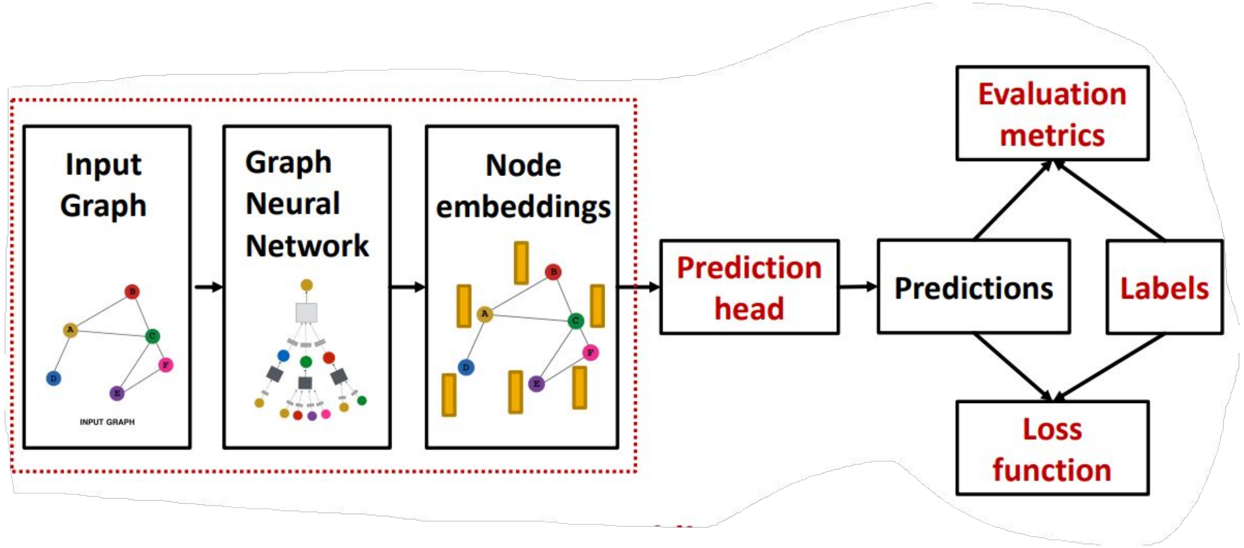
Sampling for Heterogeneous Graph Neural Networks

Ruoyu He, Feiyang Chen, YuanChing Lin, Yongqian Li

Outline

- Motivation & Challenges
- Proposed Methods
- Sampling Methods
- Experiments
- Conclusion & Future works

Motivation



Full-batch training generates embeddings of all the nodes at the same time: **not feasible for large-scale graphs**!

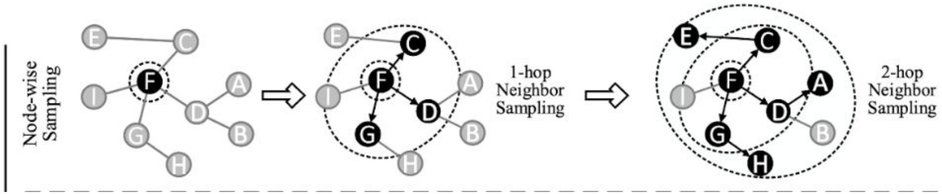
Large-scale graph: #nodes ranges from 10M to 10B. #edges ranges from 100M to 100B.

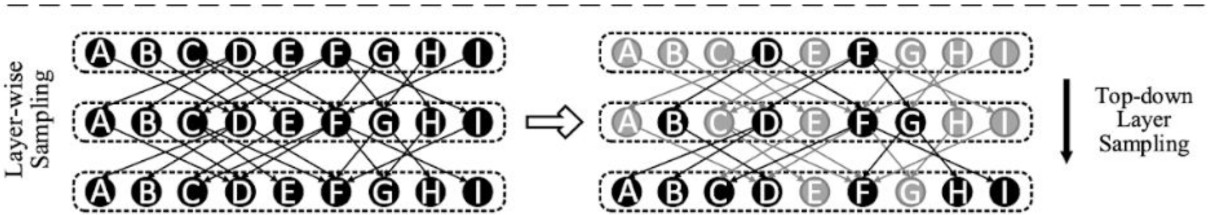
Solution: Train on a subgraph of the original graph that retains information as much as possible

Graph sampling: a set of sampling-based methods are developed during the past years.

