SGS-GNN: A Supervised Graph Sparsifier for Graph Neural Networks

Siddhartha Shankar Das‡, Naheed Anjum Arafat†, Muftiqur Rahman*, SM Ferdous‡, Alex Pothen ‡, Mahantesh Halappanavar‡, Danda B. Rawat†
Pacific Northwest National Laboratory (PNNL)‡, Howard University†, Islamic University of Technology*, Purdue University‡,
Contact: siddhartha.das@pnnl.gov

KDD 2026 August 9–13, 2026 Jeju, Korea

arXiv Code

Why SGS-GNN?

GNNs typically process all graph edges, increasing computation and GPU memory while propagating information through noisy or task-irrelevant connections. Existing supervised sparsifiers often lack precise global sparsity control, retain costly edge-level computation graphs, and overlook graph homophily and heterophily.

SGS-GNN addresses these limitations in the following manner:

Precise Sparsity Control: Explicitly samples a subgraph that strictly satisfies the target global sparsity ratio.

Scalable Learning: Backpropagates only through sampled edges and uses gradient checkpointing for efficient training.

Reduced GPU Memory: Avoids retaining edge-level computation graphs and gradients for pruned edges.

Reliable Performance: Updates the edge scorer only when it outperforms a strong degree-based sampler, preventing degradation below the baseline.

Case Study: Heterophilic Graph

The following is a two half-moon heterophilic graph, with edge homophily of 0.32 meaning only 32% of edges connect vertices of the same class.

A GNN using the full graph is only about 81% accurate. Compared to others only SGS-GNN can find sparse-subgraphs with 100% edge homophily to yield 100% accuracy.

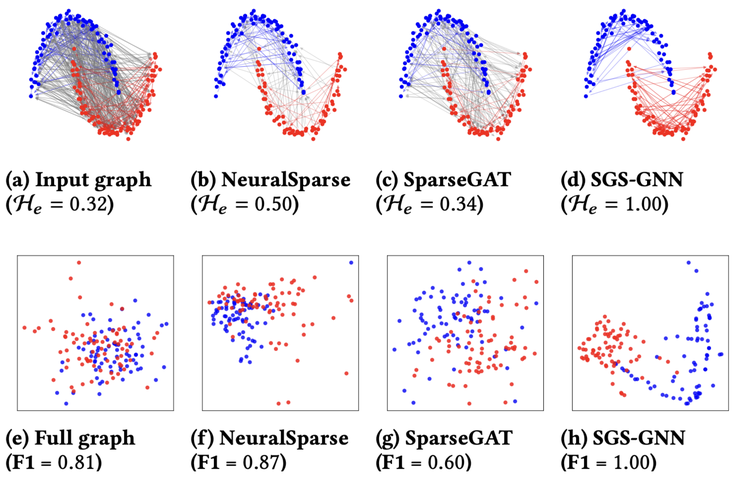

Fig. 1 Comparison of the full graph, NeuralSparse, SparseGAT, and SGS-GNN on a heterophilic graph ($|\mathcal{G}|$ = 150, $|\mathcal{E}|$ = 870, Train = 3%). The top row shows the input graph and the sampled sparse subgraphs along with edge homophily ($\mathcal{H}_e$). The bottom row shows the corresponding node embeddings and F1 scores. SGS-GNN better preserves task-relevant structure in the sparse subgraph, yielding perfect prediction performance.

Proposed Method: SGS-GNN

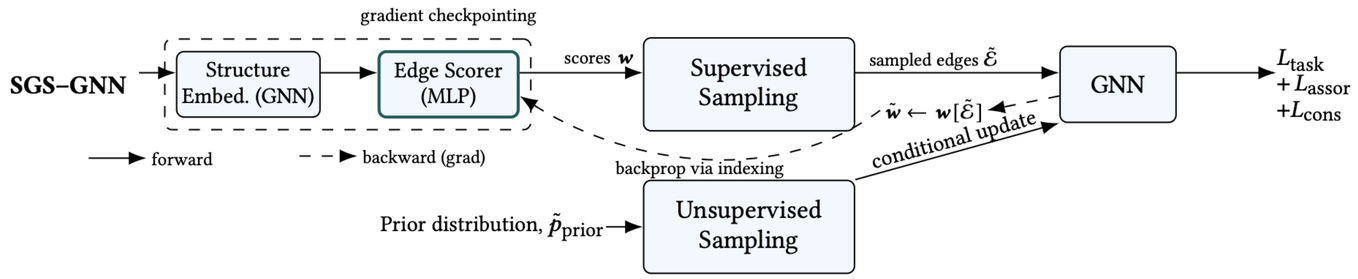

Fig. 2 SGS-GNN starts by computing edge sampling probabilities, then the sampler selects sparse subgraphs, and the downstream GNN operates on the sampled edges. To reduce high memory cost of edge-scorer, SGS-GNN applies gradient checkpointing and updates only if supervised sampling is beneficial.

