Can graph sparsification accelerate GNN workloads?

Vasiliki Kalavri
vkalavri@bu.edu

in collaboration with: Yuhang Song, Naima Abrar Shami, Romaric Duvignau
sites.bu.edu/casp

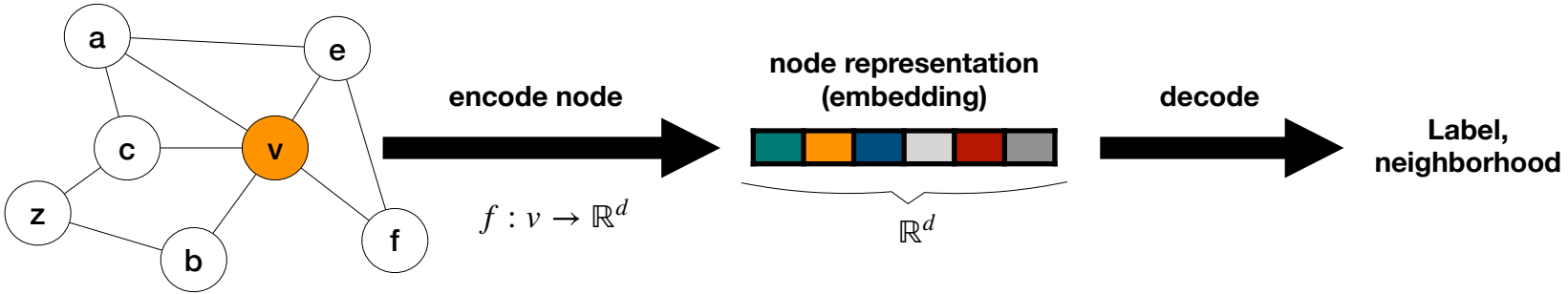
21st TUC Meeting
Boston, Sept 4 2026

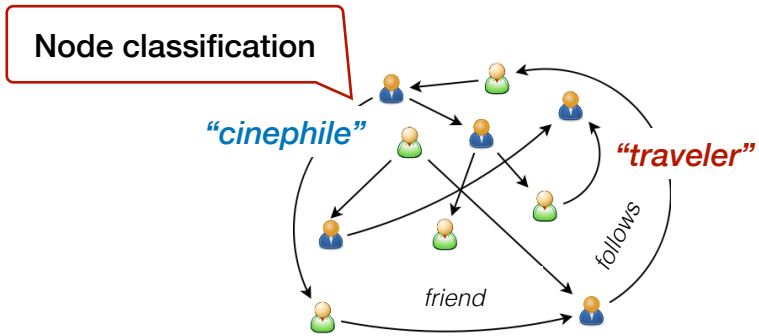
BOSTON UNIVERSITY

CASP Systems

NSF

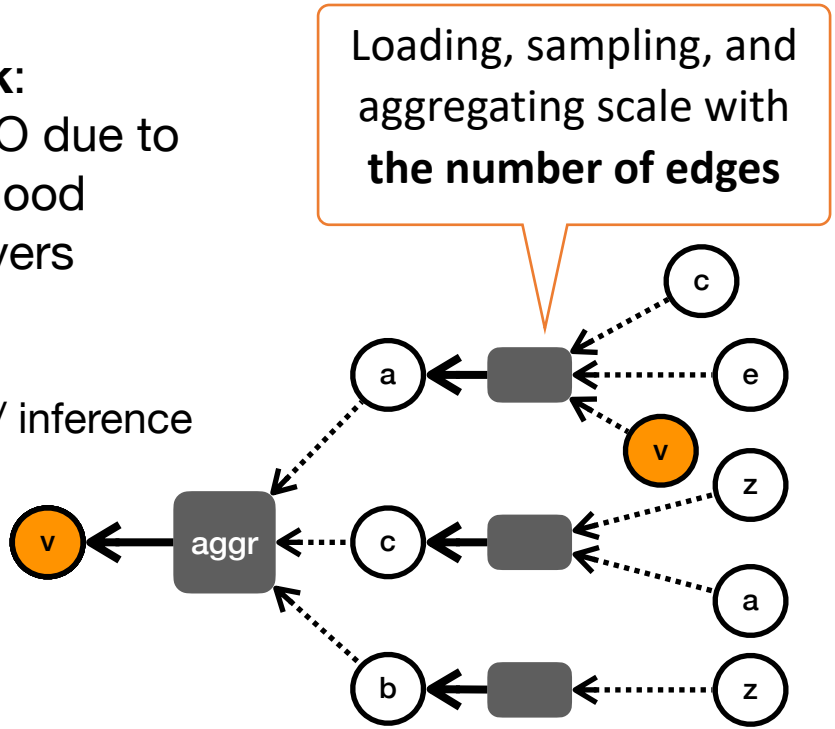
Graph neural networks (**GNNs**) learn to encode a graph's structure and characteristics and generate **embeddings**: a mapping of the graph representation to a low-dimensional space.





