Improving The Scalability of Graph Neural Networks

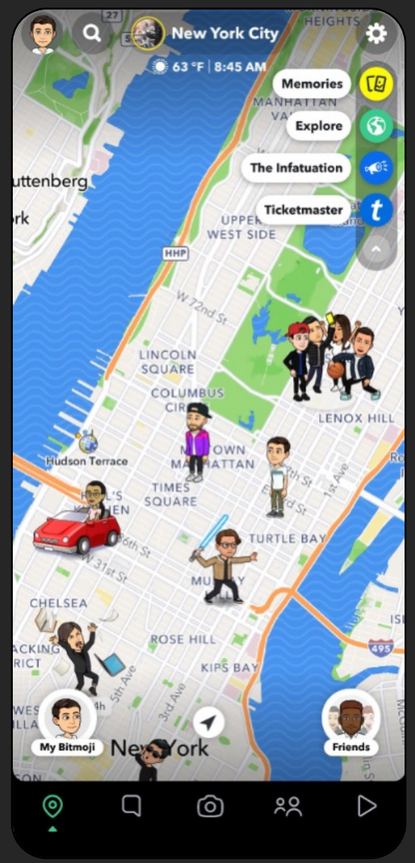
April 9th, 2024
CMU LTI

Neil Shah
Principal Research Scientist, Senior Manager @ Snap

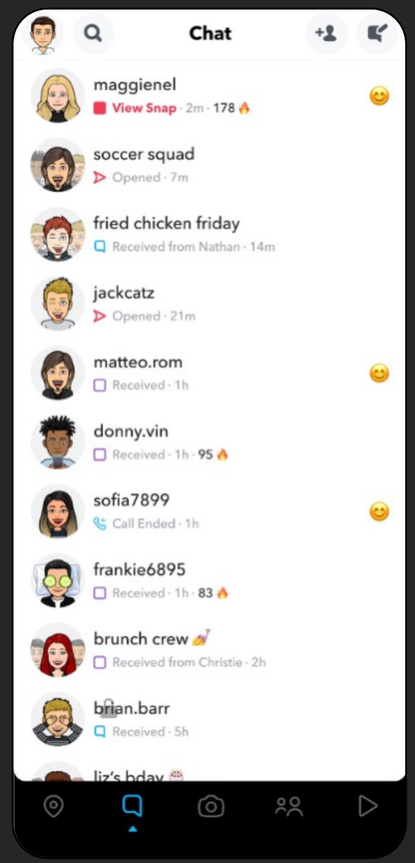
The Snapchat Experience

Five Core Platforms

Map

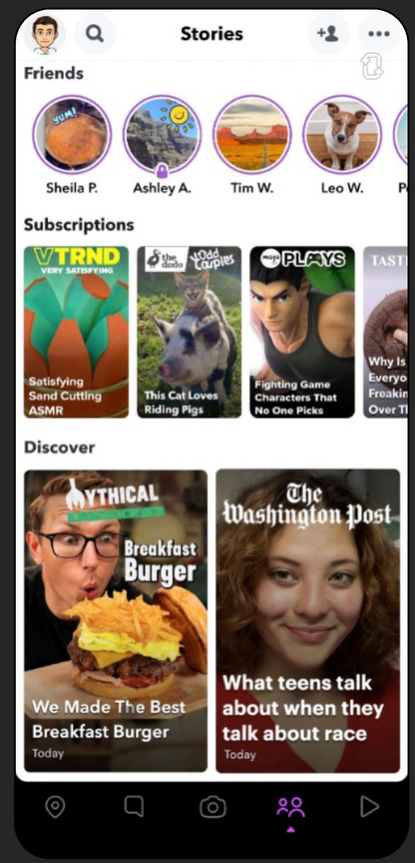


Communications

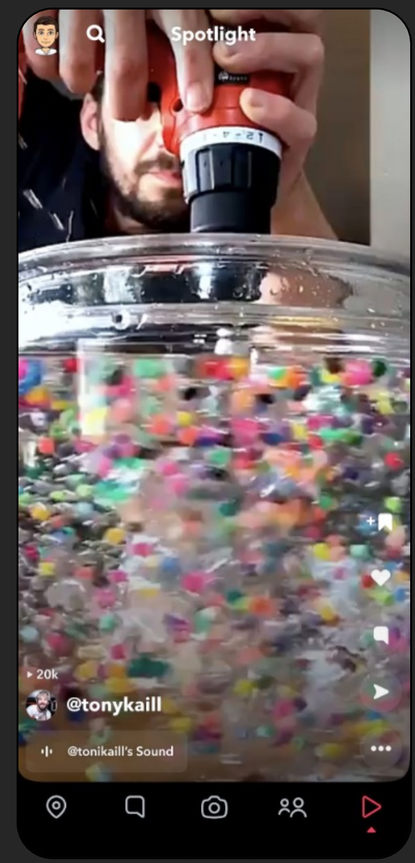


Camera

Stories



Spotlight



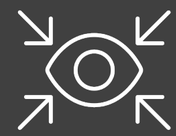
Feb 2023 Investor Presentation (investor.snap.com)

Snap Research

Science Empowering Creativity

Computational Imaging

User Modeling & Personalization

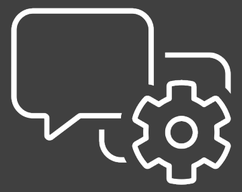


Creative Vision

Snap Research

Science Empowering Creativity

Computational Imaging



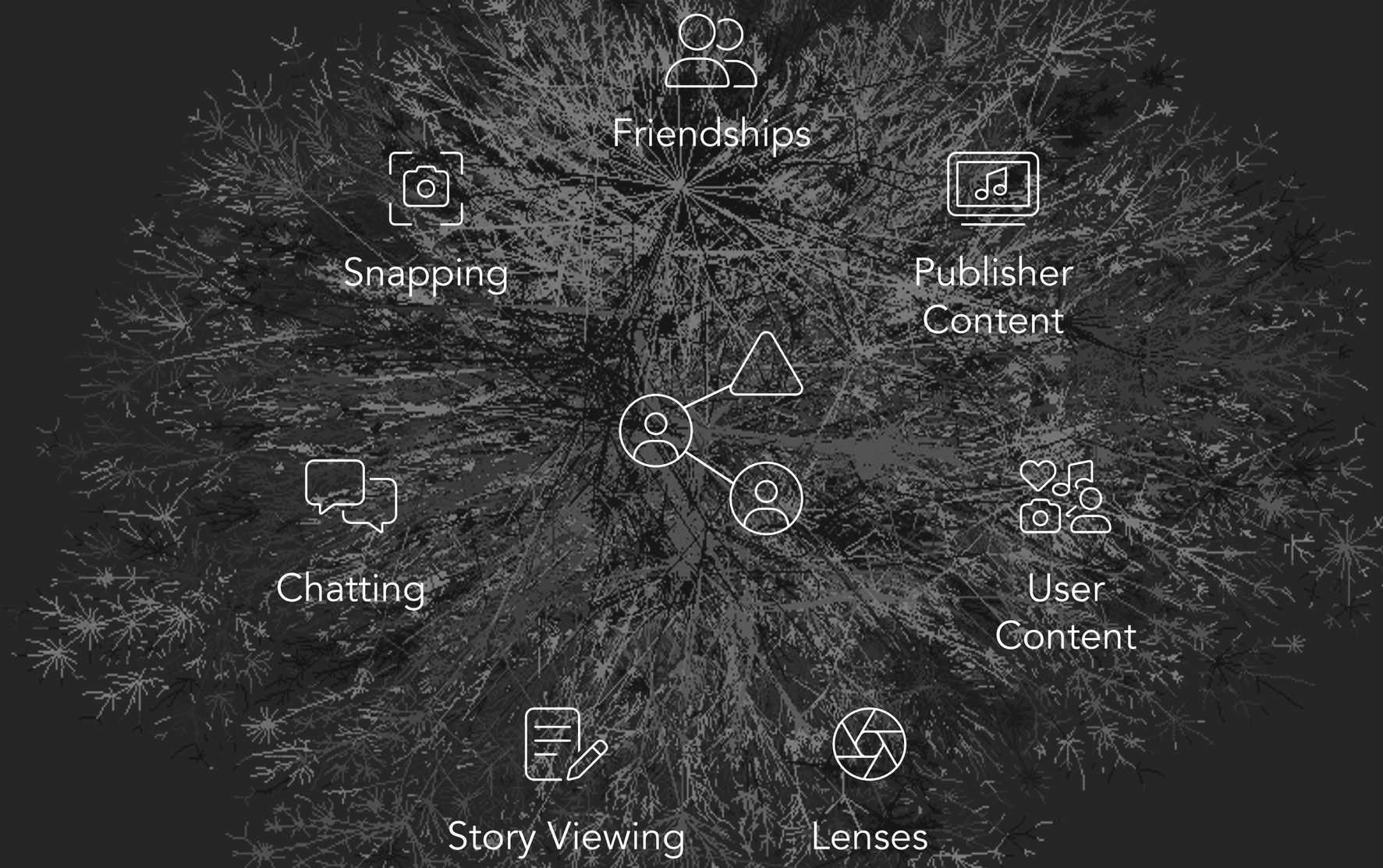
User Modeling & Personalization

Creative Vision

Our team develops innovative techniques for modeling user behavior, empowering our business partners to build world-class user-centric ML Systems.

Many works at ICLR, NeurIPS, ICML, AAAI, KDD, SIGIR, WSDM, etc.

Graphs are Everywhere at Snap



Many Graphs, With Billions of Nodes and Edges.

Embedding Based Retrieval in Friend Recommendation

Jiahui Shi, Vivek Chaurasiya, Yozen Liu, Shubham Vij, Yan Wu, Satya Kanduri
Neil Shah, Peicheng Yu, Nik Srivastava, Lei Shi, Ganesh Venkataraman, Jun Yu
{jshi3,vchaurasiya,yliu2,svij,ywu,satya.kanduri,nshah,peicheng.yu,nsrivastava,lshi3,gvenkataraman,jyu3}@snap.com
Snap Inc.
Santa Monica, CA USA

ABSTRACT

Friend recommendation systems in online social and professional networks such as Snapchat helps users find friends and build connections, leading to better user engagement and retention. Traditional friend recommendation systems take advantage of the principle of locality and use graph traversal to retrieve friend candidates, e.g. Friends-of-Friends (FoF). While this approach has been adopted and shown efficacy in companies with large online networks such as Linkedin and Facebook, it suffers several challenges: *(i)* discrete graph traversal offers limited reach in cold-start settings, *(ii)* it is expensive and infeasible in realtime settings beyond 1 or 2 hop requests owing to latency constraints, and *(iii)* it cannot well-capture the complexity of graph topology or connection strengths, forcing one to resort to other mechanisms to rank and find top-K candidates. In this paper, we proposed a new ***Embedding Based Retrieval (EBR)*** system for retrieving friend candidates, which complements the traditional FoF retrieval by retrieving candidates beyond 2-hop, and providing a natural way to rank FoF candidates. Through online A/B test, we observe statistically significant improvements in the number of friendships made with EBR as an additional retrieval source in both low- and high-density network markets. Our contributions in this work include deploying a novel retrieval system to a large-scale friend recommendation system at Snapchat, generating embeddings for billions of users using Graph Neural Networks, and building EBR infrastructure in production to support Snapchat scale.

KEYWORDS

Social Networks, Friend Recommendation, Embedding Based Retrieval, User Embedding, Graph Neural Network

ACM Reference Format:
Jiahui Shi, Vivek Chaurasiya, Yozen Liu, Shubham Vij, Yan Wu, Satya Kanduri, Neil Shah, Peicheng Yu, Nik Srivastava, Lei Shi, Ganesh Venkataraman, Jun Yu. 2023. Embedding Based Retrieval in Friend Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei,*

1 FRIEND RECOMMENDATION SYSTEMS

In online social and professional networks [2, 6, 19, 20], a user's friends or connections are critical for one's engagement and retention. Research at Linkedin [36] showed that "members with at least 13 connections from companies other than their current employer are 22.9% faster in transitioning to their next job than those who do not". Friend recommendation can be formulated as a link prediction problem [22], where the goal is to predict the links that are to be formed at timestamp T given a snapshot of a social network at timestamp $T-1$. However, unlike classical link prediction problems in academic settings, friend recommendation in online networks operate on hundreds of millions or even billions of users, and evaluating link likelihood between every pair of users is computationally infeasible. Thus, in practice, friend recommendation is often formulated as an industrial recommendation problem and follows a typical large-scale recommendation system architecture [4] which consists of two tiers: *Retrieval* and *Ranking*.

