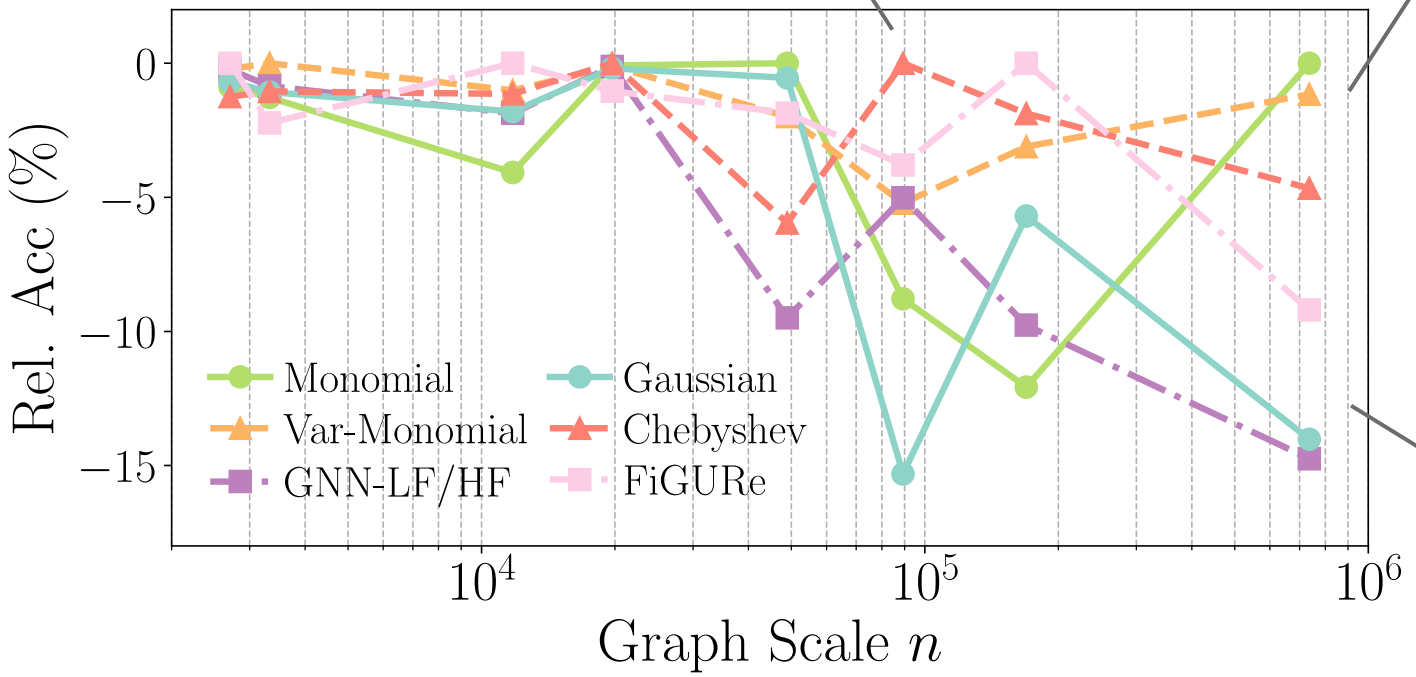
3 Filter Types

[7] N Liao, H Liu, Z Zhu, S Luo, L Lakshmanan. "A Comprehensive Benchmark on Spectral GNNs: The Impact on Efficiency, Memory, and Effectiveness". **SIGMOD 2026**.

pyg_spectral: Key Observations

Efficacy and efficiency are not mutually exclusive.

