NEURAL INFORMATION PROCESSING SYSTEMS
2025

GC4NC: A Benchmark Framework for Graph Condensation on Node Classification with New Insights

Shengbo Gong*, Juntong Ni*, Carl Yang, Wei Jin

EMORY UNIVERSITY

Noveen Sachdeva

Google DeepMind

Background

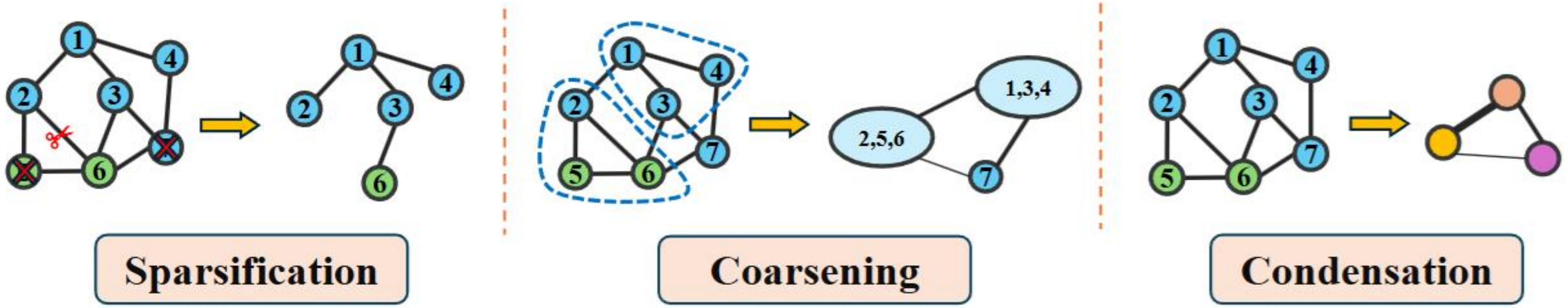


Figure 2: Illustration of key differences among three strategies of graph reduction: Graph sparsification selects significant nodes and edges while discarding others, graph coarsening groups and aggregates similar nodes and edges to construct a smaller graph, and graph condensation learns a synthetic graph from scratch.

- Large graph datasets are challenging. Graph condensation enables GNN to preserve the performance with a smaller graph.
- Many papers discussed the potential of robustness and privacy application but never verified.
- Unified protocols are needed.

Image from "A Comprehensive Survey on Graph Reduction: Sparsification, Coarsening, and Condensation", IJCAI 2024

Benchmark design

- Focus on Node Classification task
  - as over 90% of GC papers work on it
- Three stages: initialization, training, evaluation
- Unified approach for fair comparison
  - fix the frequency of intermediate validation
  - fix the downstream GCN hyperparameters

EMORY MELODY

Benchmark design

- Comprehensive baselines
  - Trajectory-matching (TM)
  - Distribution-matching (DM)
  - Gradient-matching (GM)

- Multifaceted evaluation
  - Transferability
  - Initialization (connected to graph reduction methods)
  - NAS (Neural Architecture Search)
  - robustness, privacy, and graph property (new!)

Observations: Performance and Efficiency

Condensation								Whole
TM		*DM*	*GM*					
GEOM	**SFGC**	**GCDM**	**GCondX**	**GCond**	**DosCond**	**MSGC**	**SGDD**	
67.61	66.27	70.65	67.79	70.05	69.41	60.24	**71.87**	
70.70	70.27	71.27	69.69	69.15	70.83	**72.08**	70.52	72.6
73.03	72.36	72.08	68.38	69.35	72.18	72.21	69.65	
78.14	75.11	79.21	79.74	80.17	**80.65**	80.54	80.15	
82.29	79.55	80.26	78.67	80.81	80.85	80.98	80.29	81.5
82.82	80.54	80.68	78.60	80.54	81.15	80.94	81.04	
69.64	67.61	77.62	72.03	77.36	58.13	75.25	**78.11**	
76.21	66.89	76.63	72.05	78.05	52.70	**78.26**	78.07	78.6
78.49	67.61	77.48	71.97	76.46	76.45	78.20	75.95	
64.91	64.91	60.04	59.40	60.49	55.70	57.66	58.50	
68.78	66.58	60.59	62.46	63.88	57.39	64.85	59.18	71.4
69.59	67.03	60.71	59.93	64.23	61.06	65.73	63.76	