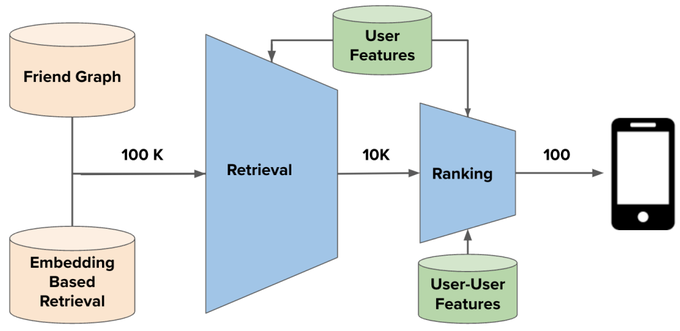
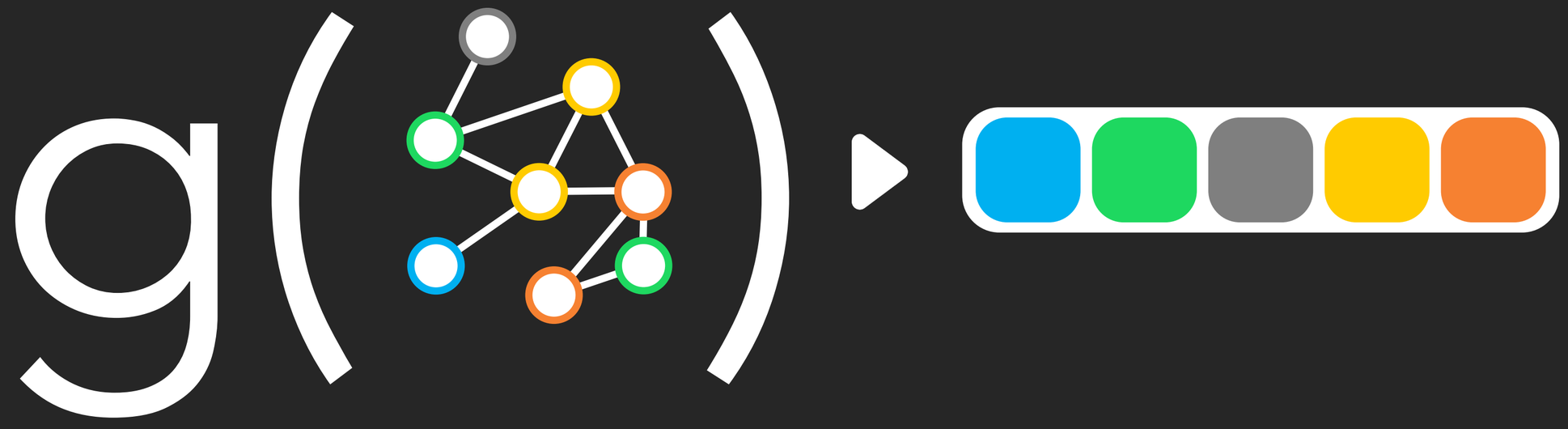


Figure 1: Friend recommendation architecture. We adopt a two-tier setup of coarse-grained retrieval (using FoF and EBR sources), and fine-grained ranking with an ML model. The funnel allows us to prune the space of all friend candidates to a highly personalized subset for each user.

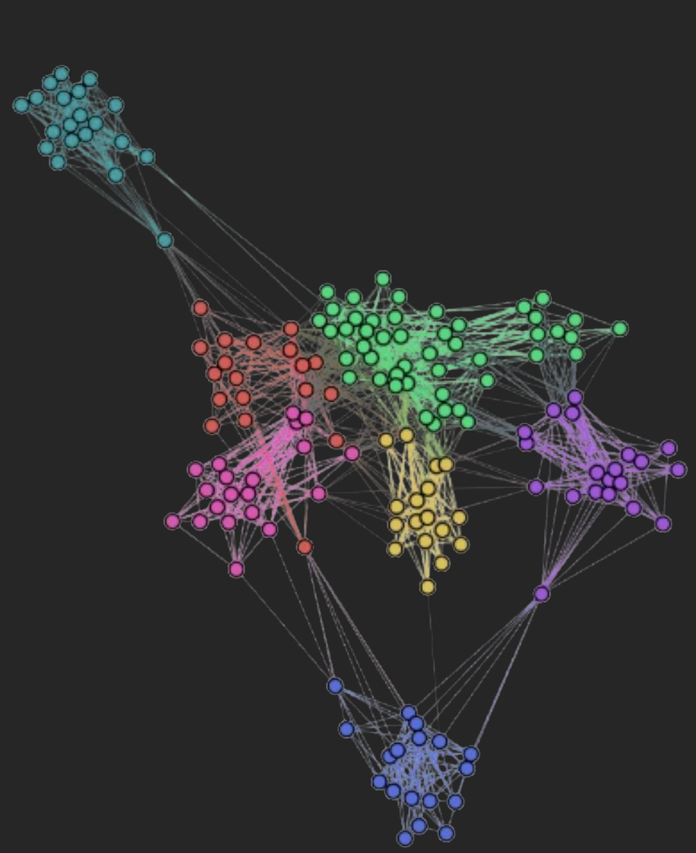
Graph Neural Networks (GNNs)

Neural Network Architectures Designed for Graph Data.



Graph Neural Networks (GNNs)

Neural Network Architectures Designed for Graph Data.



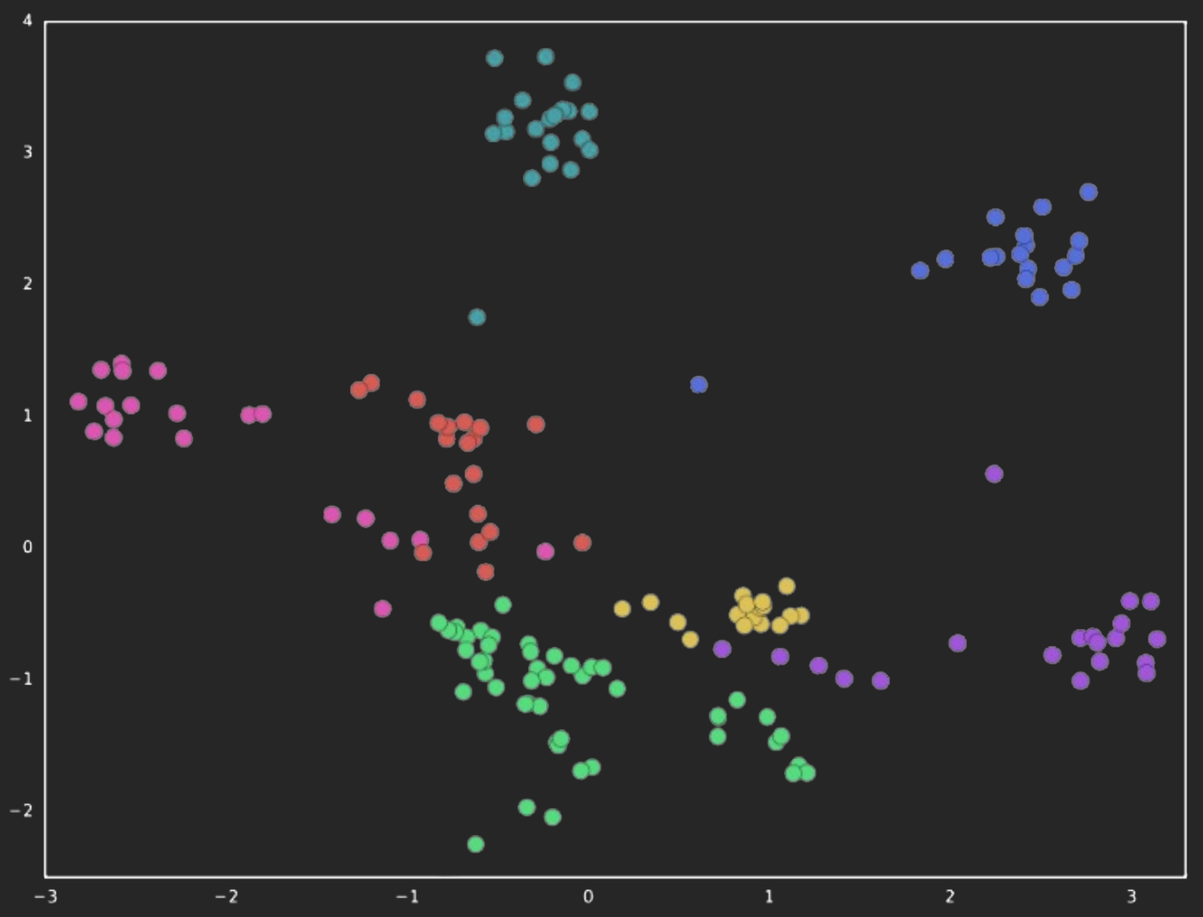
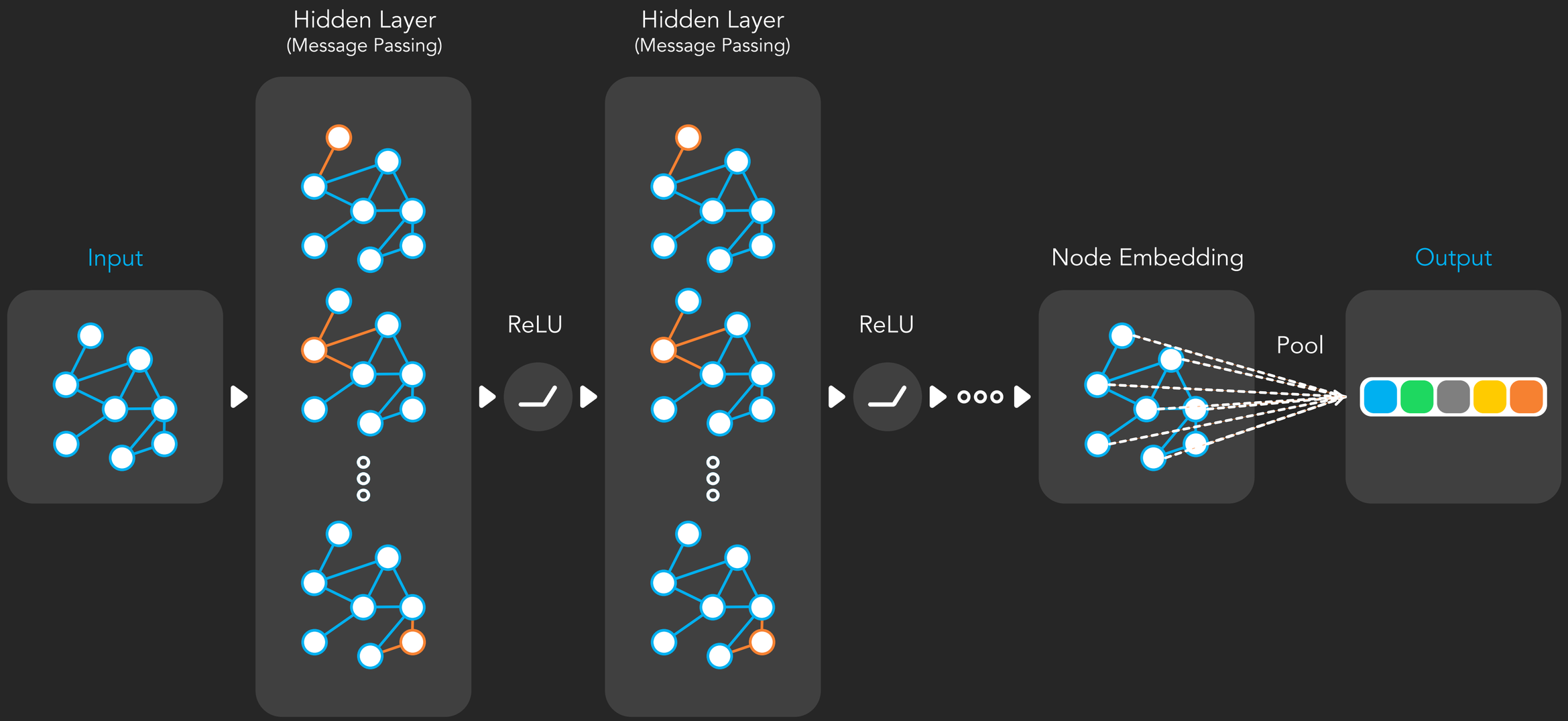


Figure credit: Alessandro Epasto. Innovations in Graph Representation Learning (2019)

Graph Neural Networks (GNNs)

- Key idea: Stack message passing layers and nonlinearities.



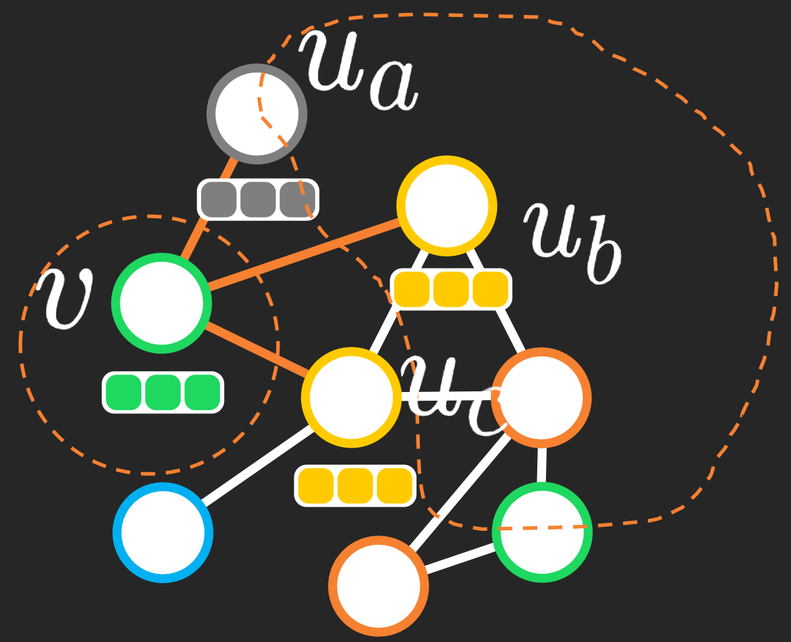
Graph Neural Network (1 layer)

$$h_v^{(t)} = \mathrm{AGG}^{(t)}\left(h_v^{(t-1)}, \left\{\mathrm{MSG}^{(t)}(h_u^{(t-1)}) | u \in \mathcal{N}(v)\right\}\right)$$