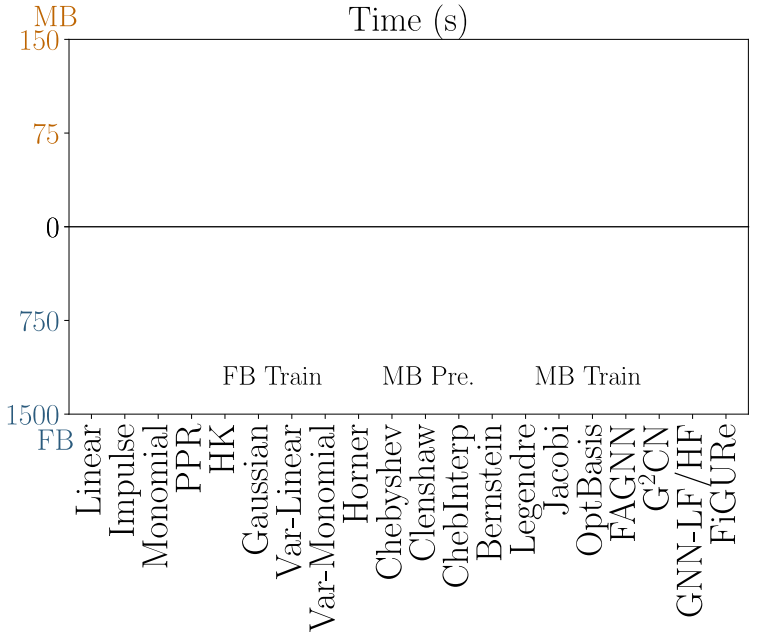
Simple, invariable filters can also capture global view.



Global view is important for large-scale prediction.

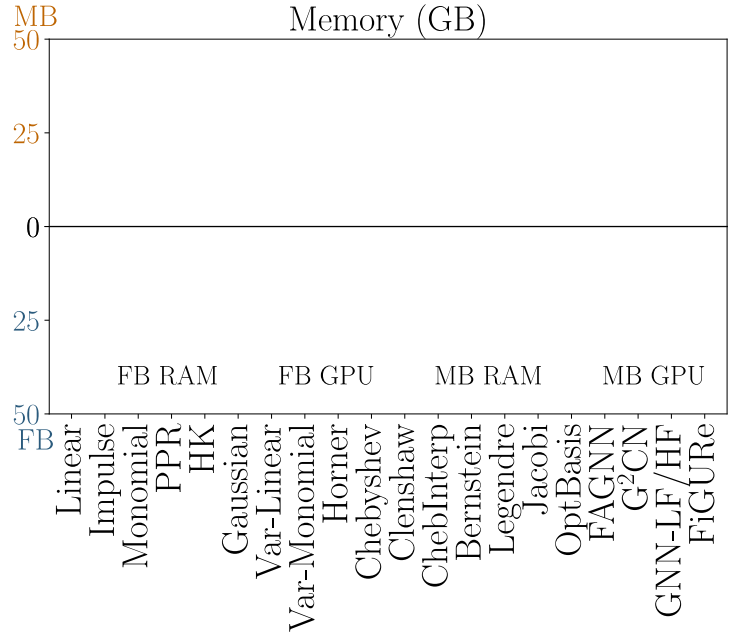
[7] N Liao, H Liu, Z Zhu, S Luo, L Lakshmanan. “A Comprehensive Benchmark on Spectral GNNs: The Impact on Efficiency, Memory, and Effectiveness”. SIGMOD 2026.

pyg_spectral: Key Observations



Time Efficiency: graph operation bottleneck shifts across data scale.

Memory Overhead: Mini-batch training facilitates device utilization.



[7] N Liao, H Liu, Z Zhu, S Luo, L Lakshmanan. "A Comprehensive Benchmark on Spectral GNNs: The Impact on Efficiency, Memory, and Effectiveness". **SIGMOD 2026**.

Takeaway

UNIFEWS: Spectral Sparsification Theory

- Framework: Unify graph & weight operators
- Theory: Bound progressive representation
- Experiment: Maintain accuracy & boost computation

pyg_spectral: Spectral Filter Benchmark

- Framework: Unify iterative & decoupled graph computation
- Benchmark: Evaluate time efficiency, memory footprint, & accuracy
- Insight: Harmonize effectiveness and efficiency