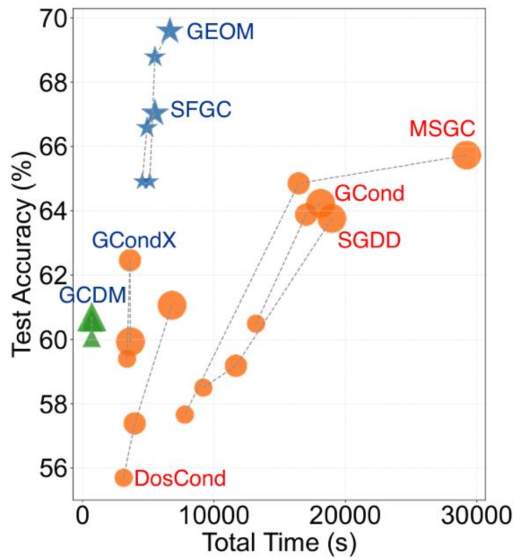


Figure 1: Test accuracy vs. total time for *structure-free* and *structure-based* condensation methods on *Arxiv*. TM is represented by ★, GM by •, and DM by ▲. Marker sizes increase with reduction rates of 0.05%, 0.25%, and 0.50%.

- **TM-based** methods excel in condensation performance but lack efficiency.

Image dataset condensation results from “DC-BENCH: Dataset Condensation Benchmark”, NeurIPS 2022

Observations: Performance and Efficiency

Condensation								Whole
TM		*DM*	*GM*					
GEOM	SFGC	GCDM	GCondX	GCond	DosCond	MSGC	SGDD	
67.61	66.27	70.65	67.79	70.05	69.41	60.24	**71.87**	
70.70	70.27	71.27	69.69	69.15	70.83	**72.08**	70.52	72.6
73.03	72.36	72.08	68.38	69.35	72.18	72.21	69.65	
78.14	75.11	79.21	79.74	80.17	**80.65**	80.54	80.15	
82.29	79.55	80.26	78.67	80.81	80.85	80.98	80.29	81.5
82.82	80.54	80.68	78.60	80.54	81.15	80.94	81.04	
69.64	67.61	77.62	72.03	77.36	58.13	75.25	**78.11**	
76.21	66.89	76.63	72.05	78.05	52.70	**78.26**	78.07	78.6
78.49	67.61	77.48	71.97	76.46	76.45	78.20	75.95	
64.91	64.91	60.04	59.40	60.49	55.70	57.66	58.50	
68.78	66.58	60.59	62.46	63.88	57.39	64.85	59.18	71.4
69.59	67.03	60.71	59.93	64.23	61.06	65.73	63.76	