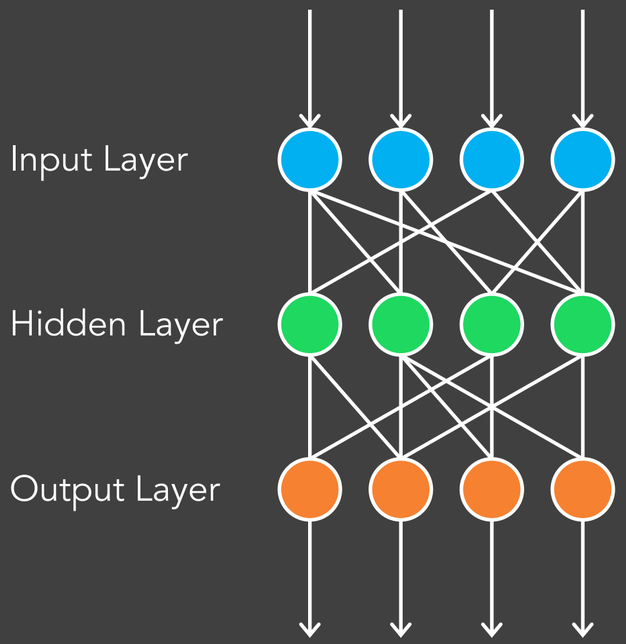
Node emb. Neighbor Messages

$$H^{(t)} = \sigma\left(AH^{(t-1)}W^{(t-1)}\right)$$

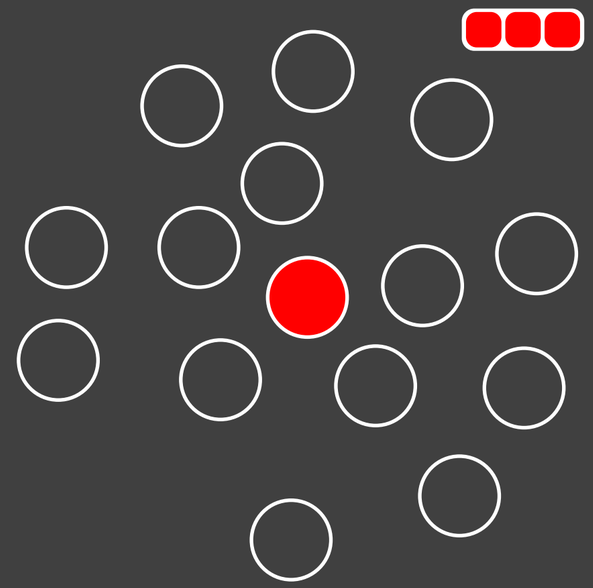
where $H^{(0)} = X$



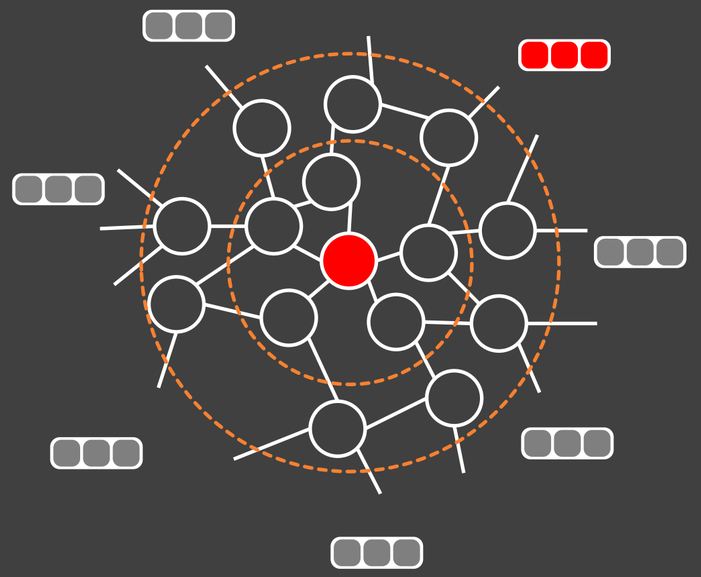
GNNs Empower MLPs with Context



Multi-layer Perceptrons
(tabular machine learning)



Tabular Setting



Relational Setting

GNNs vs. MLPs: Node-Level Tasks

Datasets	GraphSAGE	MLP
Cora	80.52 ± 1.77	59.22 ± 1.31
Citeseer	70.33 ±1.97	59.61 ± 2.88
Pubmed	75.39 ± 2.09	67.55 ± 2.31
A-computer	82.97 ± 2.16	67.80 ± 1.06
A-photo	90.90 ± 0.84	78.77 ± 1.74
Arxiv	70.92 ± 0.17	56.05 ± 0.46
Products	78.61±0.49	62.47 ± 0.10

Node classification accuracy on seven benchmarks.

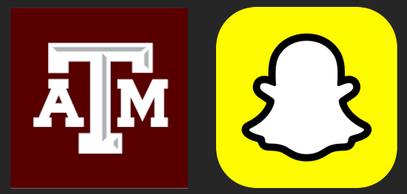
Many Graphs, With Billions of Nodes and Edges.

Graph-less Neural Networks: Teaching Old MLPs new Tricks via Distillation (Zhang et al, ICLR 2022)

Challenges in Scaling GNNs

- Storing very large graphs is non-trivial
- Models are slow to train due to data dependency
- Realtime inference is slower than traditional models
- ...

How Can We Scale Up GNN Training and Inference?



Scaling GNN Training:

MLPInit: Simple GNN Initialization with MLP

MLPInit: Embarrassingly Simple GNN Training Acceleration with MLP Initialization. (ICLR 2023)

By Xiaotian Han, Tong Zhao, Yozen Liu, Xia Hu, Neil Shah.

Why are GNNs Slow to Train?

- Message passing is commonly implemented by sparse matrix multiplication

$$\textbf{GNN:} \quad \mathbf{H}^l = \sigma(\mathbf{A}\mathbf{W}^l_{gnn}\mathbf{H}^{l-1})$$

- Sparse matrix multiplication is very slow (even on GPUs)

Operation	OGB-arXiv		
#Nodes	169343		
#Edges	1166243		
	Forward (ms)	Backward (ms)	Total (ms)
$Z^l = W^l H^{l-1}$	0.32	1.09	1.42
$H^l = AZ^l$	1.09	1028.08	1029.17
			724x

MLPInit: Embarrassingly Simple GNN Training with MLP Initialization (Han et al, ICLR 2023)

Insight: GNNs and MLPs are Similar

$$\text{GNN:} \quad \mathbf{H}^l = \sigma(\mathbf{A}\mathbf{W}^l_{gnn}\mathbf{H}^{l-1}),$$

$$\text{MLP:} \quad \mathbf{H}^l = \sigma(\mathbf{W}^l_{mlp}\mathbf{H}^{l-1}),$$

- GNNs and MLPs can share the same trainable weight space if hidden dimensions align
- We call this model the PeerMLP of the GNN

How do converged PeerMLP weights fare in a GNN?

MLPInit: Embarrassingly Simple GNN Training with MLP Initialization (Han et al, ICLR 2023)

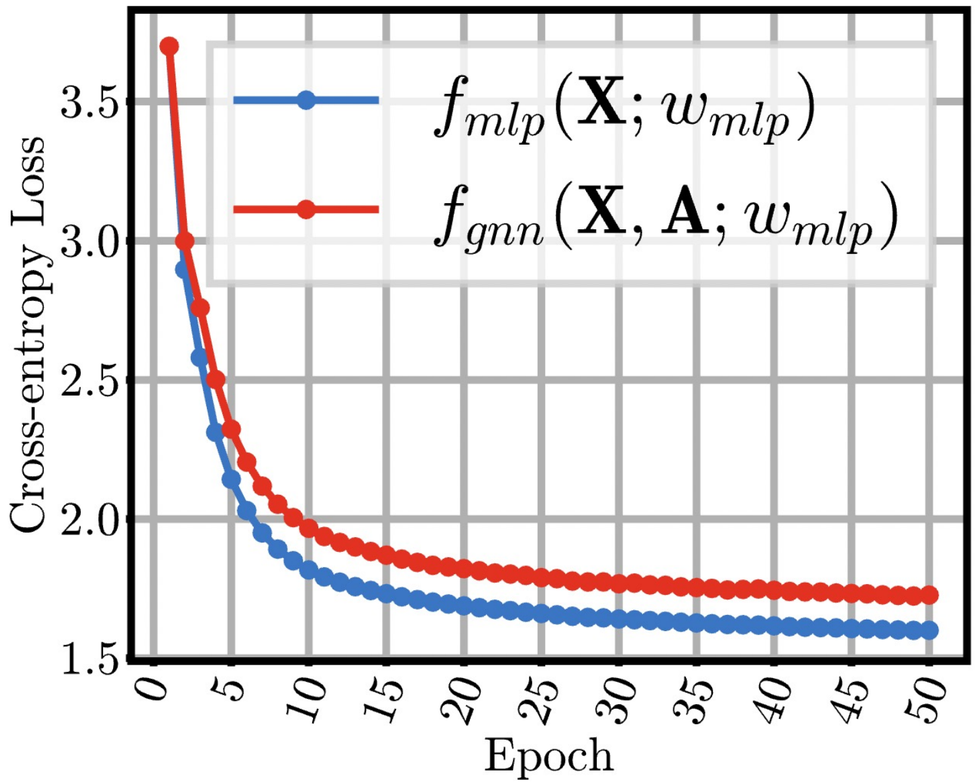
This Works Surprisingly Well (!)

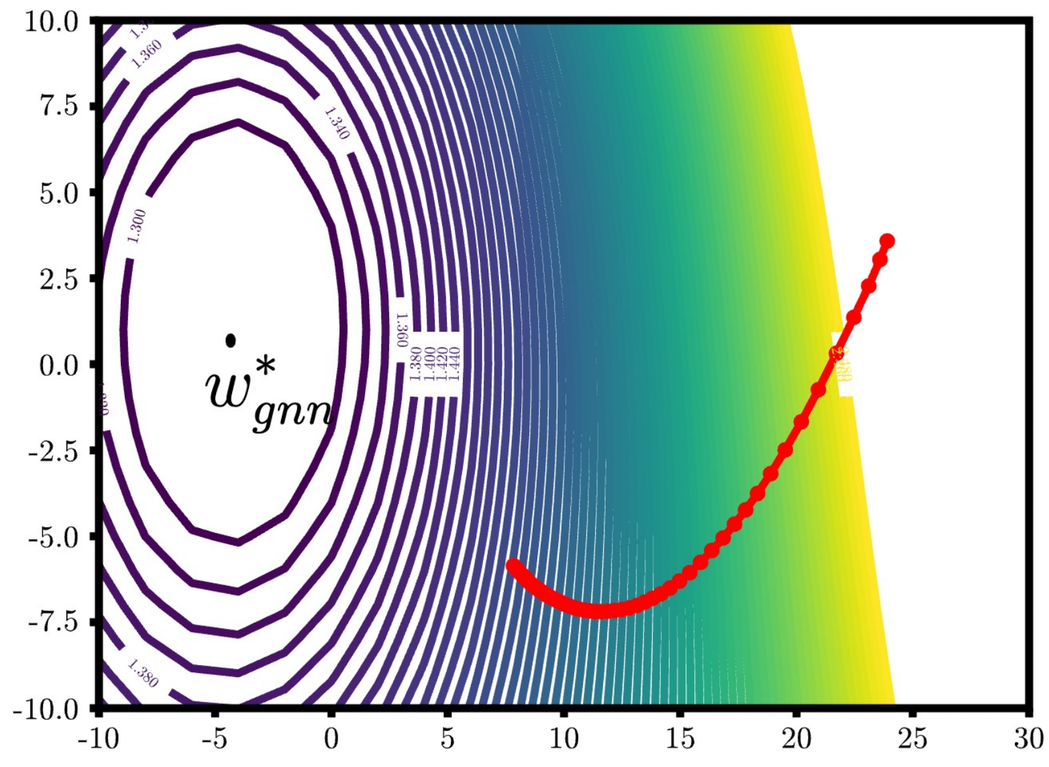
	PeerMLP (ω_{mlp})	GNN (ω_{mlp})	GNN (ω_{gnn})
Cora	58.5	77.6	82.6
CiteSeer	60.5	69.7	71.6
PubMed	73.6	78.1	79.8

- GNN (w/ω_{mlp}): GNN model with weights from PeerMLP.
- GNN (w/ω_{gnn}): GNN model trained from scratch.

MLPInit: Embarrassingly Simple GNN Training with MLP Initialization (Han et al, ICLR 2023)

GNN's Loss Decreases By Optimizing its PeerMLP





Node classification accuracy on seven benchmarks.