Outline

Introduction

Algorithms

Data Structures

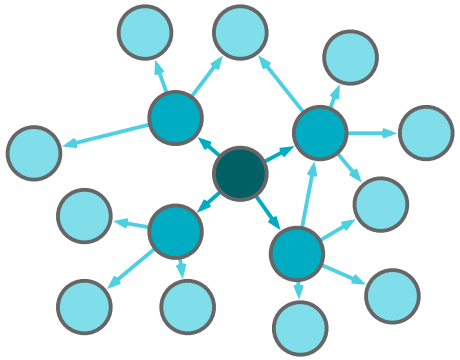
Evaluations

Conclusion and Future Directions

Scalability Challenges and Solutions

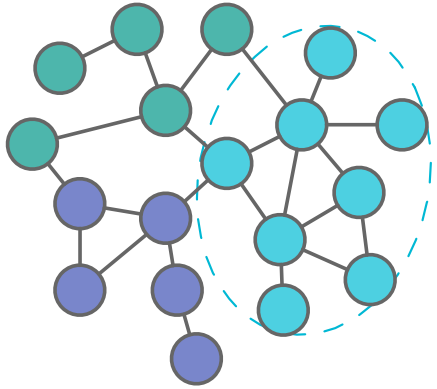
Future Research Directions

Conclusion: Scalability Challenges



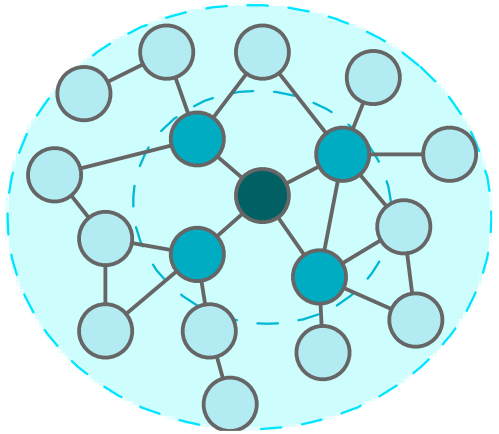
Neighbor Explosion

- Speed
- Expressive



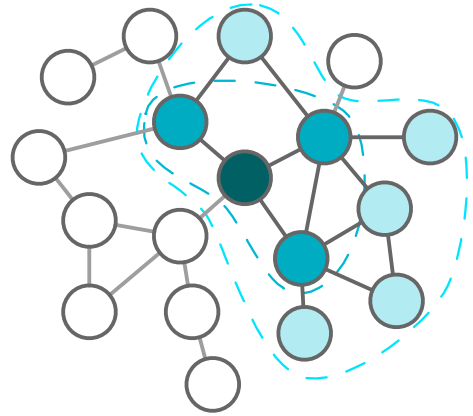
Structure Storage

- Memory
- Minibatch



Long-range

- Speed
- Heterophily
- Expressive



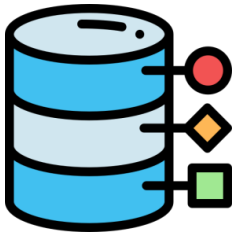
Subgraph

- Speed
- Fine-grained Computation

Conclusion: Scalability Solutions

Design Scalable Algorithms

- Feature-dimension Utilization [VLDB 2022 & VLDBJ 2023]
- Long-distance Embeddings [NeurIPS 2023]



Deploy Scalable Data Structures

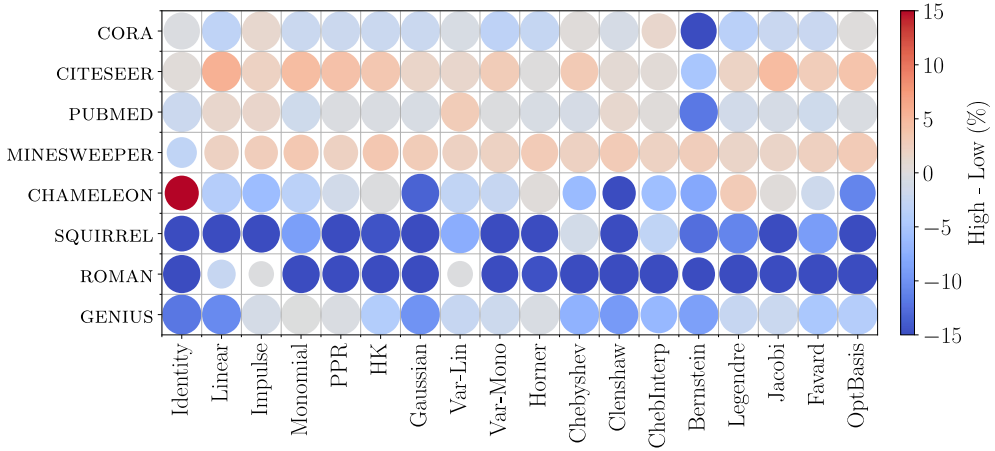
- Hub Labeling for Graph Transformer [NeurIPS 2025]
- Streaming for Subgraph GNN [VLDB 2024]