Scaling GNN pipelines is a data management problem

- **Data movement is the bottleneck**: irregular memory accesses, high I/O due to frequent feature access, neighborhood explosion when traversing GNN layers
- **Solutions** (so far):
  - Distributed and multi-GPU training / inference
  - Out-of-core storage
  - Specialized data structures
  - Indexing and prefetching
  - Algorithmic modifications

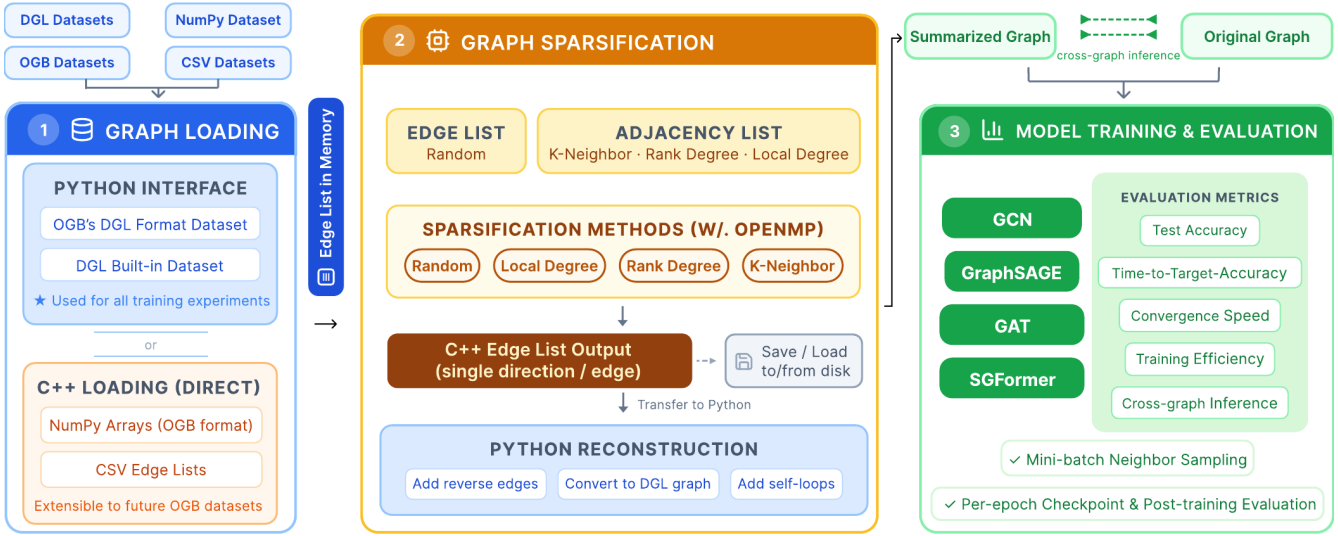


How much of the graph structure is actually necessary for effective learning?