MLPInit: Embarrassingly Simple GNN Training with MLP Initialization (Han et al, ICLR 2023)

MLP Initialization (MLPInit) for GNNs

Given a target GNN…

- Construct the PeerMLP.
- Train PeerMLP to convergence and get ω^*_{mlp}
- Initialize GNN with ω^*_{mlp}
- Fine-tune the GNN

Algorithm 1 PyTorch-style Pseudocode of MLPInit

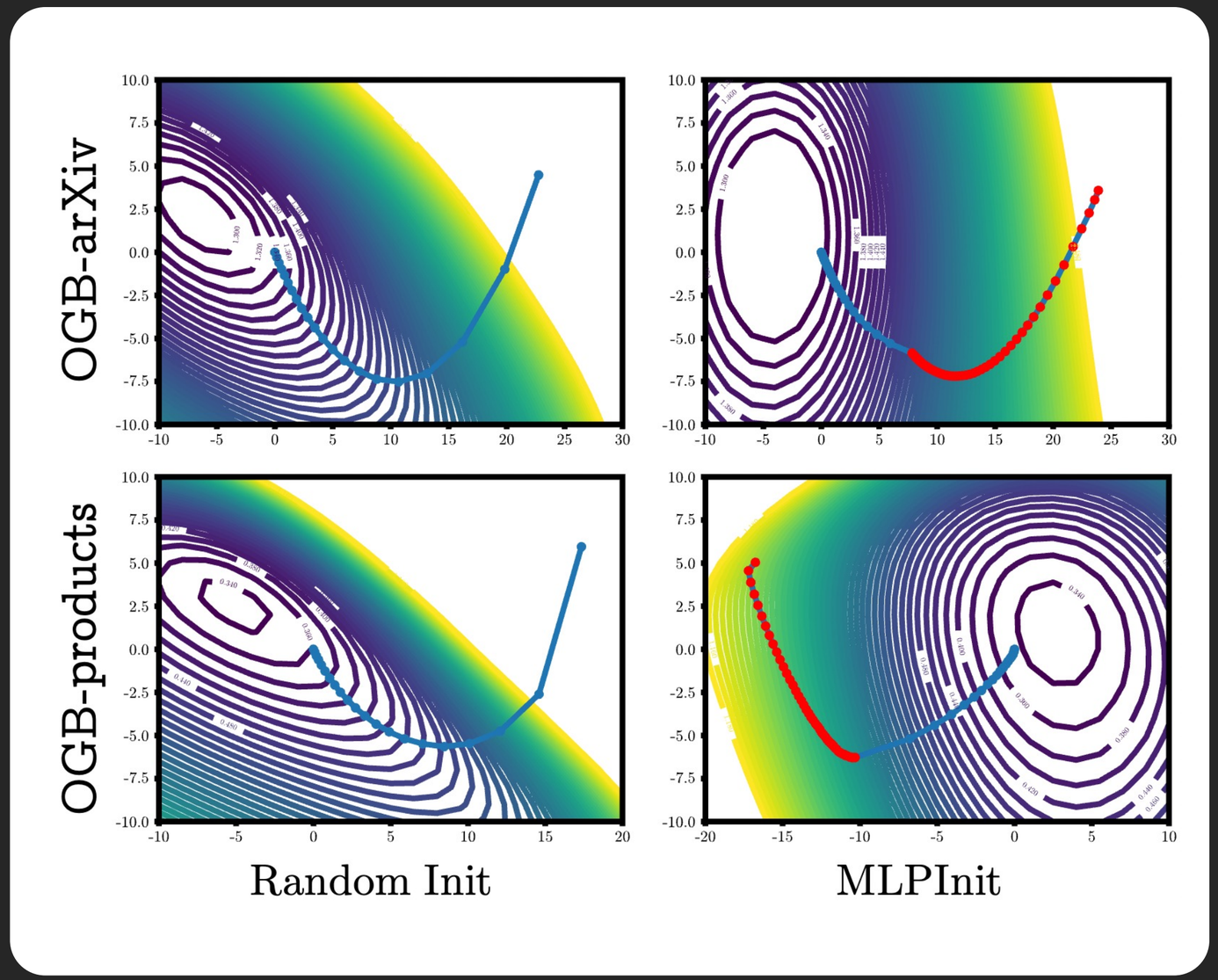
```
# f_gnn: graph neural network model
# f_mlp: PeerMLP of f_gnn

# Train PeerMLP for N epochs
for X, Y in dataloader_mlp:
    P = f_mlp(X)
    loss = nn.CrossEntropyLoss(P, Y)
    loss.backward()
    optimizer_mlp.step()

# Initialize GNN with MLPInit
torch.save(f_mlp.state_dict(), "w_mlp.pt")
f_gnn.load_state_dict("w_mlp.pt")

# Train GNN for n epochs
for X, A, Y in dataloader_gnn:
    P = f_gnn(X, A)
    loss = nn.CrossEntropyLoss(P, Y)
    loss.backward()
    optimizer_gnn.step()
```

MLPInit: Embarrassingly Simple GNN Training with MLP Initialization (Han et al, ICLR 2023)

MLPInit Provides Better Initialization for GNNs

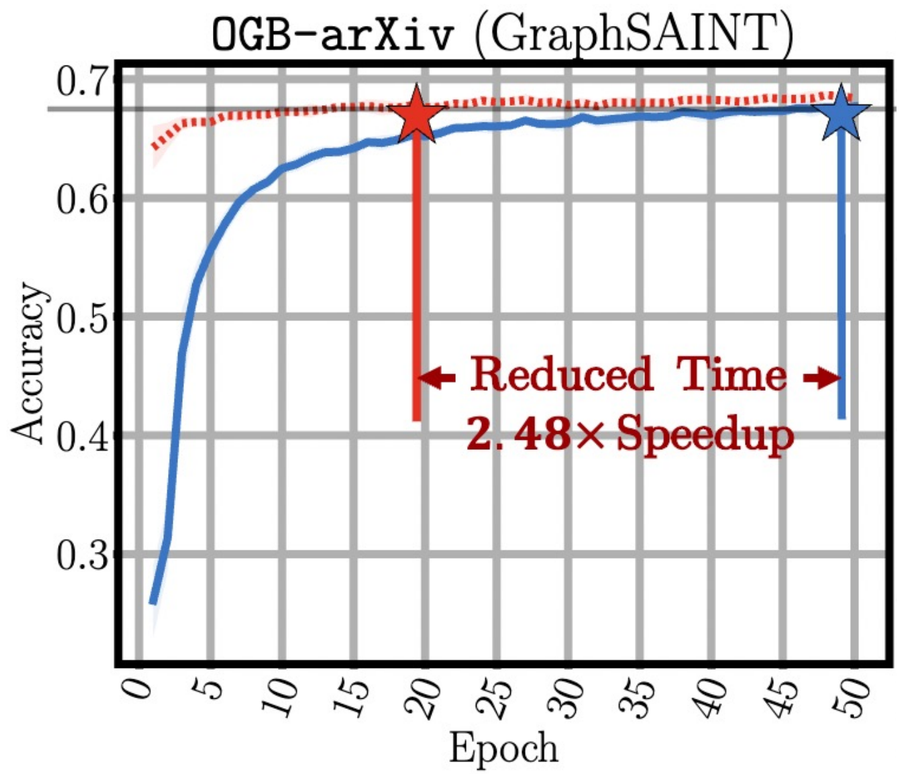
Red dots: PeerMLP training.

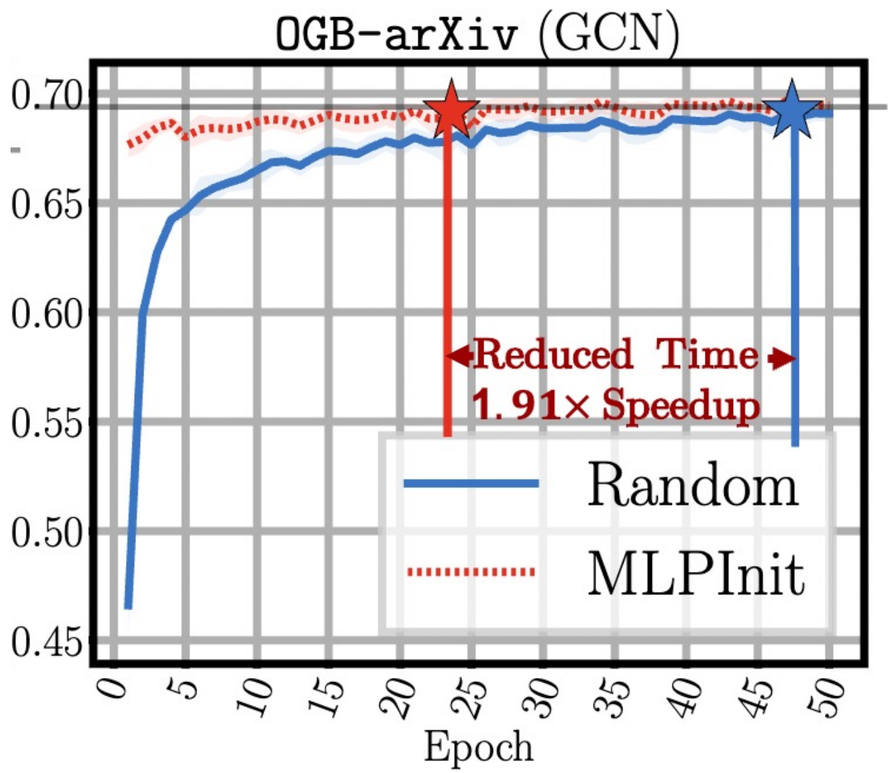
Blue dots: GNN training.



MLPInit: Embarrassingly Simple GNN Training with MLP Initialization (Han et al, ICLR 2023)

MLPInit Helps GNN Converge Faster





★ GNN w/ Random init. Convergence Point.

★ GNN w/ MLPInit
Getting Same Performance.

MLPInit: Embarrassingly Simple GNN Training with MLP Initialization (Han et al, ICLR 2023)

MLPInit Helps GNNs Converge Faster

	Methods	Flickr	Yelp	Reddit	Reddit2	A-Products	OGB-arXiv	OGB-products	Avg.
SAGE	Random(★)	45.6	44.7	36.0	48.0	48.9	46.7	43.0	44.7
	MLPInit (★)	39.9	20.3	7.3	7.7	40.8	22.7	2.9	20.22
	Improv.	1.14x	2.20x	4.93x	6.23x	1.20x	2.06x	14.83x	2.21x
SAINT	Random	31.0	35.8	40.6	28.3	50.0	48.3	44.9	40.51
	MLPInit	14.1	0.0	21.8	6.1	9.1	19.5	16.9	14.58
	Improv.	2.20x	-	1.86x	4.64x	5.49x	2.48x	2.66x	2.77x
C-GCN	Random	15.7	40.3	46.2	47.0	37.4	42.9	42.8	38.9
	MLPInit	7.3	18.0	12.8	17.0	1.0	10.9	15.0	11.7
	Improv.	2.15x	2.24x	3.61x	2.76x	37.40x	3.94x	2.85x	3.32x
GCN	Random	46.4	44.5	42.4	2.4	47.7	46.7	43.8	45.35
	MLPInit	30.5	23.3	0.0	0.0	0.0	24.5	1.3	19.9
	Improv.	1.52x	1.91x	-	-	-	1.91x	33.69x	2.27x

MLPInit: Embarrassingly Simple GNN Training with MLP Initialization (Han et al, ICLR 2023)

MLPInit Improves Node-Level Task Performance

	Methods	Flickr	Yelp	Reddit	Reddit2	A-Products	OGB-arXiv	OGB-products	Avg.
SAGE	Random	53.72±0.16	63.03±0.20	96.50±0.03	51.76±2.53	77.58 ±0.05	72.00±0.16	80.05 ±0.35	70.66
	MLPInit	53.82±0.13	63.93±0.23	96.66±0.04	89.60±1.60	77.74±0.06	72.25±0.30	80.04±0.62	76.29
	Improv.	↑ 0.19%	↑ 1.43%	↑ 0.16%	↑ 73.09%	↑ 0.21%	↑ 0.36%	↓ 0.01%	↑ 7.97%
SAINT	Random	51.37±0.21	29.42±1.32	95.58±0.07	36.45±4.09	59.31±0.12	67.95 ±0.24	73.80±0.58	59.12
	MLPInit	51.35±0.10	43.10±1.13	95.64±0.06	41.71±1.25	68.24±0.17	68.80±0.20	74.02±0.19	63.26
	Improv.	↓ 0.05%	↑ 46.47%	↑ 0.06%	↑ 14.45%	↑ 15.06%	↑ 1.25%	↑ 0.30%	↑ 7.00%
C-GCM	Random	49.95±0.15	56.39±0.64	95.70±0.06	53.79±2.48	52.74±0.28	68.00±0.59	78.71 ±0.59	65.04
	MLPInit	49.96±0.20	58.05±0.56	96.02±0.04	77.77±1.93	55.61±0.17	69.53±0.50	78.48±0,64	69.34
	Improv.	↑ 0.02%	↑ 2.94%	↑ 0.33%	↑ 44.60%	↑ 5.45%	↑ 2.26%	↓ 0.30%	↑ 6.61%
GCM	Random	50.90±0.12	40.08±0.15	92.78±0.h	27.87±3.45	36.35±0.15	70.25±0.22	77.08±0.26	56.47
	MLPInit	51.16±0.20	40.83±0.27	91.40±0.20	80.37±2.61	39.70±0.n	70.35±0.34	76.85±0.34	64.38
	Improv.	↑ 0.51%	↑ 1.87%	↓ 1.49%	↑ 188.42%	↑ 9.22%	↑ 0.14%	↑ 0.29%	↑ 14.00%

MLPInit: Embarrassingly Simple GNN Training with MLP Initialization (Han et al, ICLR 2023)

MLPInit Improves Link-Level Task Performance

	Methods	AUC	AP	Hits@10	Hits@20	Hits@50	Hits@100
PubMed	MLP random	94.76±0.30	94.28±0.36	14.682.60	24.01±3.04	40.02±2.75	54.85 2.03
	GNN random	96.66±0.29	96.78 ±0.31	28.38±6.11	42.55±4.83	60.62±4.29	75.14±3.00
	GNN mlpinit	97.31±0.19	97.53±0.21	37.58±7.52	51.83±7.62	70.57±3.12	81.42±1.52
	Improvement	↑ 0.68%	↑ 0.77%	↑ 32.43%	↑ 21.80%	↑ 16.42%	↑ 8.36%
DBLP	MLP random	95.20±0.18	95.53±0.25	28.70±3.73	39.22±4.13	53.36±3.81	64.83±1.95
	GNN random	96.29±0.20	96.64±0.23	36.55±4.08	43.13±2.85	59.98±2.43	71.57±1.00
	GNN mlpinit	96.67±0.13	97.09±0.14	40.84±7.34	53.72±4.25	67.99±2.85	77.76±1.20
	Improvement	↑ 0.39%	↑ 0.47%	↑ 11.73%	↑ 24.57%	↑ 13.34%	↑ 8.65%
A-Photo	MLP random	86.18±1.41	85.37±1.24	4.36±1.14	6.96±1.28	12.20±1.24	17.91±1.26
	GNN random	92.07±2.14	91.52±2.08	9.63±1.58	12.82±1.72	20.90±1.90	29.08±2.53
	GNN mlpinit	93.99±0.58	93.32±0.60	9.17±2.12	13.12±2.11	22.93±2.56	32.37±1.89
	Improvement	↑ 2.08%	↑ 1.97%	↓ 4.75%	↑ 2.28%	↑ 9.73%	↑ 11.32%
Physics	MLP random	96.26±0.11	95.63±0.15	5.38±1.32	8.76±1.37	15.86±0.81	24.70±1.11
	GNN random	95.84±0.13	95.38±0.15	6.62±1.00	10.39±1.04	18.55±1.60	26.88±1.95
	GNN mlpinit	96.89±0.07	96.55±0.11	8.05±1.44	13.06±1.94	22.38±1.94	32.31±1.43
	Improvement	↑ 1.10%	↑ 1.22%	↑ 21.63%	↑ 25.76%	↑ 20.63%	↑ 20.20%
	Avg.	↑ 1.05%	↑ 1.10%	↑ 17.81%	↑ 20.97%	↑ 14.88%	↑ 10.46%

MLPInit: Embarrassingly Simple GNN Training with MLP Initialization (Han et al, ICLR 2023)

Limitations & Future Work

- Understanding where else MLPInit can work (graph classification, heterogeneous graphs, other architectures?)