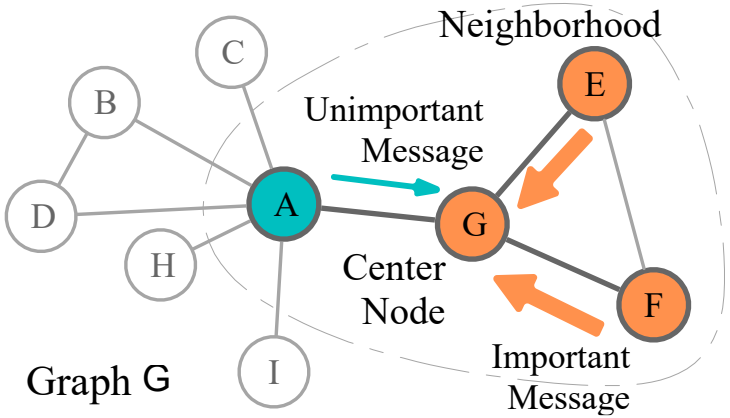
Evaluate Scalability Performance

- Spectral Sparsification Theory [ICML 2025]
- Spectral Filter Benchmark [SIGMOD 2026]

Future: Personalized Graph Analysis

- Biased performance
  - *How to ensure graph learning outcome on specific nodes?*

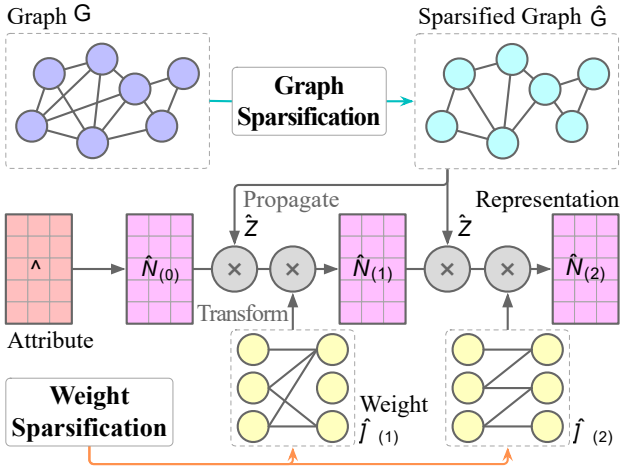




- Personalized scalability
  - *How to perform fast and scalable graph learning on specific nodes?*

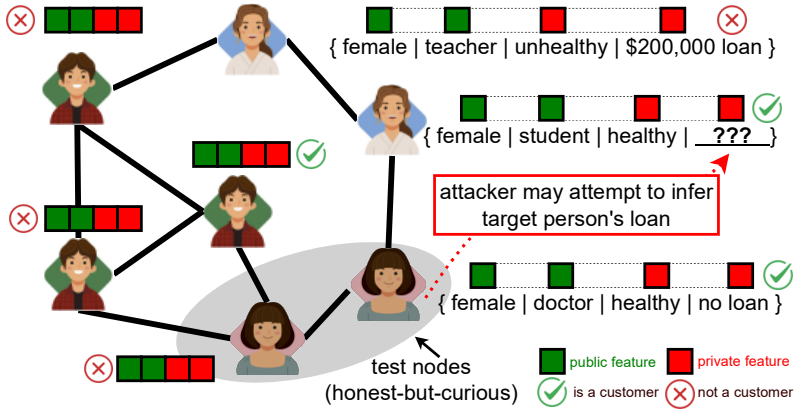
[7] N Liao, H Liu, Z Zhu, S Luo, L Lakshmanan. “A Comprehensive Benchmark on Spectral GNNs: The Impact on Efficiency, Memory, and Effectiveness”. **SIGMOD 2026**.