- Real-world graphs are **noisy**, **redundant**, and often exhibit **heavy-tailed** degree distributions

- Can we apply **graph sparsification** as a lightweight preprocessing step?
  - Compressing the graph's structure before learning can reduce memory, I/O, and sampling overhead
  - Benefits expand to every subsequent training and inference run, with no change to the model or the pipeline.

BONSAI: an extensible experimental framework to study how sparsification interacts with GNNs



C++ sparsifiers

High-performance, parallel implementations.

Exact time-to-accuracy

Per-epoch checkpoints, post-hoc evaluation.

Billion-edge scale

GraphBolt streaming pipeline with memory-mapped features.

Extensible

New sparsifiers and models can be easily added.

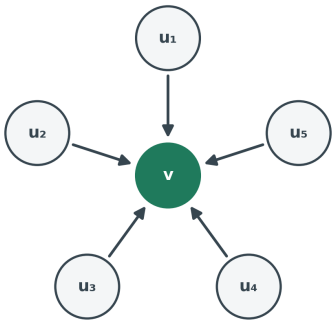
Experimental setting: datasets

Dataset	Nodes	Edges	Feature Dim.	File Size	Avg. deg.	Max deg.	Deg. Gini	Edge Homophily
PubMed	19.7K	44.3K	500	54.5 MB	4.5	171	0.60	0.80
CoauthorCS	18.3K	81.9K	6805	477.8 MB	8.9	136	0.46	0.81
Roman-empire	22.7K	32.9K	300	19.5 MB	2.9	14	0.18	0.05
Arxiv	169.3K	1.2M	128	79.2 MB	13.7	13,161	0.63	0.65
Products	2.4M	61.9M	100	1.4 GB	50.5	17,481	0.62	0.81
Papers100M	111.1M	1.6B	128	56.2 GB	29.1	251,471	0.67	0.57

Degree Gini: measures the skew of the degree distribution
Edge homophily: the fraction of edges whose endpoints share a label