- Understanding what breaks MLPInit (heterophilous graphs, low feature-label correlation settings)

MLPInit: Embarrassingly Simple GNN Training with MLP Initialization (Han et al, ICLR 2023)

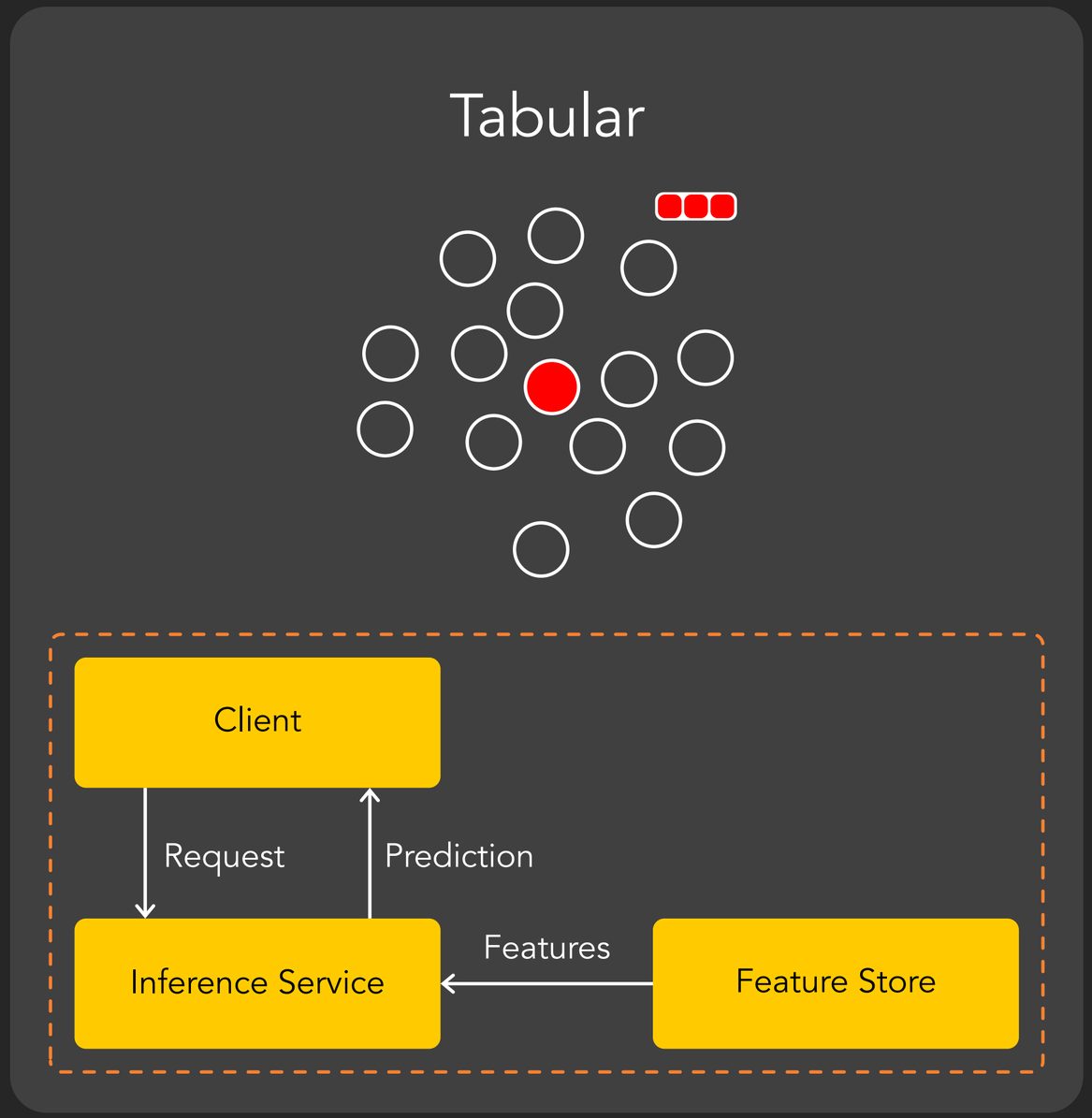
Scaling GNN Inferencing:

Graph-less Graph Learning via Knowledge Distillation

Graph-less Neural Networks: Teaching old MLPs new Tricks via Distillation. (ICLR 2022)
By Shichang Zhang, Yozen Liu, Yizhou Sun, Neil Shah.

Link-less Link Prediction via Relational Distillation. (ICML 2023)
By Zhichun Guo, William Shiao, Shichang Zhang, Yozen Liu, Nitesh Chawla, Neil Shah, Tong Zhao.

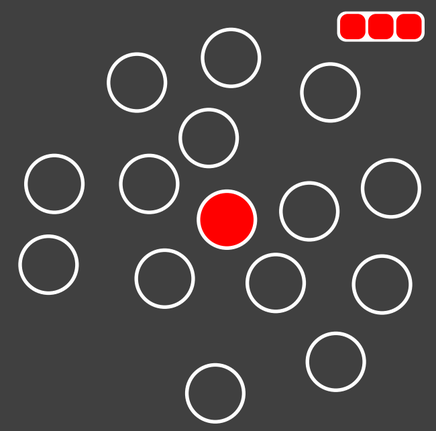
Tabular vs. Relational Real-time Inference

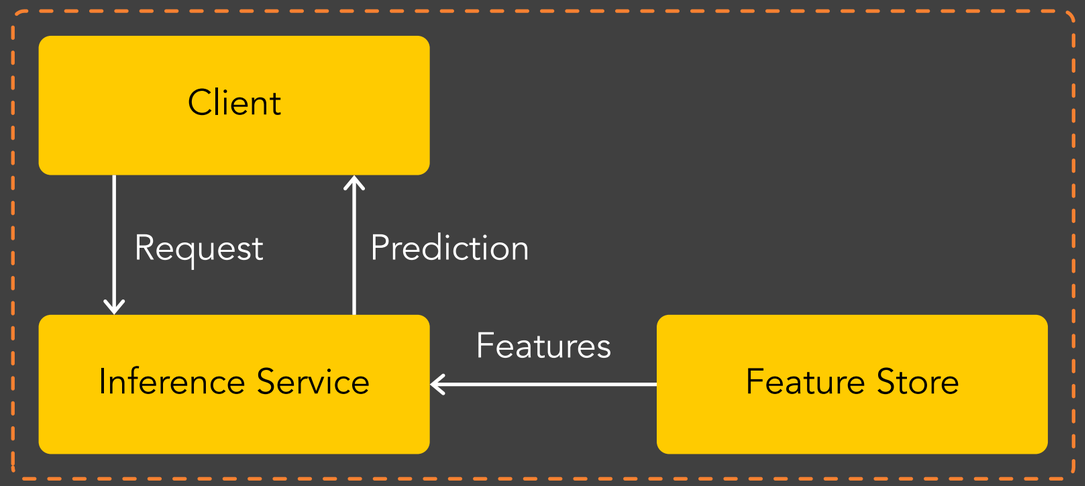


Graph-less Neural Networks: Teaching Old MLPs new Tricks with Distillation (Zhang et al, ICLR 2022)

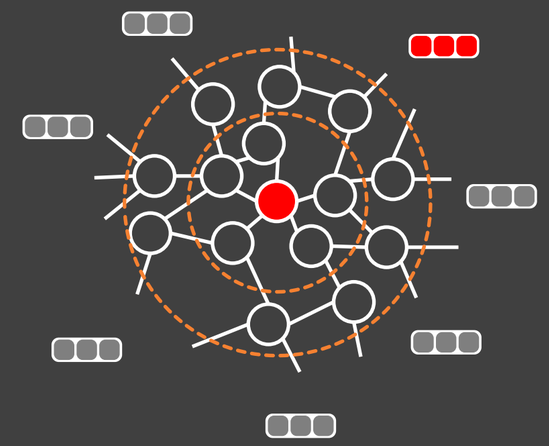
Tabular vs. Relational Real-time Inference

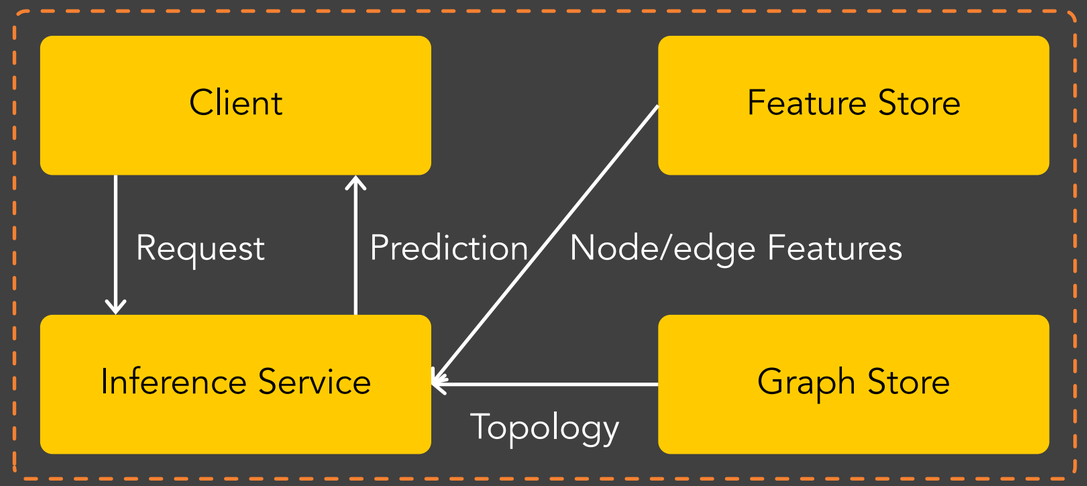
Tabular





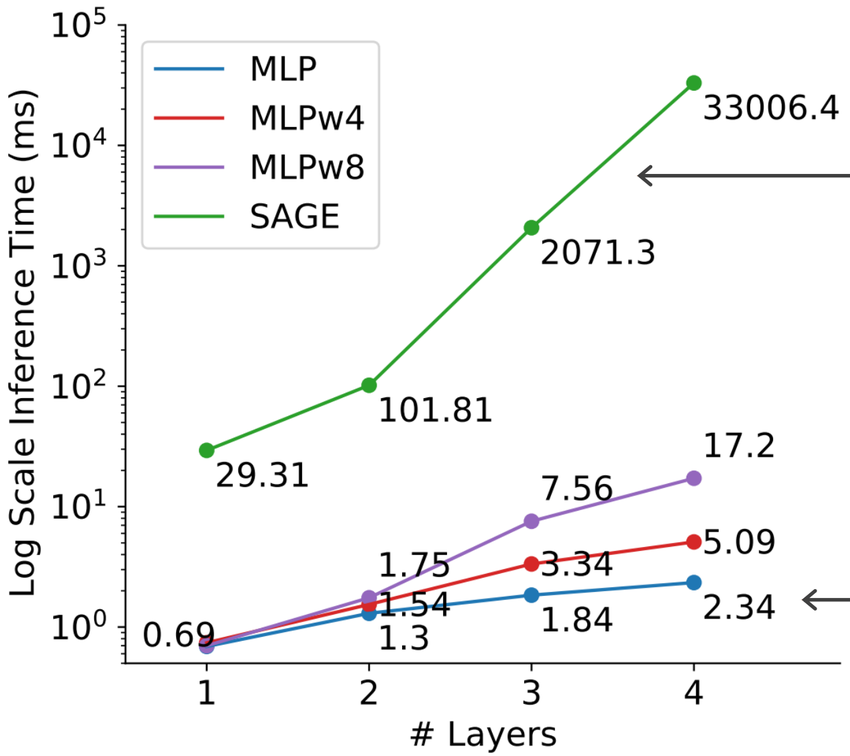
Relational





Graph-less Neural Networks: Teaching Old MLPs new Tricks with Distillation (Zhang et al, ICLR 2022)