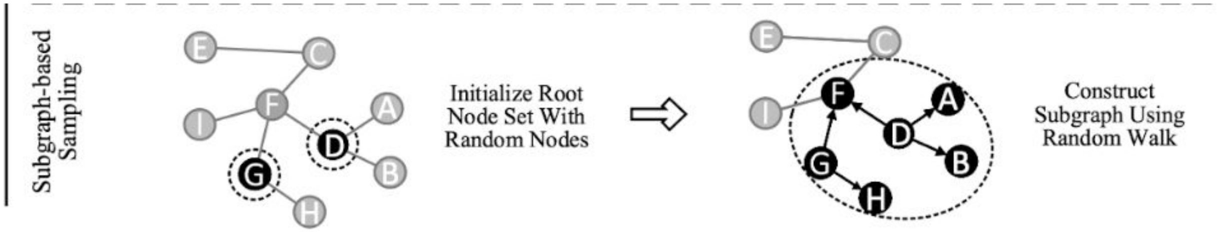
Sampling Methods

Node-wise: sampling a small set of neighbors of a single node in one sampling batch. **E.g. GraphSAGE**





Layer-wise: sampling multiple nodes simultaneously in the sampling operation in each layer. **E.g. FastGCN**



Subgraph-wise: sampling subgraphs, and for each batch, use one subgraph to do the training.

Challenge

Previous sampling methods focuses on **homogeneous graph**

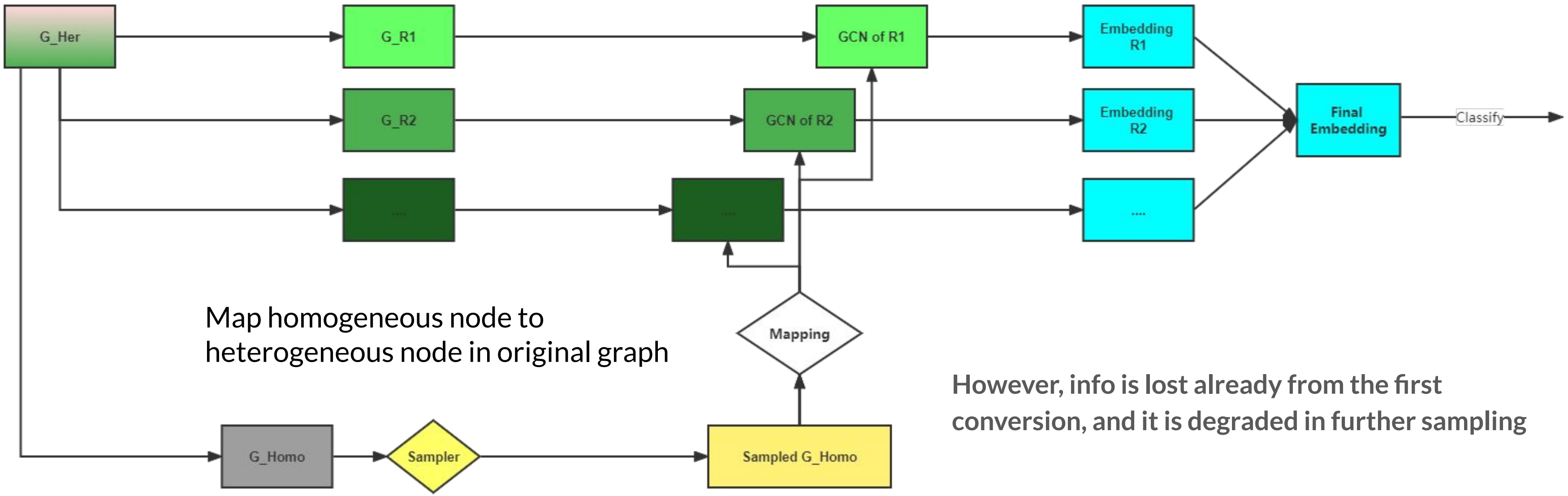
Only a few recent works have paid attention to sampling for **heterogeneous graphs** (HetGNN and HGSampling).