Training Steps

1. Start with a strong unsupervised prior: A degree-based sampler creates an initial sparse graph that preserves important connections.
For example, edge sampling proability is $\mathrm{P}_{\text{prior}}(\mathrm{u},\mathrm{v}) \propto (1/d_u + 1/d_v)$, another option effective resistance $\mathrm{P}_{\text{prior}}(\mathrm{u},\mathrm{v}) \propto \boldsymbol{x}^{\mathrm{T}}\mathrm{L}^{+}\boldsymbol{x}$.

2. Score the original Edges: A neighborhood-aware encoder (GNN with sampled edges and Edge Scorer) estimates how useful each edge is for the prediction task.

3. Sample Edges within Budget: SGS-GNN samples exactly the user-specified percentage of edges using both learned scores and the degree prior.

4. Train on the Sampled Sparse Graph: The downstream GNN learns using only the selected edges, reducing message-passing and memory costs.

5. Update Only When Helpful: The edge scorer is updated only when its sampled graph outperforms the degree-based baseline.

Which Edges Are Removed?

The SGS-GNN prefers to drop following types of edges from the graphs.

(I) $\mathbf{P_{prior}}$: Unsupervised prior prefers bridge edges and drops non-bridge edges
(II) $\mathbf{L_{CE}}$: Remove task-irrelevant edges by down-weighting them.
(III) $\mathbf{L_{assor}}$: Discourages heterophilic labeled edges through this regularizers.
(IV) $\mathbf{L_{cons}}$: Discourages edges where inconsistency between edge-weight and embedding-distance presents.

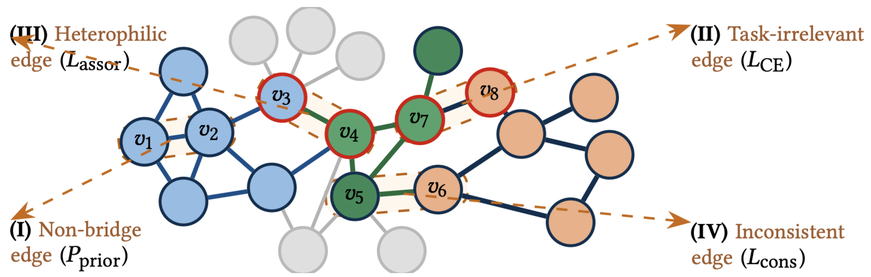

Fig. 3. Four inductive-bias cases guiding edge removal (training nodes have red borders)

Key Results

33 graphs evaluated
4–7% better F1 than prior sparsifiers at similar sparsity
Up to 30% improvement on heterophilic graphs
Up to 3.9× lower peak memory

Data.	GSAINT-E	DropEdge	MoG	SparseGAT	NeuralSp.	SGS-GNN
Corn.	46.49 ± 2.65	43.24 ± 2.20	42.16 ± 3.08	51.35 ± 0.10	72.43 ± 6.48	**74.59 ± 1.32**
Texa.	69.19 ± 2.76	54.95 ± 3.37	57.30 ± 4.83	66.66 ± 1.27	**84.44 ± 1.53**	76.22 ± 2.02
Wisc.	66.27 ± 2.29	48.36 ± 0.92	53.33 ± 1.64	56.86 ± 0.00	52.83 ± 47.00	**76.08 ± 3.14**
reed.	54.51 ± 1.98	60.03 ± 0.05	55.75 ± 2.61	55.69 ± 0.25	58.54 ± 1.60	**64.15 ± 2.28**
amhe.	57.40 ± 0.58	59.00 ± 18.80	56.78 ± 2.05	49.00 ± 0.20	56.85 ± 75.00	**72.75 ± 0.59**
penn.	68.35 ± 0.36	65.87 ± 0.30	OOM	65.00 ± 0.10	OOM	**75.65 ± 0.41**
Roma.	42.91 ± 0.63	46.00 ± 1.20	39.27 ± 0.60	41.18 ± 0.07	44.91 ± 5.79	**64.69 ± 0.12**
corn.	63.47 ± 0.46	61.40 ± 1.45	OOM	53.77 ± 0.39	OOM	**69.15 ± 0.33**
Squi.	42.44 ± 1.10	48.60 ± 0.00	27.67 ± 0.51	29.50 ± 0.00	38.24 ± 0.00	**52.35 ± 0.35**
john.	62.80 ± 0.74	64.14 ± 1.75	OOM	57.57 ± 0.29	57.56 ± 1.06	**73.80 ± 0.33**
Acto.	30.53 ± 0.59	34.64 ± 1.40	27.74 ± 0.96	25.05 ± 0.60	27.85 ± 0.19	**33.88 ± 0.42**
Mine.	79.84 ± 0.06	**80.00 ± 0.00**	**80.00 ± 0.00**	**80.00 ± 0.00**	**80.00 ± 0.10**	**80.00 ± 0.00**
Ques.	**97.08 ± 0.01**	97.00 ± 0.01	97.04 ± 0.01	**97.08 ± 0.01**	97.02 ± 0.01	97.05 ± 0.01
Cham.	57.11 ± 1.22	50.60 ± 0.04	53.25 ± 0.63	60.60 ± 0.15	60.52 ± 0.78	**62.37 ± 0.98**
Tolo.	78.59 ± 0.21	78.40 ± 0.20	78.49 ± 0.28	78.20 ± 0.72	78.16 ± 0.00	**79.98 ± 0.17**
Amaz.	45.75 ± 0.35	43.87 ± 0.67	41.18 ± 0.49	44.23 ± 0.05	47.05 ± 0.47	**50.15 ± 0.34**
GM.	56.44	55.04	51.15	53.04	58.25	65.99
Cora	57.63 ± 14.73	65.09 ± 0.44	**67.26 ± 1.11**	61.07 ± 0.39	56.68 ± 0.32	65.58 ± 0.69
DBLP	81.19 ± 0.33	87.20 ± 0.15	72.37 ± 0.63	**84.68 ± 1.00**	73.39 ± 0.67	80.37 ± 0.16
Comp.	90.37 ± 0.25	60.65 ± 4.66	OOM	88.91 ± 0.00	75.32 ± 4.11	**90.97 ± 0.31**
PubM.	**87.62 ± 0.14**	86.00 ± 1.21	83.84 ± 0.58	75.30 ± 0.35	73.97 ± 0.40	87.52 ± 0.15
Cora.	**85.29 ± 0.60**	84.80 ± 0.20	OOM	80.60 ± 0.40	79.30 ± 0.87	83.99 ± 0.53
Smal.	76.44 ± 1.21	76.47 ± 0.31	78.43 ± 0.73	75.70 ± 0.43	75.79 ± 9.00	**76.94 ± 0.76**
CS	94.09 ± 0.09	93.30 ± 0.32	72.88 ± 0.32	92.35 ± 0.06	94.36 ± 0.40	**94.25 ± 0.15**
Phot.	93.63 ± 0.25	80.47 ± 59.10	83.84 ± 0.58	**95.30 ± 0.02**	94.16 ± 0.25	93.99 ± 0.25
Phys.	96.23 ± 0.12	97.28 ± 0.70	OOM	95.96 ± 0.06	96.38 ± 0.04	**96.27 ± 0.09**
Cite.	86.75 ± 0.22	77.40 ± 0.10	78.43 ± 0.73	58.75 ± 0.10	65.40 ± 1.20	**86.78 ± 0.28**
wiki	80.19 ± 0.16	80.19 ± 0.16	72.88 ± 0.32	79.26 ± 0.11	73.03 ± 1.10	**81.49 ± 0.31**
GM.	81.36	80.36	76.04	76.78	74.89	82.87