GNNs Infer Much More Slowly than MLPs



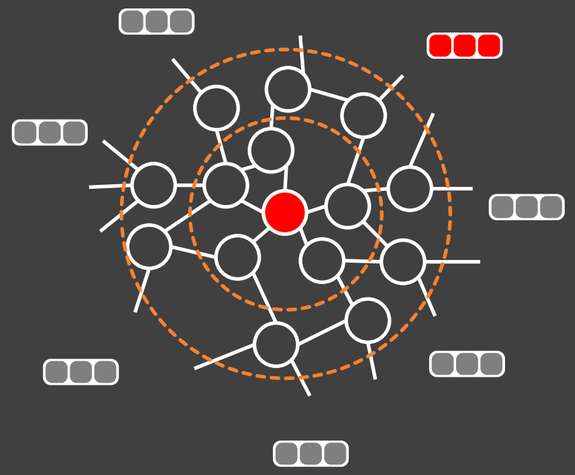
GNN: Exponential in # layers

MLP: Linear in # layers

- Infer 10 randomly selected nodes (graph of 2.5M nodes)
- Inference time = fetching data + forward pass

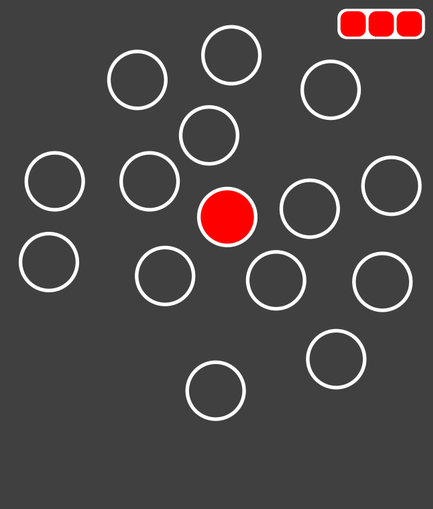
Graph-less Neural Networks: Teaching Old MLPs new Tricks with Distillation (Zhang et al, ICLR 2022)

GNN and MLP: Reconciling Differences



Accurate GNN:

- Uses graph in training
- Uses graph in inference



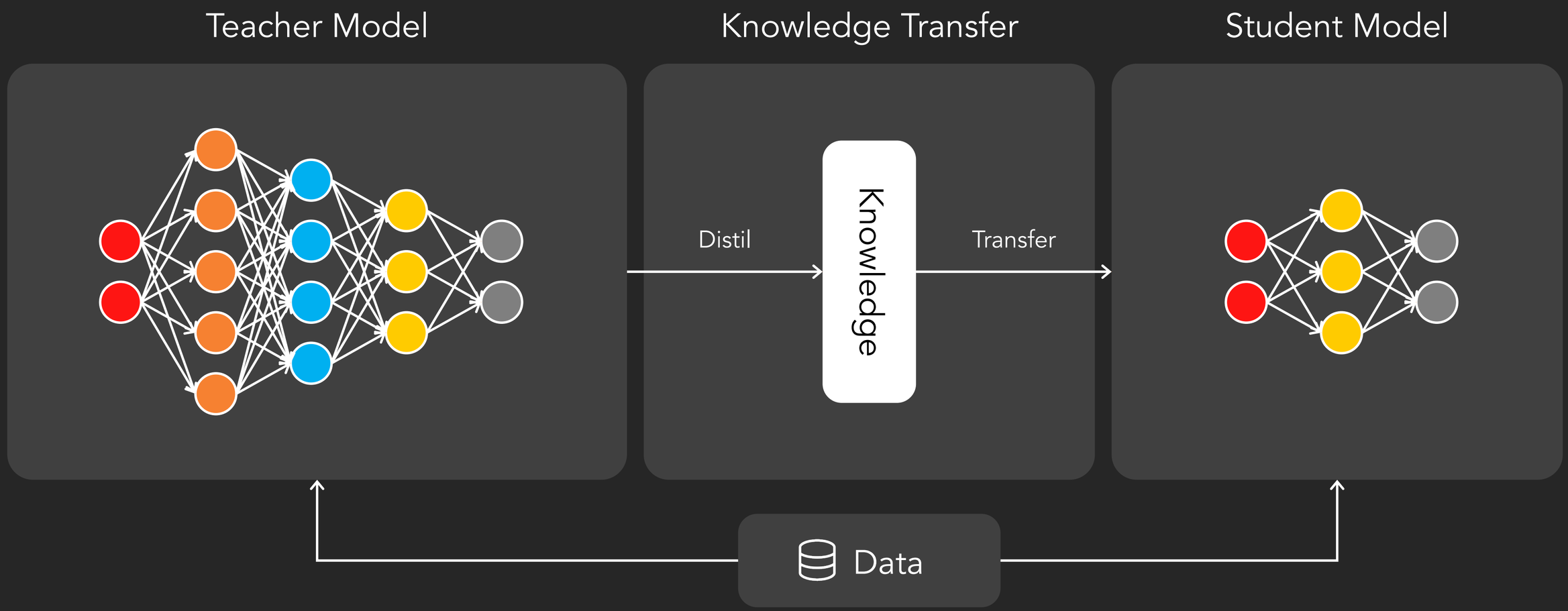
Fast MLP:

- Doesn’t use graph in training
- Doesn’t use graph in inference

Can we get context-awareness without using the graph in inference?

Graph-less Neural Networks: Teaching Old MLPs new Tricks with Distillation (Zhang et al, ICLR 2022)

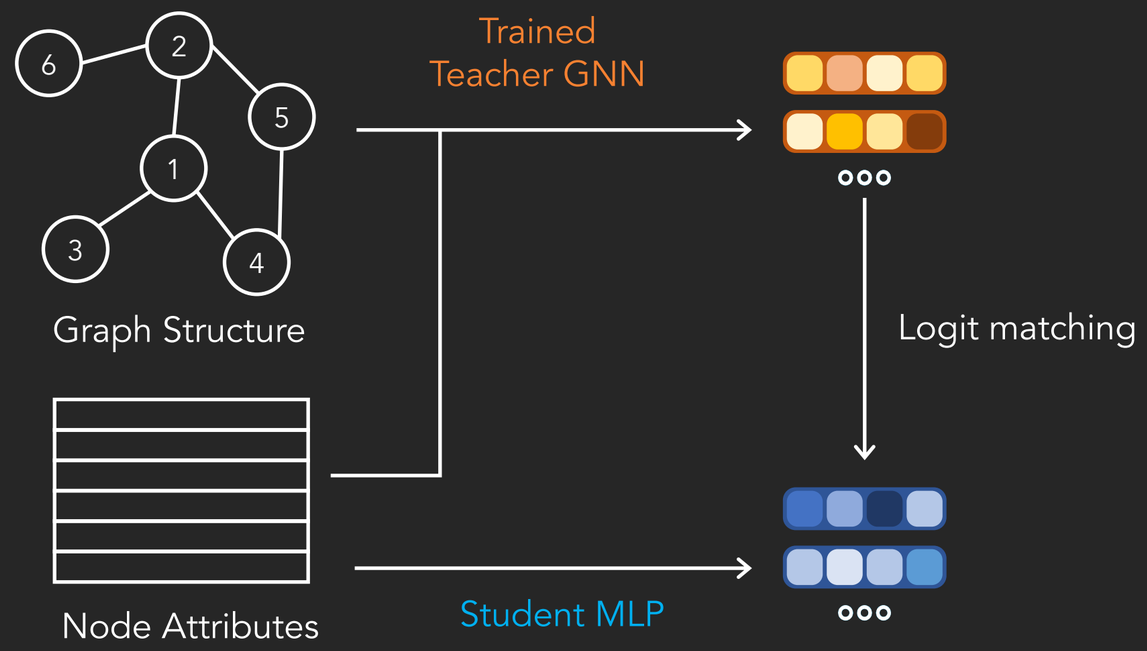
Context: Knowledge Distillation



Knowledge Distillation Transfers The Knowledge From A Teacher Model To A Student Model.

Graph-less Neural Networks: Teaching Old MLPs new Tricks with Distillation (Zhang et al, ICLR 2022)

Graph-Less Neural Networks (GLNN)



- Offline training: use graph to train GNN; distill to MLP
- Online prediction: faster, more accurate graph-less inference for new nodes

Graph-less Neural Networks: Teaching Old MLPs new Tricks with Distillation (Zhang et al, ICLR 2022)

How to Train GLNNs

- Train a teacher GNN and produce logits for each node.
- Train a student GLNN (MLP) using the below loss.

Cross entropy loss

MSE

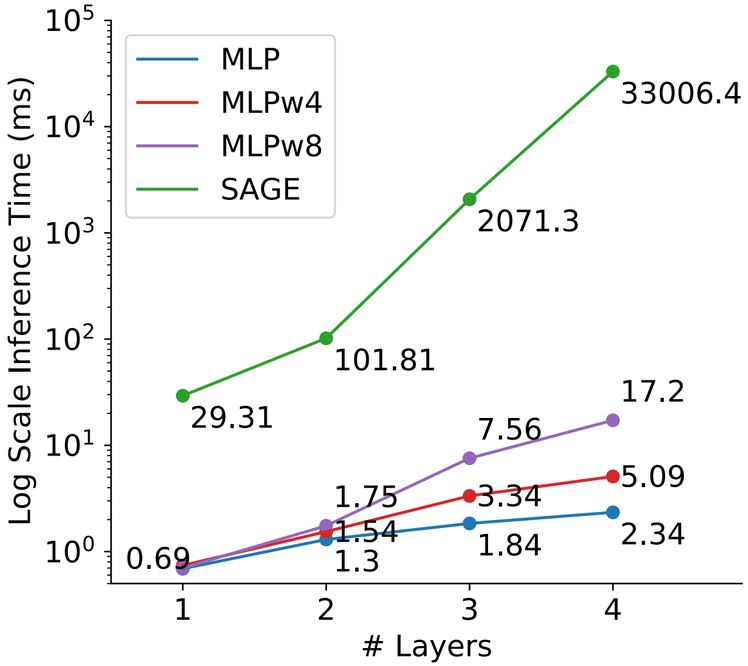
$$\mathcal{L} = \lambda \Sigma_{v \in \mathcal{V}^L} \mathcal{L}_{label}(\hat{\boldsymbol{y}}_v, \boldsymbol{y}_v) + (1 - \lambda) \Sigma_{v \in \mathcal{V}} \mathcal{L}_{teacher}(\hat{\boldsymbol{y}}_v, \boldsymbol{z}_v)$$

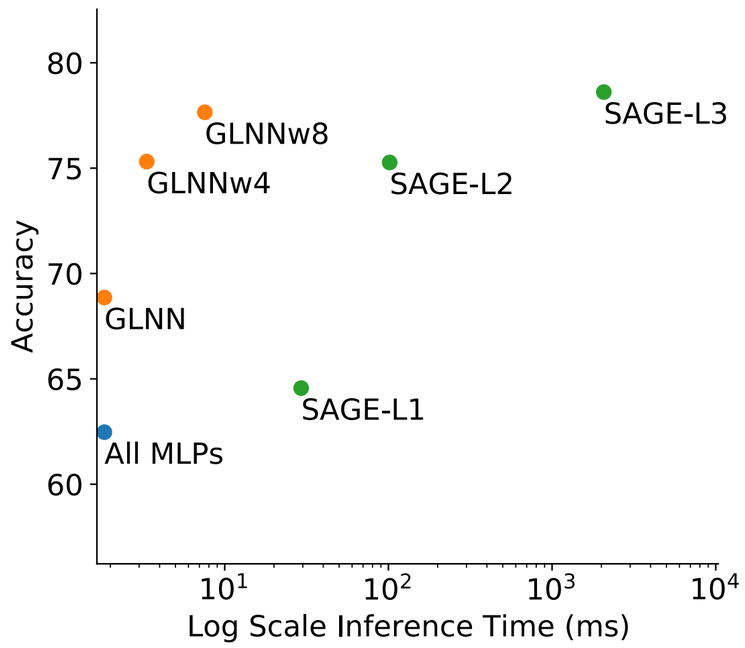
Encourage MLP to correctly predict labeled nodes from logits

Encourage MLP/GNN logit consistency for all nodes

Graph-less Neural Networks: Teaching Old MLPs new Tricks with Distillation (Zhang et al, ICLR 2022)

Trade-offs Between Speed and Accuracy





- MLPs/GLNNs are **much faster** than GNNs
- This holds even with wider MLPs/GLNNs

- GLNNs are significantly **more accurate** than MLPs
- GLNNs are **comparably accurate** to GNNs while being **much faster**

Graph-less Neural Networks: Teaching Old MLPs new Tricks with Distillation (Zhang et al, ICLR 2022)

GLNN Empower MLPs To Be Like GNNs

Datasets	Eval	SAGE	MLP/MLP+	GLNN/GLNN+
Cora	prod	**79.29**	58.98	78.28
	ind	81.33±2.19	59.09±2.96	73.82±1.93
	tran	78.78±1.92	58.95±1.66	79.39±1.64
Citeseer	prod	68.38	59.81	**69.27**
	ind	69.75 ±3.59	60.06±5.00	69.25±2.25
	tran	68.04±3.34	59.75±2.48	69.28±3.12
Pubmed	prod	**74.88**	66.80	74.71
	ind	75.26±2.57	66.85±2.96	74.30±2.61
	tran	74.78±2.22	66.79±2.90	74.81±2.39
A computer	prod	82.14	67.38	**82.29**
	ind	82.08±1.79	67.84±1.78	80.92±1.36
	tran	82.15 1.55	67.27±1.36	82.63±1.40
A-photo	prod	91.08	79.25	**92.38**
	ind	91.50±0.79	79.44±1.72	91.18±0.81
	tran	90.80±0.77	79.20±1.64	92.68±0.56
Arxiv	prod	**70.73**	55.30	65.09
	ind	70.64±0.67	55.40±0.56	60.48±0.46
	tran	70.75 ± 0.27	55.28±0.49	71.46±0.33
Products S	prod	**76.60**	63.72	75.77
	ind	76.89±0.53	63.70±0.66	75.16±0.34
	tran	76.53±0.55	63.73±0.69	75.92±0.61

Significant accuracy improvement over MLPs.