Multi-types relationship have **interdependence** between them, and a good neighborhood should retain those

Therefore, it is critical for a sampling method to distinguish different types of nodes and compute the effect.

Intuitive Idea

Converting the heterogeneous graph into a homogeneous and using the existing sampler.

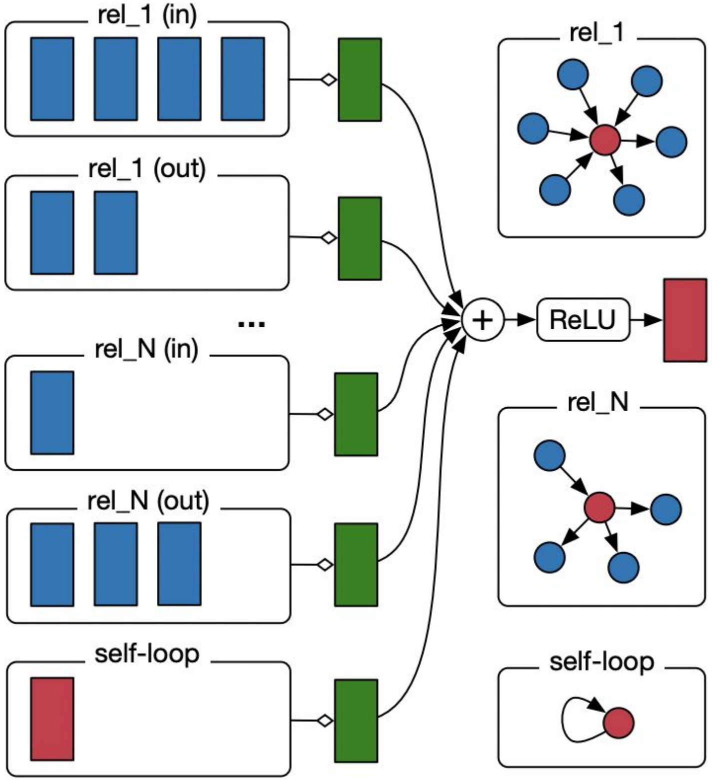


Map homogeneous node to heterogeneous node in original graph

However, info is lost already from the first conversion, and it is degraded in further sampling