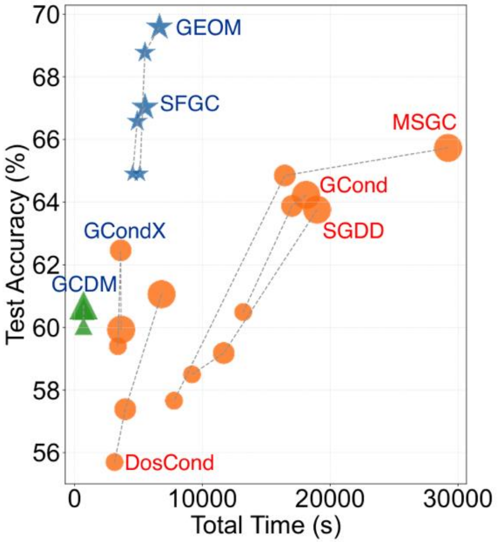
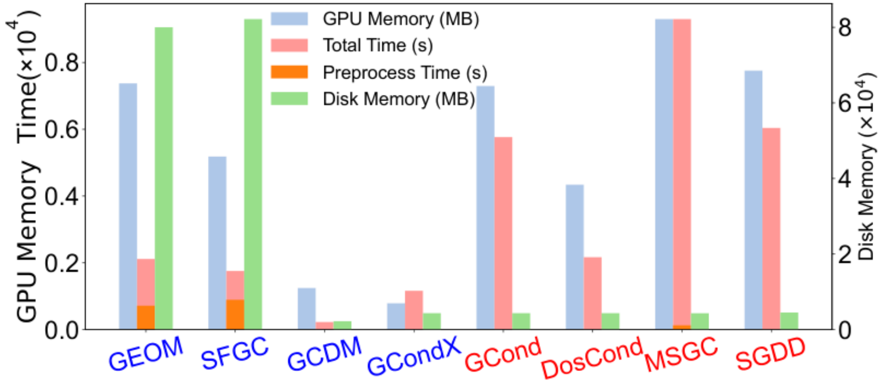


Figure 1: Test accuracy vs. total time for *structure-free* and *structure-based* condensation methods on *Arxiv*. TM is represented by ★, GM by ●, and DM by ▲. Marker sizes increase with reduction rates of 0.05%, 0.25%, and 0.50%.



- **Structure-free** methods are more efficient than structure-based ones.

Image dataset condensation results from “DC-BENCH: Dataset Condensation Benchmark”, NeurIPS 2022

Observations: Performance and Efficiency

Condensation								**Whole**
TM		*DM*	*GM*					
GEOM	SFGC	GCDM	GCondX	GCond	DosCond	MSGC	SGDD	
67.61	66.27	70.65	67.79	70.05	69.41	60.24	**71.87**	
70.70	70.27	71.27	69.69	69.15	70.83	**72.08**	70.52	72.6
73.03	72.36	72.08	68.38	69.35	72.18	72.21	69.65	
78.14	75.11	79.21	79.74	80.17	**80.65**	80.54	80.15	
82.29	79.55	80.26	78.67	80.81	80.85	80.98	80.29	81.5
82.82	80.54	80.68	78.60	80.54	81.15	80.94	81.04	
69.64	67.61	77.62	72.03	77.36	58.13	75.25	**78.11**	
76.21	66.89	76.63	72.05	78.05	52.70	**78.26**	78.07	78.6
78.49	67.61	77.48	71.97	76.46	76.45	78.20	75.95	
64.91	64.91	60.04	59.40	60.49	55.70	57.66	58.50	
68.78	66.58	60.59	62.46	63.88	57.39	64.85	59.18	71.4
69.59	67.03	60.71	59.93	64.23	61.06	65.73	63.76	

Figure 2: Comparison of GPU memory, disk memory, preprocess time, and total time on *Arxiv* ($r = 0.5\%$).

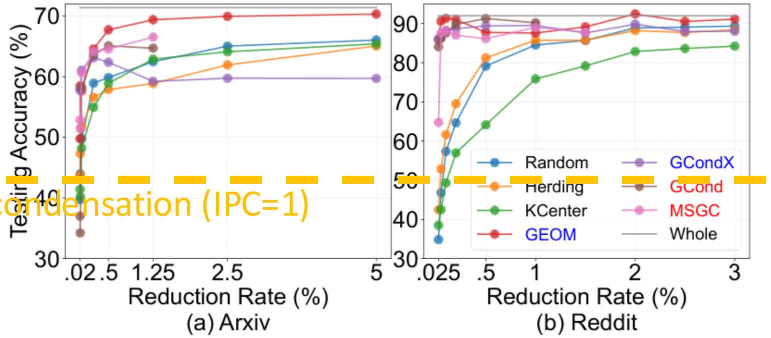


Figure 3: Varying reduction rates on Arxiv and Reddit. No mark represents OOM when the reduction rate is too large for a method.