Table 1: Mean F1-score (%) ± std. dev. for baseline sparsifiers with 20% edges. Bold: best method. OOM: out of memory. GM: geometric mean, computed over datasets where all methods report results

Data	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	NS	SG	MoG	SGS*	SGS+	Imp.(%)	#
amhe.	2.24K	182K	2.42 (1.8×)	3.12 (2.3×)	7.44 (5.6×)	**1.33** (1.0×)	1.33 (1.0×)	–	1
amaz.	24.5K	186K	3.85 (2.5×)	2.68 (1.8×)	4.50 (3.0×)	1.57 (1.0×)	**1.51** (1.0×)	3.2	1
Tolo.	11.8K	1.04M	5.60 (1.5×)	8.48 (2.2×)	4.24 (1.1×)	6.35 (1.7×)	**3.83** (1.0×)	39.6	2
John.	5.18K	373K	8.12 (3.0×)	6.09 (2.2×)	OOM	**2.72** (1.0×)	2.72 (1.0×)	–	1
corn.	18.7K	1.58M	OOM	18.79 (3.9×)	OOM	8.58 (1.8×)	**4.80** (1.0×)	44.0	2
arxi.(3)	169K	2.32M	22.06 (2.7×)	22.01 (2.7×)	16.34 (2.0×)	11.62 (1.4×)	**8.08** (1.0×)	30.5	3
arxi.(5)	169K	2.32M	22.06 (3.9×)	22.01 (3.9×)	16.34 (2.9×)	7.69 (1.4×)	**5.62** (1.0×)	26.9	5
Wiki	11.7K	432K	3.62 (1.2×)	3.96 (1.3×)	5.51 (1.8×)	**3.11** (1.0×)	3.12 (1.0×)	–	1
Redd.(115)	233K	114.6M	OOM	OOM	OOM	12.96 (1.4×)	**9.03** (1.0×)	30.4	115
Redd.(230)	233K	114.6M	OOM	OOM	OOM	6.58 (1.3×)	**4.92** (1.0×)	25.2	230

Table 2: Peak GPU memory (GB). SGS*/SGS+ denote runs without/with gradient checkpointing. For each dataset, the minimum memory is normalized to 1.0 (bold); parentheses show relative factors. Imp.(%) reports the memory reduction from checkpointing. #-METIS partitions of SGS-GNN; NS: NeuralSparse; SG: SparseGAT.

Acknowledgements: This work used computing resources at PNNL and Purdue University and was supported by DOE Advanced Computing Technologies, PNNL LDRD, DOE grant SC-0022260, and the DoW CoE-AIML at Howard University under Contract W911NF-20-2-0277. PNNL is operated by Battelle under Contract DE-AC05-76RL01830. The views expressed are those of the authors.