RGCN

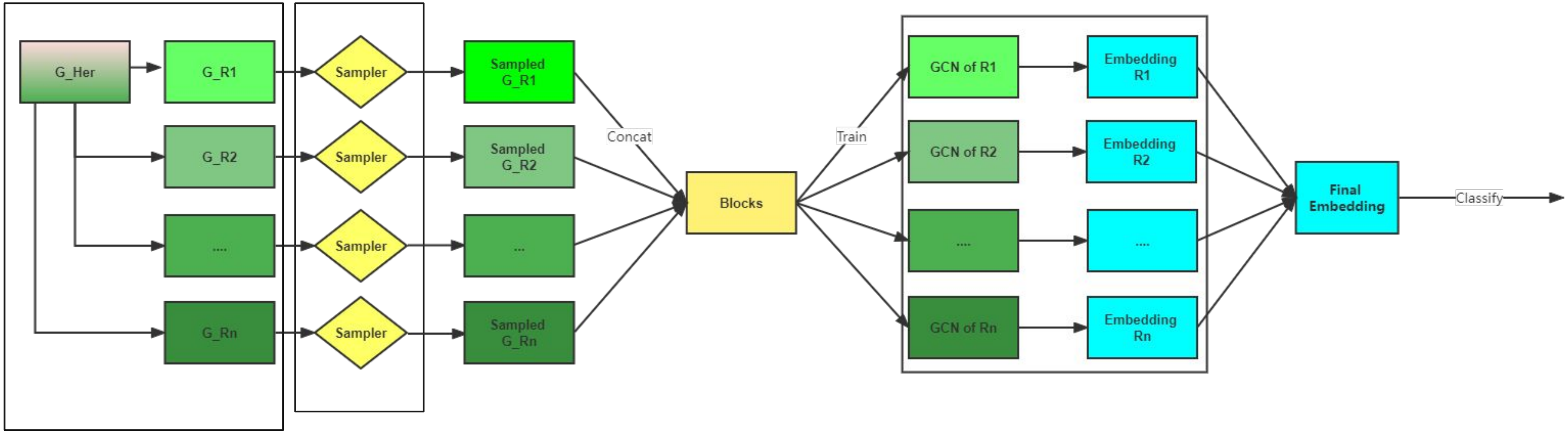
- Heterogeneous GNNs: aggregates neighborhood information differently for different relation types.
- RGCN is the most classic work about heterogeneous GNNs



Use different neural network weights for different relation types → relation-specific transformations

Proposed Method

Heterogeneous GNNs (e.g. RGCN)



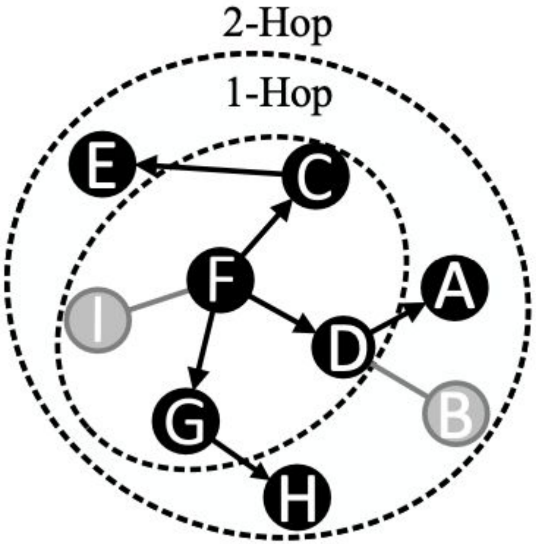
Sampling Methods

relation-specific transformations

Node-wise Sampling

Follow **GraphSAGE** sampler, for each node in the training graph, samples k-hop neighbors by search depth.

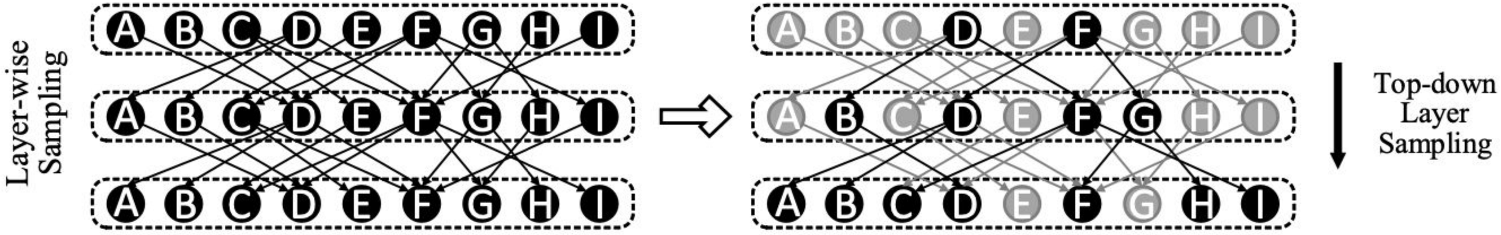
However, these methods sample neighbors for each node recursively, which leads to the exponential neighborhood extension.