- GC outperforms image dataset condensation at the extremely low reduction rate.

Image dataset condensation results from “DC-BENCH: Dataset Condensation Benchmark”, NeurIPS 2022

EMORY MELODY

Observations: Privacy Protection

MIA Acc: Membership Inference Attack, MIA Acc reflects the probability that an adversary (confidence threshold) can correctly identify whether a node belongs to the training set.

Methods	*Cora, r = 2.6%*		*Citeseer, r = 1.8%*		*Arxiv, r = 0.5%*	
	MIA Acc (↓)	Acc (↑)	MIA Acc (↓)	Acc (↑)	MIA Acc (↓)	Acc (↑)
Whole	74.87 ± 1.16	81.50 ± 0.50	81.76 ± 1.01	72.61 ± 0.27	54.26 ± 0.11	71.43 ± 0.11
GCond	72.10 ± 0.96	80.54 ± 0.67	74.11 ± 0.61	69.35 ± 0.82	**53.04 ± 0.18**	64.23 ± 0.16
GCondX	66.83 ± 0.81	78.60 ± 0.31	71.97 ± 0.58	68.38 ± 0.45	54.64 ± 0.17	59.93 ± 0.54
DosCond	69.70 ± 0.50	81.15 ± 0.50	74.33 ± 0.34	72.18 ± 0.61	54.04 ± 0.79	61.06 ± 0.59
SGDD	70.43 ± 1.63	81.04 ± 0.54	77.07 ± 4.32	69.65 ± 1.68	53.29 ± 0.46	63.76 ± 0.22
GDEM	**60.66 ± 1.26**	81.76 ± 0.53	70.01 ± 2.94	71.74 ± 0.90	-	-
GEOM	67.90 ± 0.55	**82.82 ± 0.17**	**67.55 ± 0.62**	**73.03 ± 0.31**	53.80 ± 0.19	**69.59 ± 0.24**
SFGC	67.29 ± 1.02	80.54 ± 0.45	72.12 ± 0.44	72.36 ± 0.53	54.49 ± 0.53	67.03 ± 0.48

- TM and eigen-decomposition methods balance privacy and performance.

Observations: Robustness and Denoising Ability

Adversarial Structural Noise: We employ PR-BCD, a scalable adversarial noise using Projected Gradient Descent.

Dataset	Method	Feature Noise		Structural Noise		Adversarial Structural Noise	
		Test Acc. ↑	Perf. Drop ↓	Test Acc. ↑	Perf. Drop ↓	Test Acc. ↑	Perf. Drop ↓
Citeseer 1.8%	Whole	64.07	11.75%	57.63	20.62%	53.90	25.76%
	GCond	**64.06**	**7.63%**	**65.64**	**5.35%**	**66.19**	**4.55%**
	GCondX	61.27	10.40%	60.42	11.65%	60.75	11.15%
	GEOM	58.77	19.53%	51.41	29.60%	57.94	20.67%
Cora 2.6%	Whole	74.77	8.26%	72.13	11.49%	66.63	18.24%
	GCond	67.62	16.04%	63.14	21.61%	68.90	14.45%
	GCondX	**67.72**	**13.85%**	**63.95**	**18.63%**	**69.24**	**11.91%**
	GEOM	49.68	40.01%	53.59	35.29%	66.32	19.93%
Flickr 1%	Whole	46.68	1.51%	42.60	10.13%	44.44	6.24%
	GCond	**46.29**	**1.49%**	**46.97**	**0.04%**	43.90	6.58%
	GCondX	45.60	2.11%	46.19	0.83%	42.00	9.83%
	GEOM	45.38	1.63%	45.52	1.32%	**44.72**	**3.06%**

- **Structure-based** approaches offer superior robustness.