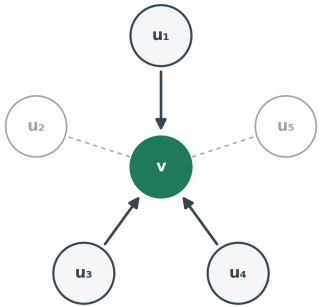
Experimental setting: GNN models

GCN



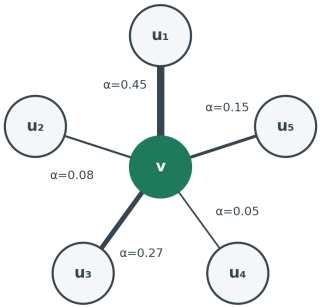
Spectral convolution, traditional GNN

GraphSAGE



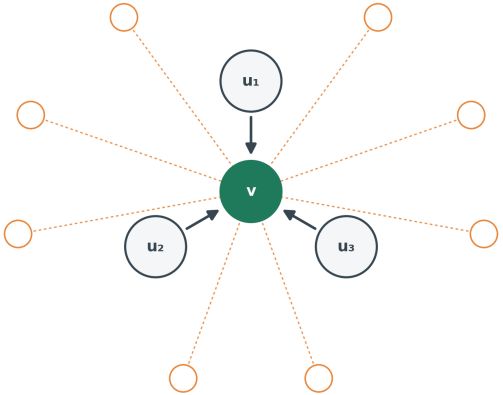
Inductive, sampling-based method

GAT



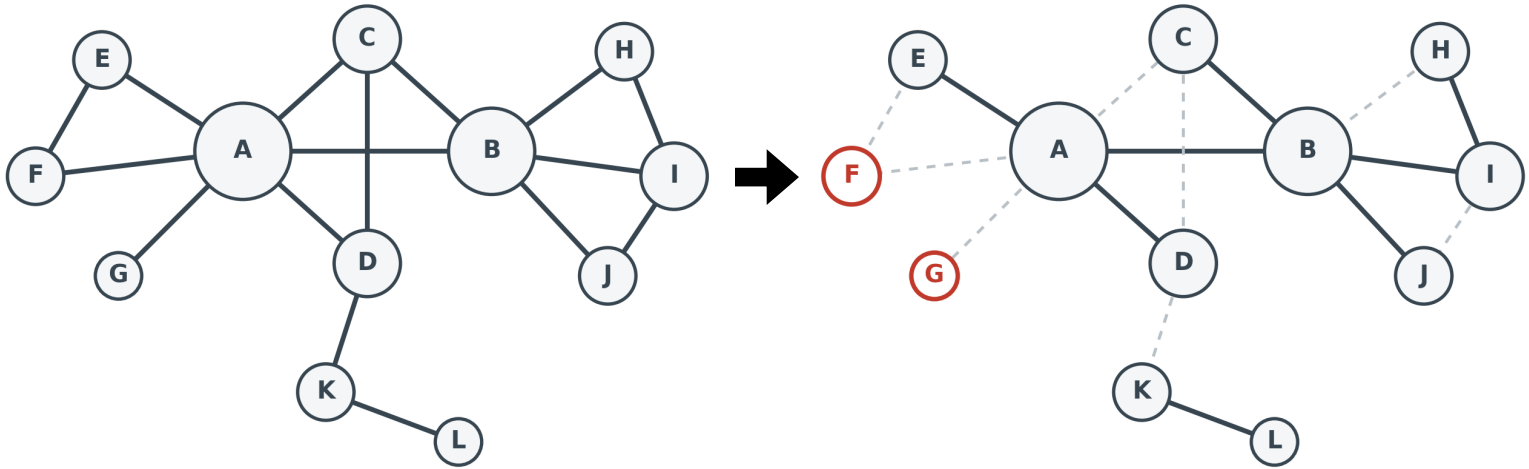
Attention-based message passing

SGFormer



Graph transformer combining a GNN branch with global attention

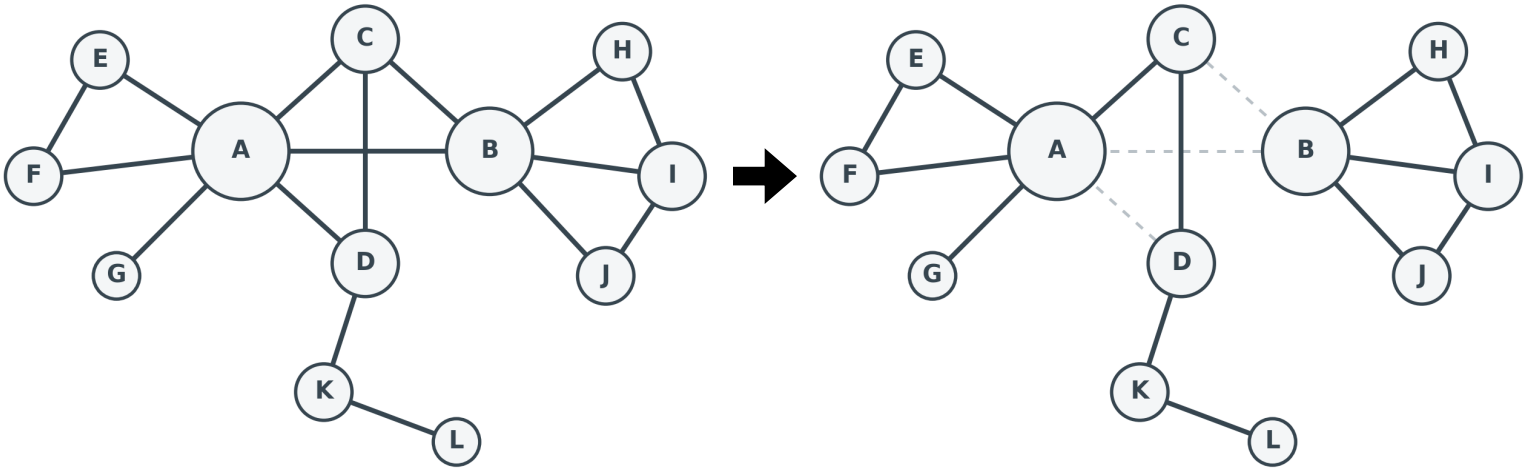
Experimental setting: Random Sparsification