Layer-wise Sampling

Follow **FastGCN**'s setting, sampling a certain number of nodes in each layer independently based on the pre-set probability distribution.

To reduce the variance, using importance sampling technique to alter the probability distribution.



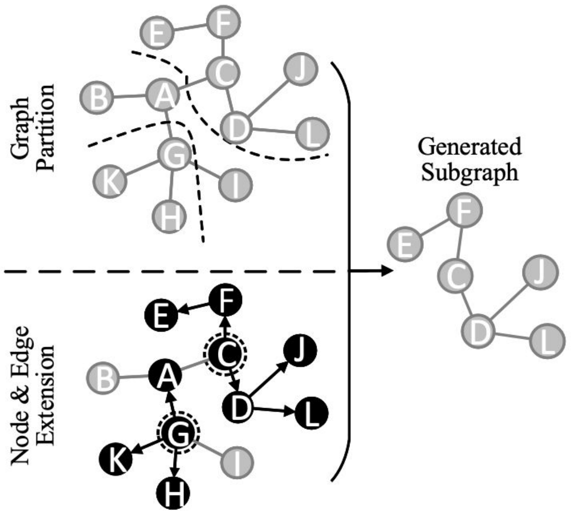
Subgraph-wise Sampling

Cluster-GCN: first partitions the original graph into multiple clusters by using graph clustering algorithms.

Randomly samples a fixed number of clusters as a batch and forms a subgraph by combining the chosen clusters.

ShaDow-GNN: starting from the target node, the sampler traverses up to k hops.

Then the subgraph is induced from all the nodes selected by the sampler.



Here, we use our first pipeline since the representations of sampled subgraphs have different dimensions.

Experiments

Dataset	AIFB	MUTAG	BGS	AM
Entities	8,285	23,644	333,845	1,666,764
Relation Types	45	23	103	133
Edges	29,043	74,227	916,199	5,988,321
Labeled	176	340	146	1,000
Entity Types	7	5	27	7
Classes	4	2	2	11

Node classification task for heterogeneous graphs.

We evaluate our sampling methods on four heterogeneous datasets in Resource Description Framework (RDF) format: AIFB, MUTAG, BGS, and AM.

Our experiments are all base on Deep Graph Library (DGL), which is an easy-to-use, high performance and scalable Python package for deep learning on graphs.

```
peak memory: 15.775 GB
memory: 15.775 GB
_hetero.cc:87: Partition a graph with 1885136 nodes and 6080376 edges into 9 parts and get 342701 edge cuts
```

Evaluation of different sampling methods

Datasets	Sample Methods	Acc (%)	Total Time (s)	Batch Time (s)
AIFB	Original R-GCN	95.83	39.1	0.52
	Layer-wise sampling	**97.22**	8.97	0.22
	Node-wise sampling	91.67	18.57	0.21
	Subgraph sampling (ShaDowK)	47.22	5.06	0.25
	Subgraph sampling (Cluster-GCN)	58.33	19.23	0.32
MUTAG	Original R-GCN	73.23	28.98	0.43
	Layer-wise sampling	**75.00**	6.99	0.11
	Node-wise sampling	69.12	65.26	0.11
	Subgraph sampling (ShaDowK)	66.18	2.23	0.22
	Subgraph sampling (Cluster-GCN)	67.65	6.76	0.11
BGS	Original R-GCN	83.10	36.58	0.35
	Layer-wise sampling	**96.55**	10.18	0.25
	Node-wise sampling	89.66	312.08	0.23
	Subgraph sampling (ShaDowK)	65.52	2.32	0.23
	Subgraph sampling (Cluster-GCN)	65.52	8.10	0.24
AM	Original R-GCN	89.29	149.4	2.31
	Layer-wise sampling	86.36	80.78	0.32
	Node-wise sampling	52.02	2423.15	0.25
	Subgraph sampling (ShaDowK)	23.23	17.31	0.13
	Subgraph sampling (Cluster-GCN)	53.03	56.22	0.21

1. Layer achieved best accuracy and a considerable speedup

2. The poor performance of subgraph may be because the original relationship is broken to some extent, thus losing more information compared with node/layer sampling

exponential neighborhood extension!