Observations: Graph Properties Preservation

Supervised DBI (Davies-Bouldin Index) measures the quality of a **class** by evaluating the average similarity between each class and its most similar class

Graph Property		VNG	GCond	MSGC	SGDD	Avg.	Whole
Density% (Struc.)	*Cora*	52.17	82.28	22.00	100.00	64.11	0.14
	Corr.	-0.81	0.07	0.55	0.13	-0.02	-
Max Eigenvalue (Spectra)	*Cora*	3.73	34.90	1.69	14.09	13.60	169.01
	Corr.	0.85	0.25	0.95	0.28	0.58	-
DBI (Label & Feature)	*Cora*	3.69	1.84	0.70	4.34	2.64	9.28
	Corr.	0.81	0.93	0.94	0.97	**0.91**	-
DBI-AGG (Label & Feat. & Stru.)	*Cora*	3.59	0.38	0.57	0.18	1.18	4.67
	Corr.	0.99	0.93	0.95	0.89	**0.94**	-
Homophily (Label & Struc.)	*Cora*	0.14	0.16	0.19	0.13	0.16	0.81
	Corr.	-0.83	-0.68	-0.46	-0.80	-0.69	-

- **Node and aggregated features** are preserved in condensed graphs.

Observations: Graph Properties Preservation

Table 17: Correlation between condensed graph properties and model performance.

	VNG	GCond	MSGC	SGDD	Correlation
Avg ACC	63.34	69.40	69.02	68.95	-
Density (%)	-0.81	0.07	0.55	0.13	0.91
Max Eigen	0.85	0.25	0.95	0.28	-0.51
DBI	0.81	0.93	0.94	0.97	0.96
DBI-AGG	0.99	0.93	0.95	0.89	-0.80
Homophily	-0.83	-0.68	-0.46	-0.80	0.54

- Preserving Density%, DBI tends to be beneficial for downstream tasks.

Observations: Transferability and Initialization

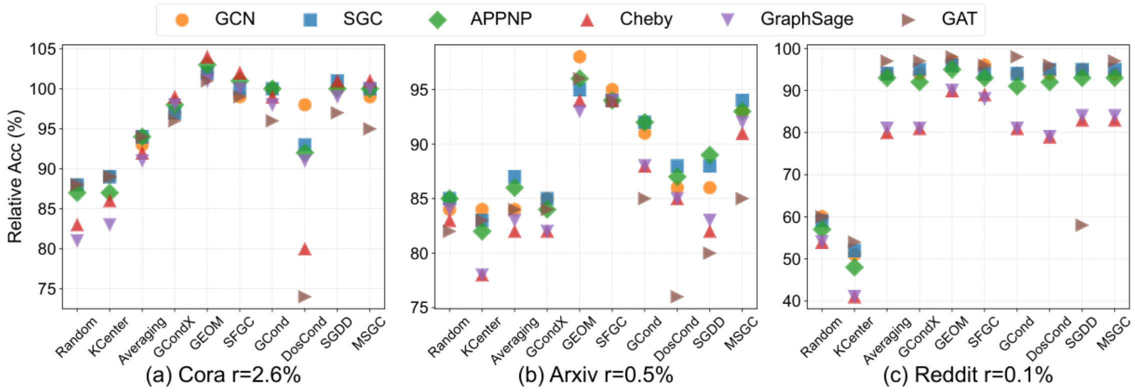


Figure 4: Condensed graph performance evaluated by different GNNs. The **relative accuracy** refers to the accuracy preserved compared to training on the whole dataset.