Preserve each edge with probability p (*0.5 in this example*)

Degree distribution keeps its shape, scaled by p.
Low-degree nodes can lose all edges (e.g., *F and G*)

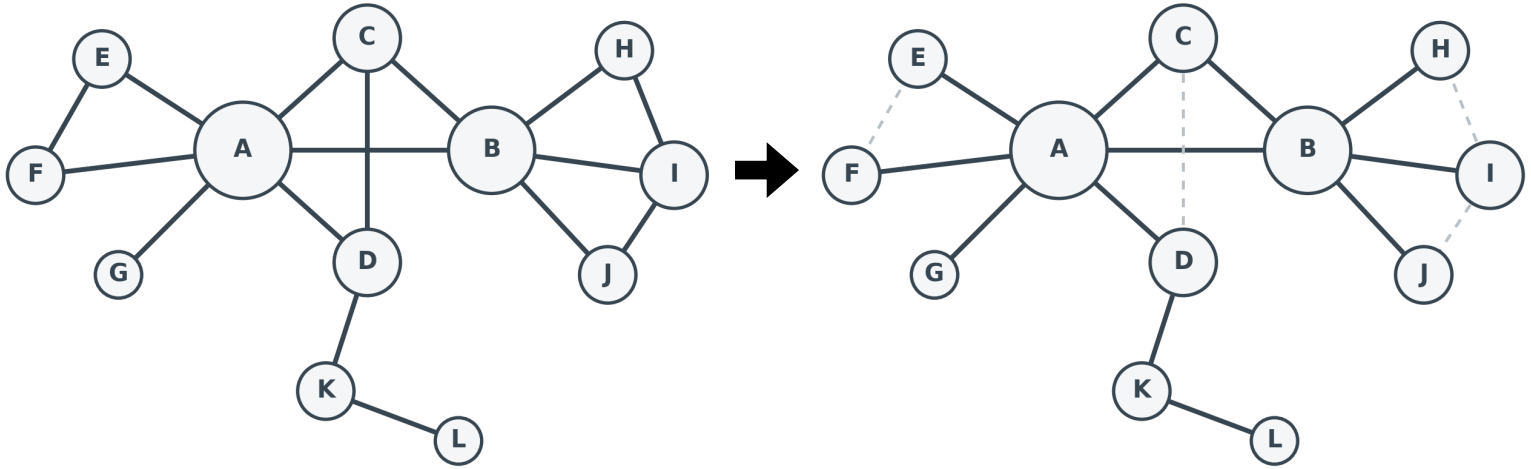
Experimental setting: K-Neighbor Sparsification



Each node picks up to k neighbors uniformly at random *(here k = 2)*.
An edge is preserved if either endpoint selects it.

Avoids isolated nodes and flattens the tail (caps degree at ~k).