Evaluation of different sampling numbers

Datasets	Sample Methods	Sample Nums	Acc (%)	Total Time (s)	Batch Time (s)
AIFB	Layer-wise sampling	4	91.67	8.67	0.21
		8	94.44	8.34	0.20
		16	97.22	8.97	0.22
	Node-wise sampling	4	94.44	18.89	0.21
		8	94.44	19.39	0.22
		16	97.22	19.06	0.21
	ShadowK sampling	4	22.22	5.07	0.25
		8	38.89	5.07	0.25
		16	47.22	5.06	0.25
	Cluster-GCN pipeline	4	58.33	18.13	0.32
		8	58.33	19.23	0.32
		16	50.00	27.04	0.31
MUTAG	Layer-wise sampling	4	67.65	6.76	0.11
		8	72.06	6.89	0.11
		16	75.00	6.99	0.11
	Node-wise sampling	4	45.59	67.40	0.12
		8	67.65	67.08	0.11
		16	64.71	66.20	0.11
	ShadowK sampling	4	54.41	2.18	0.22
		8	33.82	2.20	0.22
		16	66.18	2.23	0.22
	Cluster-GCN pipeline	4	67.65	4.48	0.12
		8	66.18	3.89	0.12
		16	67.65	4.35	0.12
BGS	Layer-wise sampling	4	89.66	9.75	0.24
		8	93.10	9.70	0.24
		16	96.55	10.18	0.25
	Node-wise sampling	4	96.55	321.63	0.25
		8	93.10	313.62	0.24
		16	89.66	322.96	0.24
	ShadowK sampling	4	62.07	2.28	0.23
		8	44.83	2.32	0.23
		16	65.52	2.32	0.23
	Cluster-GCN pipeline	4	58.62	6.54	0.25
		8	58.62	6.95	0.24
		16	65.52	8.10	0.24
AM	Layer-wise sampling	4	74.24	59.18	0.32
		8	80.81	79.57	0.32
		16	86.36	80.78	0.32
	Node-wise sampling	4	52.02	2423.15	0.26
		8	43.43	2440.09	0.25
		16	53.54	2402.25	0.26
	ShadowK sampling	4	23.23	17.31	0.13
		8	3.03	19.26	0.14
		16	0.51	19.6	0.14
	Cluster-GCN pipeline	4	45.45	83.09	0.22
		8	47.47	99.96	0.22
		16	46.46	58.95	0.23

Evaluation of scalability on larger-scale heterogeneous graphs

OGBN-MAG, including four types of entities—papers (736,389 nodes), authors (1,134,649 nodes), institutions (8,740 nodes), and fields of study (59,965 nodes)

Datasets	Sample Methods	Acc (%)	Total Time (s)	Batch Time (s)
OGBN-MAG	Original R-GCN	39.77	-	-
	Cluster R-GCN	37.32	-	-
	NARS	52.09	26241	26.2
	Layer-sampling R-GCN	46.86	800	8.61

We achieves comparable performance to the current state-of-the-art method NARS, but takes less time.
Our method achieves good efficiency without compromising on the effectiveness on heterogeneous graphs.

Contributions

- We propose 2 general pipelines of sampling for heterogeneous GNNs.
- We are the first to provide a thorough discussion about how each type of sampling methods work on heterogeneous graphs.
- Scalable experimental results also show we achieve the trade-off between efficiency and effectiveness.

Future works

- How to concat subgraph sampler?
- How to address imbalanced neighbors' numbers in different types?
- Improve the original sampling method by merging the characteristics of several sampling methods in different categories

Thank you!