Future: GNN under Various Requirements



- Computational device
  - *How to deploy GNN training/ inference on different devices?*

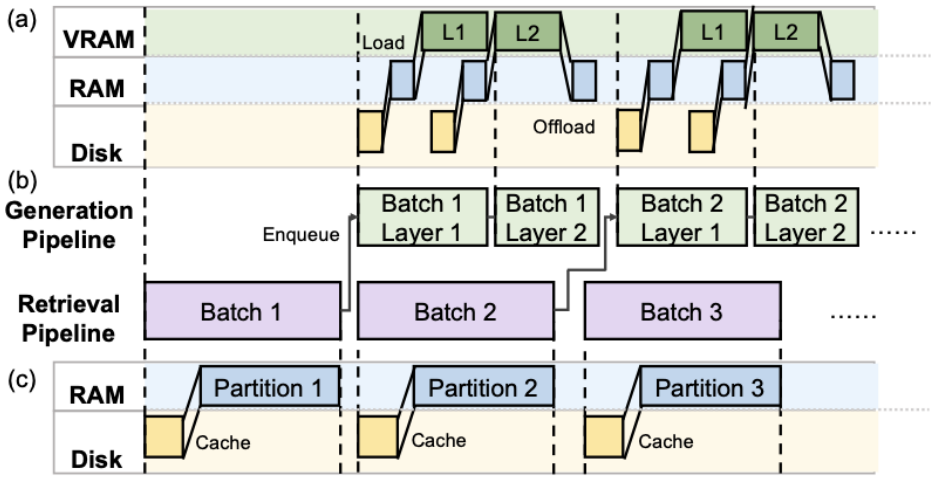
- Privacy & fairness
  - *How to effective protect local graph & attribute information?*

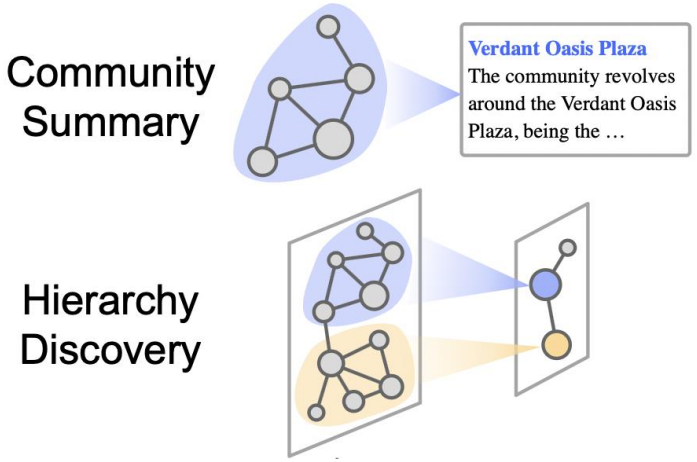


[12] Y Qi, N Liao, S Luo, X Lin, J Li, K Han. "Fine-Grained Differentially Private Graph Neural Network with Node and Layer Noise Adaptation". Under review.

Future: Graph Learning + LLM

- GNN + RAG
  - *How to perform efficient RAG retriev with GNN embeddings?*





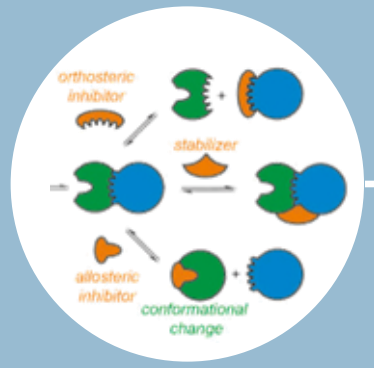
- Locality & explainability
  - *How to utilize local graph structures for explainable LLM inference?*

[13] W Yu*, N Liao*, S Luo, J Liu. “RAGDoll: Efficient Offloading-based Online RAG System on a Single GPU”. arXiv:2504.15302.
[15] I Lim*, N Liao*, Y Yang, G Yip, S Luo. “SEQ-GPT: LLM-assisted Spatial Query via Example”. arXiv:2508.10486.