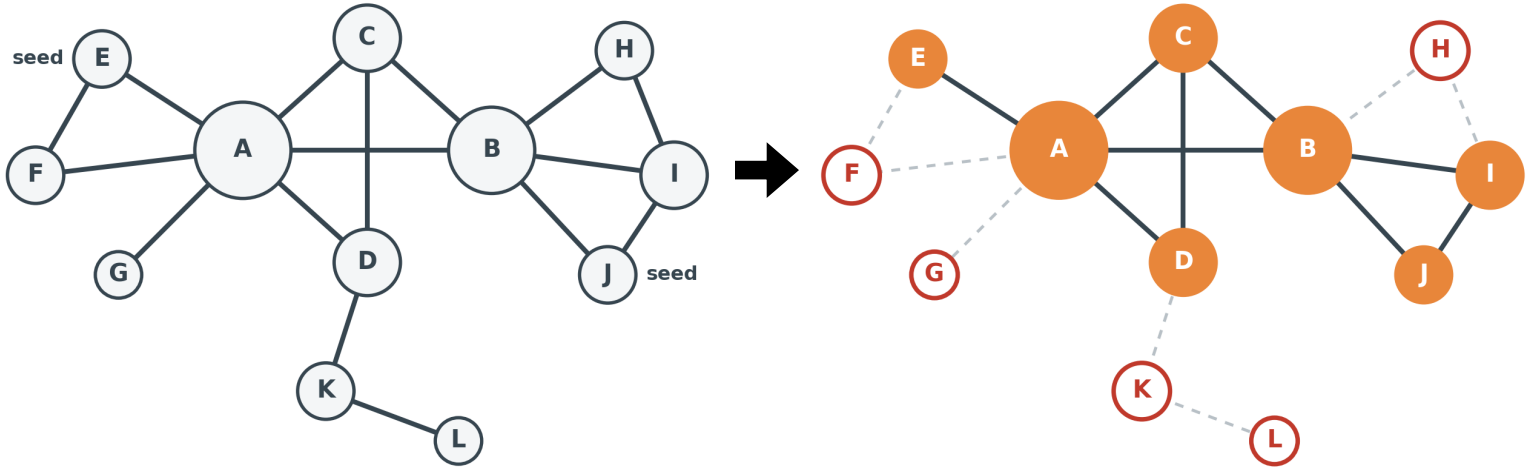
Experimental setting: Local Degree Sparsification



Each node keeps edges to its d_i^{α} *highest-degree* neighbors (*here α = 0.5: A and B keep 2, everyone else keeps 1*)

Hub-biased; preserves high-degree backbone

Experimental setting: Rank Degree Sparsification



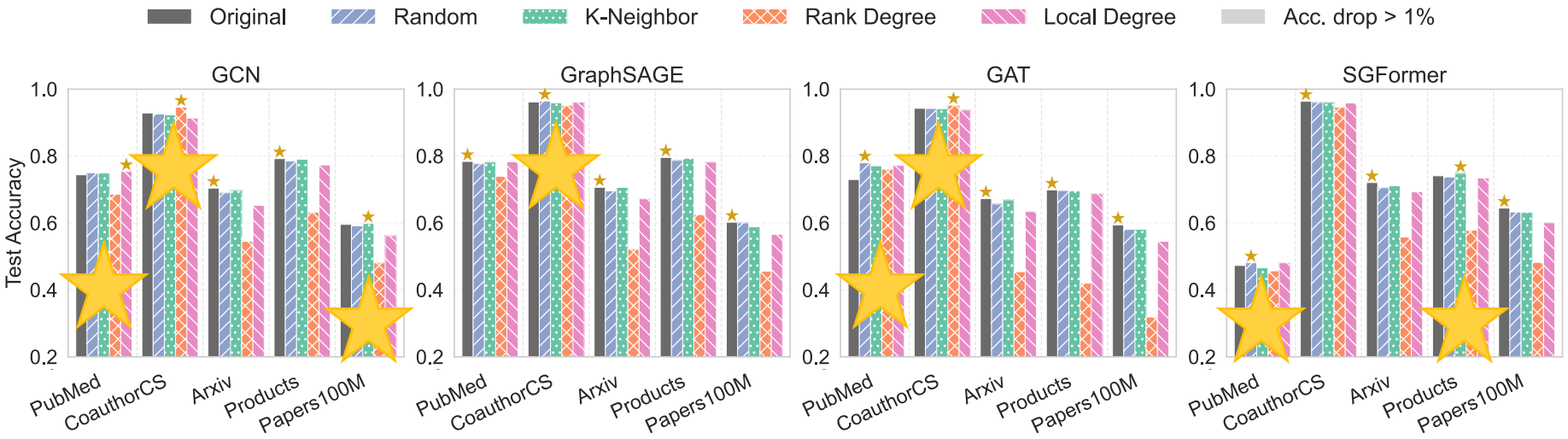
BFS from seed nodes, expanding to top-ρ highest-degree neighbors, keeping only edges of selected nodes