Competitive accuracy to GNNs on 6/7 datasets.

Graph-less Neural Networks: Teaching Old MLPs new Tricks with Distillation (Zhang et al, ICLR 2022)

GLNNs Offer Speedups Over Acceleration Methods

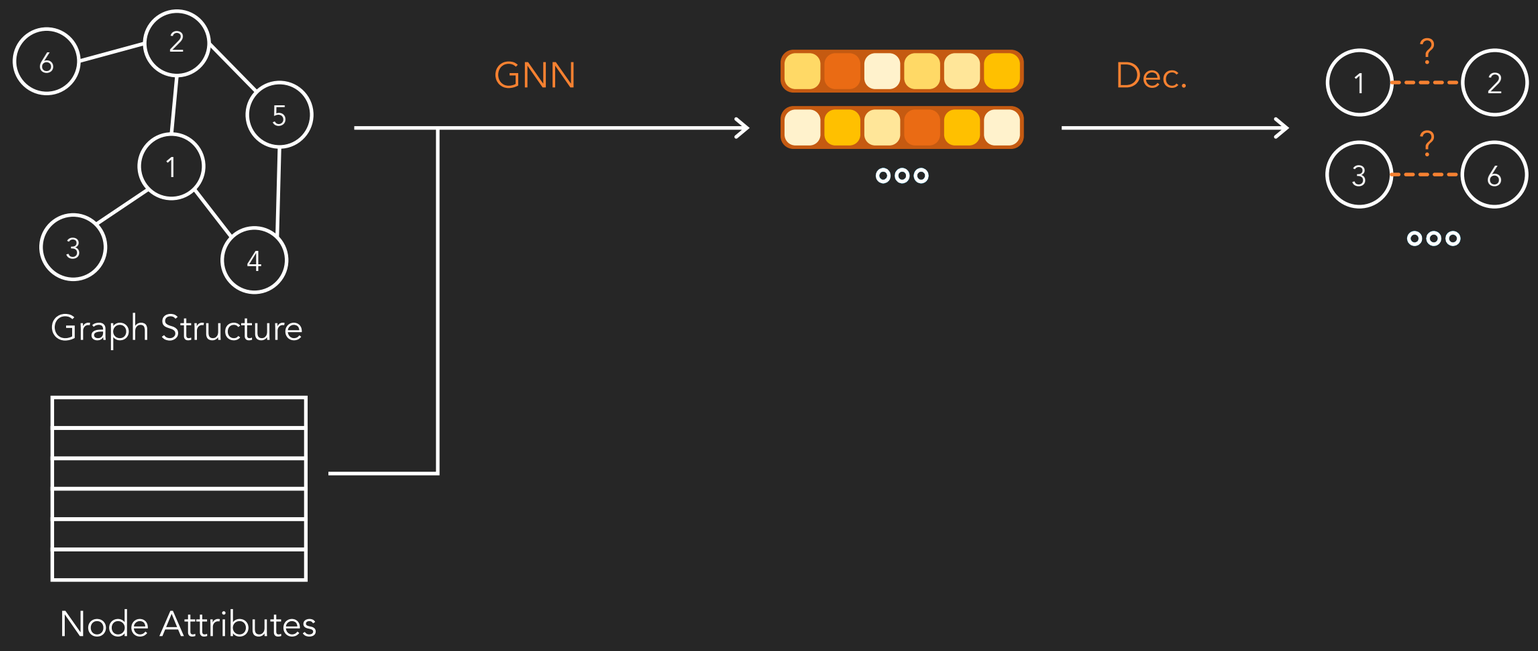
Datasets	SAGE	QSAGE	PSAGE	Neighbor Sample	GLNN+
Arxiv	489.49	433.90 (1.13x)	465.43 (1.05x)	91.03 (5.37x)	3.34 (146.55x)
Products	2071.30	1946.49 (1.06x)	2001.46 (1.04x)	107.71 (19.23x)	7.56 (273.98x)

Inference Time for 10 Randomly Chosen Nodes.

- **SAGE:** Base GNN model
- **QSAGE:** Quantized SAGE, FP32 to INT8
- **PSAGE:** Pruned SAGE, with 50% model parameters pruned
- **Neighbor Sampling:** sampling 15 nodes per layer

Graph-less Neural Networks: Teaching Old MLPs new Tricks with Distillation (Zhang et al, ICLR 2022)

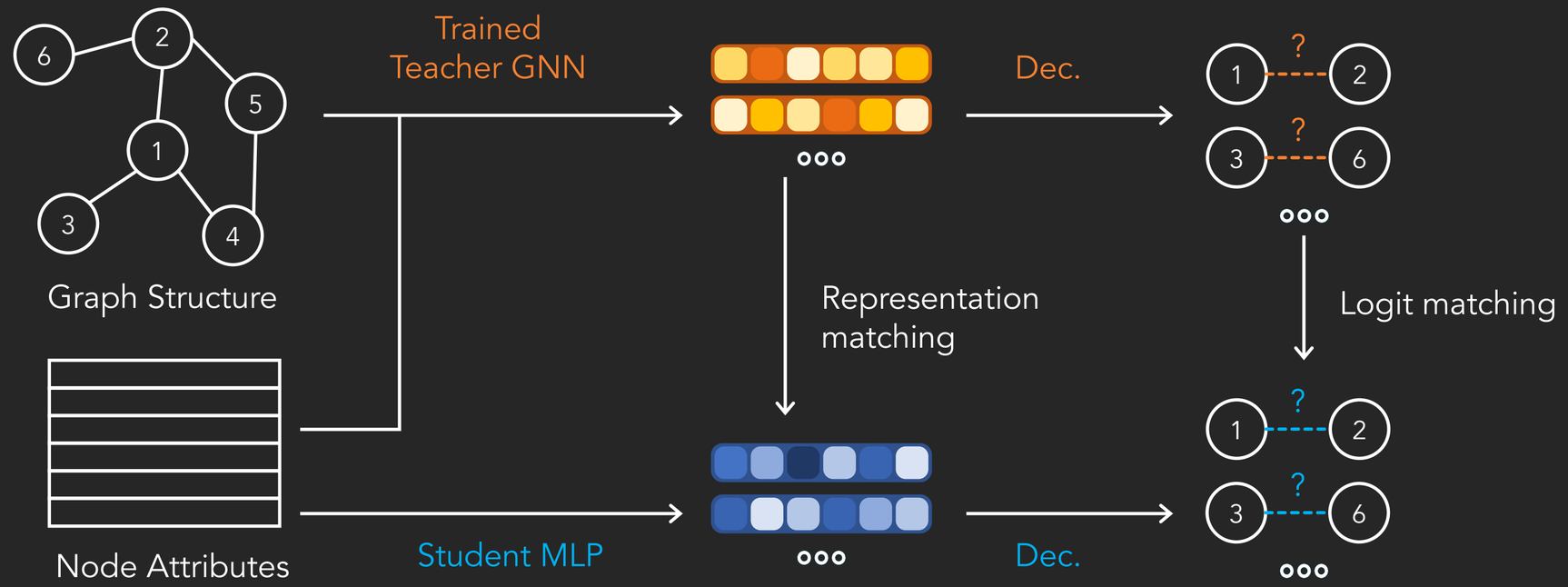
What About Link Prediction?



- GNN-based link prediction models usually involves two modules:
  - Encoder (GNN) to learn node representations
  - Decoder to make predictions on representations of node-pairs

Link-less Link Prediction via Relational Distillation (Guo et al, ICML 2023)

Knowledge Distillation for Link Prediction



- Two direct analogs of GLNN on link prediction

- Node representation-based matching ($\mathcal{L}_{RM}$)
- Predicted logit-based matching ($\mathcal{L}_{LM}$)

Link-less Link Prediction via Relational Distillation (Guo et al, ICML 2023)

Direct KD Methods Aren't Ideal for Link Prediction

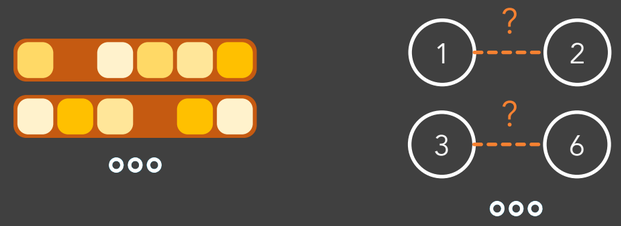
	GNN	MLP	$\mathcal{L}_{LM}$	$\mathcal{L}_{RM}$
Cora	27.80±2.11	22.90+2.22	22.65±2.51	22.24±0.55
Citeseer	38.78±2.59	31.21±3.75	29.35±2.55	26.23±1.08
Pubmed	52.71±1.81	38.01±1.67	39.03±4.21	43.27±3.12
CS	60.69±3.17	38.15±10.78	48.07 ±2.39	58.90±1.32
Physics	55.82±2.43	29.99±1.96	22.74±1.03	36.32±2.29
Computers	34.38±1.41	19.43±0.82	12.79±1.43	20.28±1.01
Photos	51.03±6.05	34.29±2.49	24.63±2.20	40.58±1.63

- Logit-based matching mostly performs worse than MLP.
- Representation-based matching outperforms MLP in some cases with small margins.

Link-less Link Prediction via Relational Distillation (Guo et al, ICML 2023)

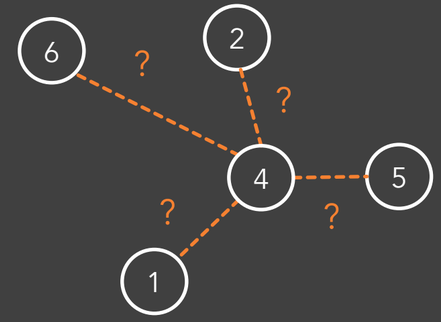
Relational Knowledge Matters for Link Prediction

- Transferring direct knowledge from teacher to student is inefficient in this setting



Direct knowledge on nodes / links

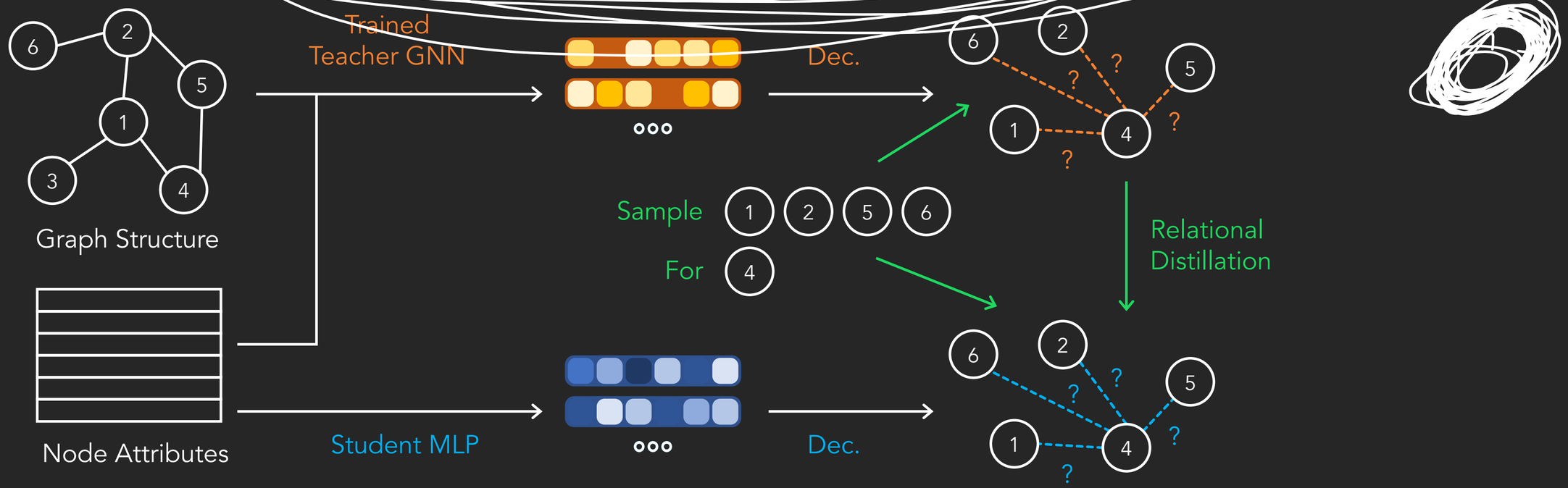
- Graph structure is critical for link prediction



Relational knowledge centered at one anchor node

Link-less Link Prediction via Relational Distillation (Guo et al, ICML 2023)