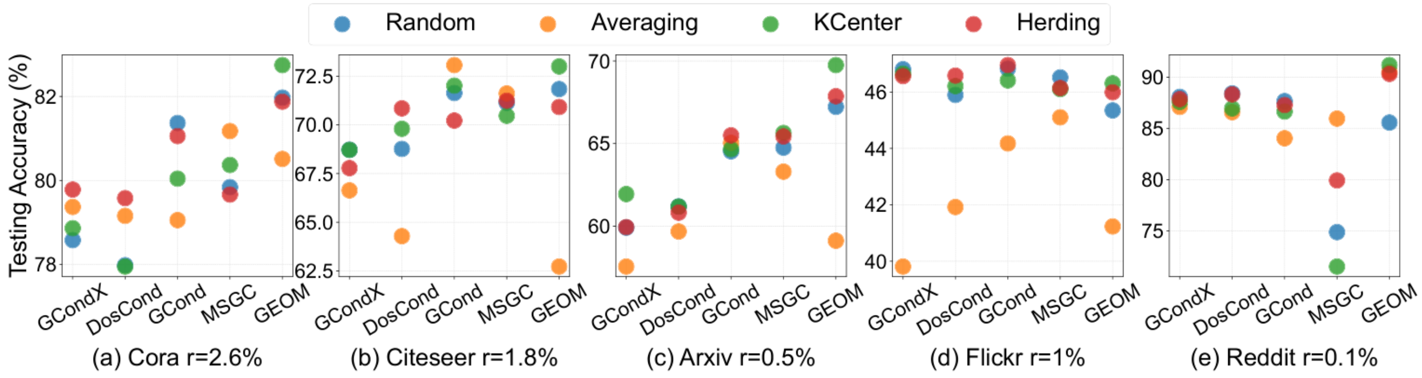


Figure 5: Test accuracy for different methods with different initialization.

- GC methods vary in transferability across datasets.
- Current initialization strategies lack consistent impact.

Observations: NAS

Recall: how many of the G top-K architectures also appear in the top K on the G’
Top 1: the performance of the G’ top 1 architecture on the G

Table 13: Recall@K of top-K architectures evaluated on Cora with $r = 0.5$.

	Random	**K-Center**	**GCondX**	**SFGC**	**GEOM**	**GCond**	**DosCond**	**MSGC**
Recall@5	0.0	0.0	0.2	0.0	0.2	0.0	0.0	0.0
Recall@10	0.0	0.0	0.2	0.1	0.2	0.2	0.1	0.1
Recall@20	0.4	0.35	0.5	0.45	0.4	0.45	0.25	0.4

	Random	**K-Center**	**GCondX**	**SFGC**	**GEOM**	**GCond**	**DosCond**	**MSGC**	**Whole**
Top 1 (%)	81.88	81.74	81.49	**82.42**	82.19	81.82	81.91	82.40	82.51
Acc. Corr.	0.56	0.47	0.40	**0.72**	0.65	0.70	0.14	0.71	-
Rank Corr.	0.64	0.60	0.57	0.71	0.74	0.66	0.20	**0.78**	-

- TM or inner optimization (like GCondX, GCond) is key for NAS effectiveness.

Conclusions

- First benchmark for GC methods with multi-dimension evaluation.
- Novel insights on privacy, robustness and graph properties.
- Inspires future directions for better performance and scalability.

Toolkit

- Run Graph Condensation with 3 lines of code and evaluate with another 2!

```
from graphslim.dataset import *
from graphslim.evaluation import *
from graphslim.condensation import GCond
from graphslim.config import cli

args = cli(standalone_mode=False)
args.reduction_rate = 0.5
args.device = 'cuda:0'
graph = get_dataset('cora', args=args)
agent = GCond(setting='trans', data=graph, args=args)
reduced_graph = agent.reduce(graph, verbose=True)
evaluator = Evaluator(args)
res_mean, res_std = evaluator.evaluate(reduced_graph, model_type='GCN')
```

GraphSlim

https://github.com/Emory-Melody/GraphSlim

THANK YOU