Isolates non-selected nodes; keeps a small core of hubs

Evaluation axes

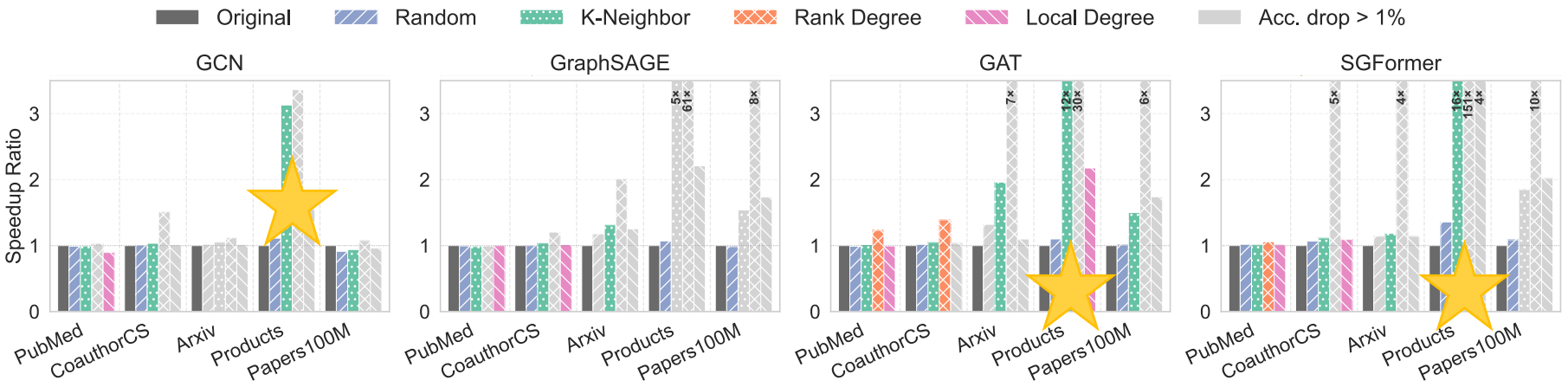
- **Accuracy and time to convergence**: How does graph sparsification affect the *best accuracy* a GNN model can achieve, and the time needed to reach it?
- **Training efficiency**: Does sparsification reduce the time to reach a target accuracy?
- **Serving-time trade-offs**: Given a model *trained on the original graph*, can we accelerate *serving using the sparsified graph* while preserving accuracy?
- **Pre-processing overhead**: What is the upfront cost of sparsification and how does it compare to downstream savings?
- **Compression vs. performance**: How do reductions in graph size translate into runtime gains?

Graph sparsification can **preserve**, and even **improve**, predictive performance



Across all combinations, there exists at least one instance of a model trained on a sparsified graph that exceeds the accuracy of the same model trained on the original graph

Cross-graph inference can be both **faster** and **highly-accurate**



K-Neighbor achieves significant inference speedups (e.g., 12x-16x) on large graphs, with at most 1-2% accuracy loss.

Other interesting findings

- **K-Neighbor** is the most robust sparsification method, while Rank Degree is the least effective one
- Sparsification delivers significant speedups on large but not small datasets, and **can occasionally prolong training**
- Sparsification cost is often **amortized in a single training run**

Why do sparsifiers behave differently?

- Two important properties:
    - whether sparsification **isolates nodes** (drops all edges)
    - whether sparsification **preserves homophily**
- In practice, sparsification is worthwhile on large, heavy-tailed graphs, where both achievable compression and speedup are significant.

Next steps

- Validating our hypotheses across more GNN architectures and datasets
- Explore summarization methods that also consider nodes and features, e.g., metric backbone (VLDB'16).

Can graph sparsification accelerate GNN workloads?

Vasiliki Kalavri
vkalavri@bu.edu

in collaboration with: Yuhang Song, Naima Abrar Shami, Romaric Duvignau
sites.bu.edu/casp

21st TUC Meeting
Boston, Sept 4 2026

BOSTON UNIVERSITY

CASP Systems

NSF