Vision: Towards Scalable Graph Learning

Image credits: linkedin.com, yerliteknoloji.net.

Friend Suggestion



Drug Discovery

Facial Recognition

Outline

Introduction

Algorithms

Data Structures

Evaluations

Conclusion and Future Directions

Q&A

Publications

[1] N Liao*, D Mo*, S Luo, X Li, P Yin. “SCARA: Scalable Graph Neural Networks with Feature-Oriented Optimization”. **VLDB 2022**.

[2] N Liao, D Mo, S Luo, X Li, P Yin. “Scalable Decoupling Graph Neural Network with Feature-Oriented Optimization”. **VLDBJ 2023**.

[3] N Liao, S Luo, X Li, J Shi. “LD^2: Scalable Heterophilous Graph Neural Network with Decoupled Embedding”. **NeurIPS 2023**.

[4] N Liao*, Z Yu*, S Luo. “HubGT: Fast Graph Transformer with Decoupled Hierarchy Labeling”. **NeurIPS 2025**.

[5] Z Yu*, N Liao*, S Luo. “GENTI: GPU-powered Walk-based Subgraph Extraction for Scalable Representation Learning on Dynamic Graphs”. **VLDB 2024**.

[6] N Liao, Z Yu, R Zeng, S Luo. “UNIFEWS: You Need Fewer Operations for Efficient Graph Neural Network”. **ICML 2025**.

[7] N Liao, H Liu, Z Zhu, S Luo, L Lakshmanan. “A Comprehensive Benchmark on Spectral GNNs: The Impact on Efficiency, Memory, and Effectiveness”. **SIGMOD 2026**.

Publications

[8] H Liu, N Liao, S Luo. “Similarity-based Efficient Global Aggregation for Heterophilous Graph Neural Networks”. **ICDE 2025**.

[9] J Yew, N Liao, D Mo, S Luo. “Example Searcher: A Spatial Query System via Example”. **ICDE Demo 2023**.

[10] K Yow, N Liao, S Luo, R Cheng. “Machine Learning for Subgraph Extraction: Methods, Applications and Challenges”. **VLDB Tutorial 2023**.

[11] N Liao, S Luo, X Xiao, R Cheng. “Advances in Designing Scalable Graph Neural Networks: The Perspective of Graph Data Management”. **SIGMOD Tutorial 2025**.

[12] Y Qi, N Liao, S Luo, X Lin, J Li, K Han. “Fine-Grained Differentially Private Graph Neural Network with Node and Layer Noise Adaptation”. Under review.

[13] W Yu*, N Liao*, S Luo, J Liu. “RAGDoll: Efficient Offloading-based Online RAG System on a Single GPU”. arXiv:2504.15302.

[14] S Zhong, N Liao, D Mo, S Luo. “Efficient Approximate Nearest Neighbor Search with Regular Expression Filtering”. Under review.

[15] I Lim*, N Liao*, Y Yang, G Yip, S Luo. “SEQ-GPT: LLM-assisted Spatial Query via Example”. arXiv:2508.10486.

Acknowledgements

PhD Supervisor: Prof. Siqiang Luo

TAC: Prof. Cheng Long, Prof. Guosheng Lin

Co-author/ Collaborator: Prof. Gao Cong (NTU), Prof. Xiaokui Xiao (NUS), Prof. Laks V.S. Lakshmanan (UBC), Prof. Reynold Cheng (HKU), Prof. Xiang Li (ECNU), Prof. Jieming Shi (PolyU), Dr. Pengcheng Yin (Deepmind), Dr. Cheng Chen (ByteDance)

Colleague/ Mentee: Dingheng Mo, Haoyu Liu, Zihao Yu, Zhulun Zhu, Weiping Yu, Shurui Zhong, Dr. Kai Wang, Dr. Yuxin Qi, Dr. Kai Siong Yow, Jun Xuan Yew, Ruixiao Zeng, Ivan Lim, Gerald Yip

Funding Agency: NTU-WeBank Joint Research Centre