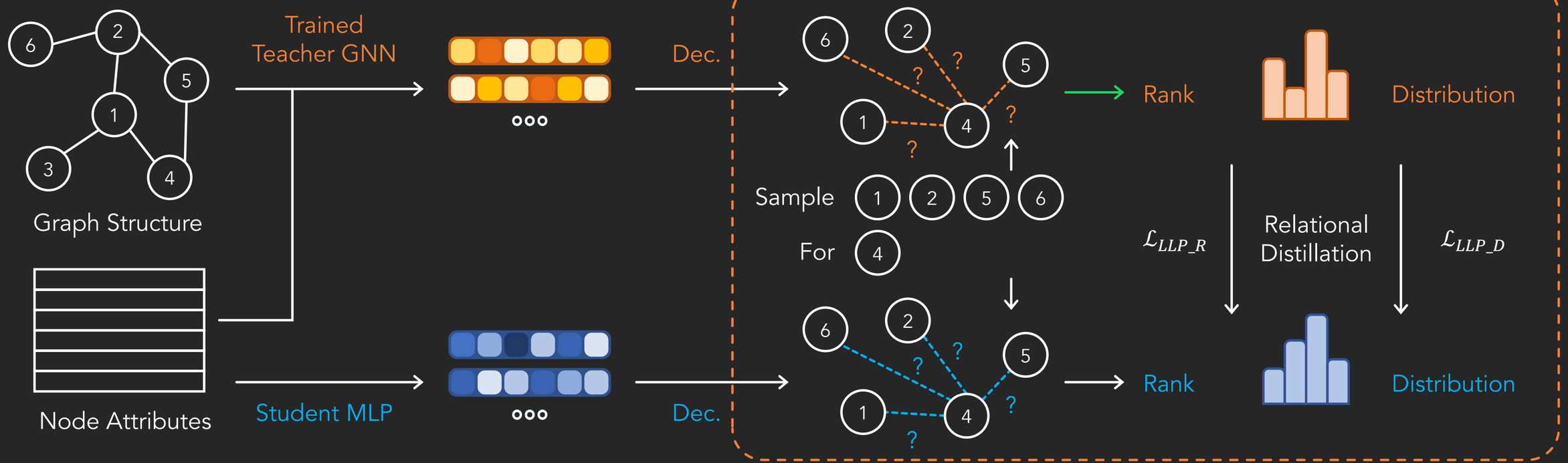
Link-less Link Prediction (LLP)



Idea: Node-anchored relational distillation.

Link-less Link Prediction via Relational Distillation (Guo et al, ICML 2023)

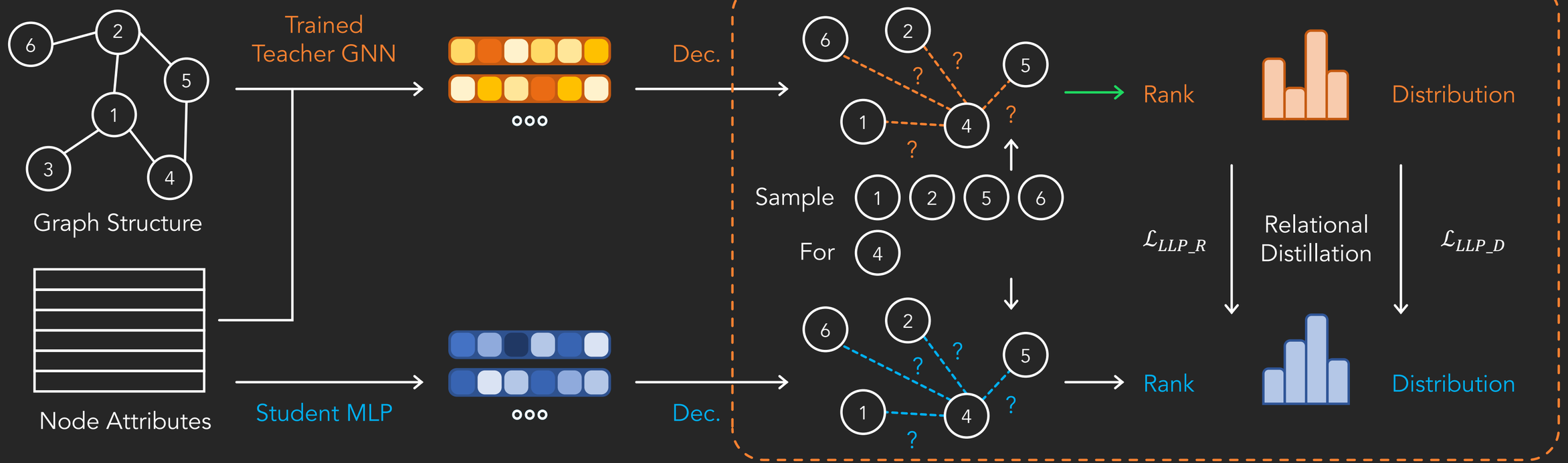
Link-less Link Prediction (LLP)



- Two novel relational distillation designs
  - Rank-based matching ($\mathcal{L}_{LLP\_R}$)
  - Distribution-based matching ($\mathcal{L}_{LLP\_D}$)

Link-less Link Prediction via Relational Distillation (Guo et al, ICML 2023)

LLP Training

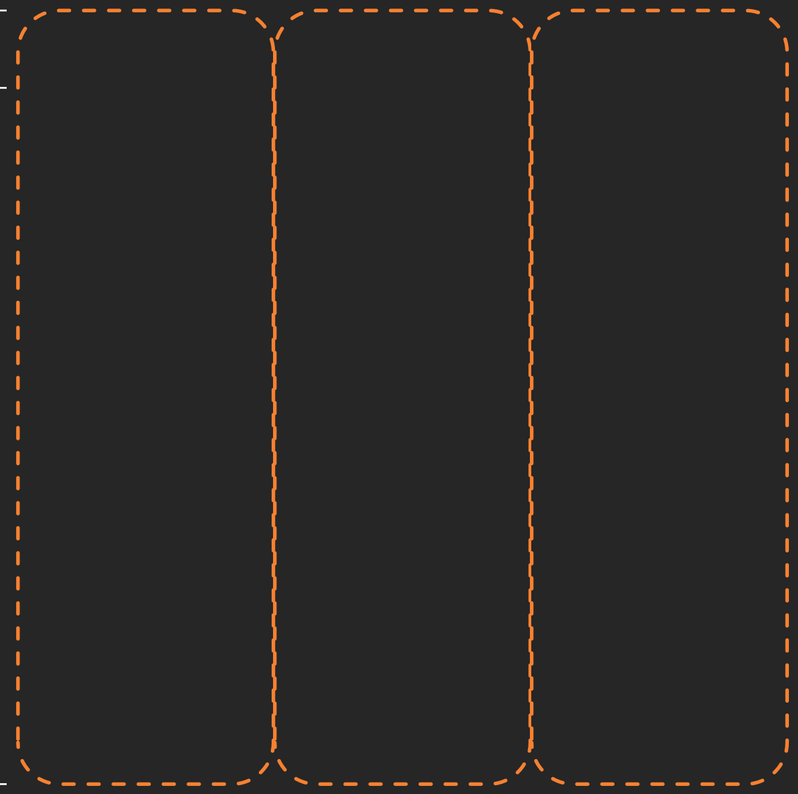


$$\mathcal{L} = \alpha \cdot \mathcal{L}_{sup} + \beta \cdot \mathcal{L}_{\text{LLP\_}R} + \gamma \cdot \mathcal{L}_{\text{LLP\_}D}$$

Link-less Link Prediction via Relational Distillation (Guo et al, ICML 2023)

Direct KD methods aren't ideal for Link Prediction

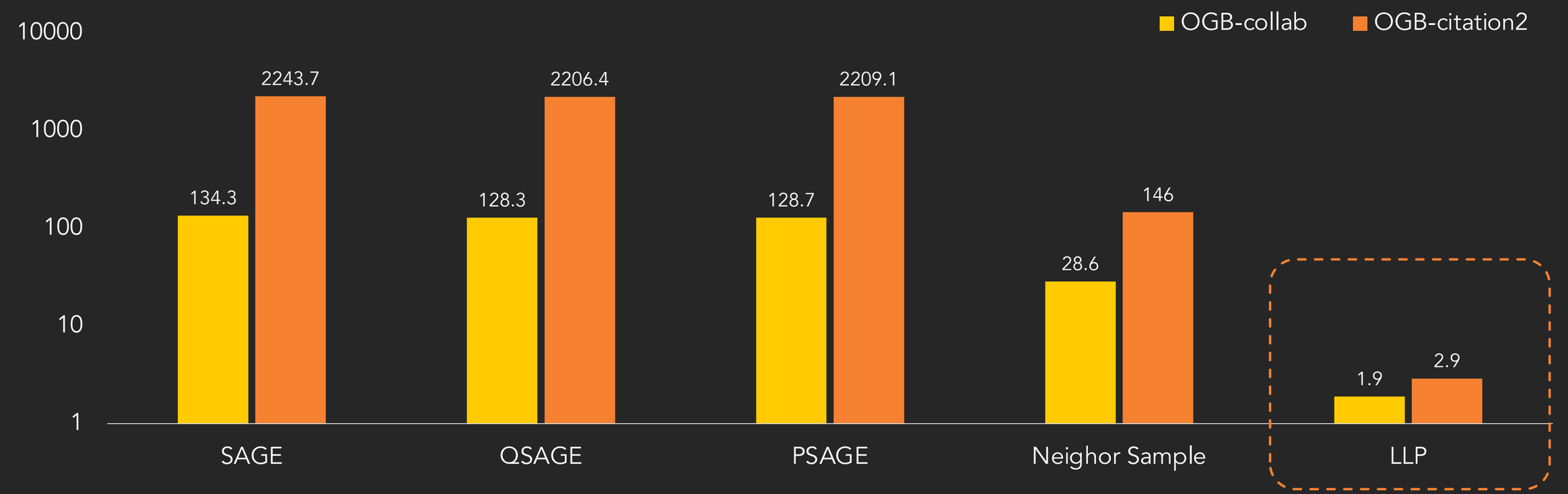
	GNN	MLP	$\mathcal{L}_{LM}$	$\mathcal{L}_{RM}$	LLP
Cora	74.38±1.54	**78.06**±1.50	74.72±4.27	75.75±1.51	**78.82**±1.74
Citeseer	73.89±0.95	71.21±3.22	72.44±1.52	65.19±5.54	**77.32**±2.42
Pubmed	51.98±5.25	42.89±1.67	42.78±3.15	44.44±2.40	**57.33**±2.42
CS	59.51±7.34	34.01±9.37	40.69±5.12	61.10±2.83	**68.62**±1.46
Physics	66.74±1.53	31.26±9.12	52.11±2.44	52.34±3.78	**72.01**±1.89
Computers	31.66±3.08	20.19±1.58	12.81±1.80	21.75±1.96	**35.32**±2.28
Photos	**51.50**±4.48	27.83±4.90	24.24±2.79	38.47±2.76	49.32±2.64
OGB-Collab	**48.69**±0.87	36.95±1.37	35.97±0.96	36.86±0.45	45.27±0.79
OGB-Citation2	**82.56**±0.04	40.63±0.00	38.42±0.01	42.50±0.01	53.20±1.20



- Significant improvements over direct methods
- Significant improvements over MLP
- Comparable or close performance to GNNs in many cases

Link-less Link Prediction via Relational Distillation (Guo et al, ICML 2023)

LLP is faster than other Inference Acc. methods



- **SAGE:** Base GNN model
- **QSAGE:** Quantized SAGE, FP32 to INT8
- **PSAGE:** Pruned SAGE, with 50% model parameters pruned
- **Neighbor Sampling:** sampling 15 nodes per layer

Link-less Link Prediction via Relational Distillation (Guo et al, ICML 2023)

LLP outperforms GNN on Cold-Start Link Prediction

	GNN	MLP	LLP	ΔMLP	ΔGNN
Cora	6.39	17.92	**22.01**	4.09	15.62
Citeseer	11.04	29.33	**32.09**	2.76	21.05
Pubmed	4.63	22.74	**37.68**	14.94	33.05
CS	9.46	29.09	**46.83**	17.74	37.37
Physics	5.46	20.22	**39.37**	19.15	33.91
Computers	1.53	10.72	**14.64**	3.92	13.11
Photos	0.87	20.44	**23.79**	3.35	22.92

Link prediction performance on newly appeared nodes without any edges.

Link-less Link Prediction via Relational Distillation (Guo et al, ICML 2023)

Limitations & Future Work

- How can we adapt LLP successfully to heterogeneous settings (KG completion, user-item recommendation)?

- How do we know whether we need a graph during inference?

- Are we making subgroup-specific tradeoffs in graph-less models?

MLPInit: Embarrassingly Simple GNN Training with MLP Initialization (Han et al, ICLR 2023)

Recent Research Themes

Overcoming model limitations

- Evaluation pitfalls in link prediction

  ACL'23, NeurIPS'23

- Boosting model expressiveness

  ICLR'22, NeurIPS'22, WSDM'23

- Understanding homophily

  ICLR'22, NeurIPS'23

- Self-supervised learning

  ICLR'22, ICLR'23, ICLR'23, KDD'23

- Learning under data sparsity

  AAAI'21, CIKM'22, ICML'22, NeurIPS'23

Overcoming scale limitations

- Faster real-time inference

  ICLR'22, ICML'23

- Faster training time & HPO

  ICLR'22, ICLR'23

Aligning with business problems

- Friend recommendation

  WWW'21, SIGIR'23

- Friend story ranking

  WSDM'22

- Forecasting growth metrics

  KDD'20

Takeaways

- GNNs are powerful, but suffer scalability challenges
- We can exploit cross-model connections to alleviate them
- Much room for future work in this space
- We're fortunate to work with brilliant collaborators

e-mail me at nshah@snap.com