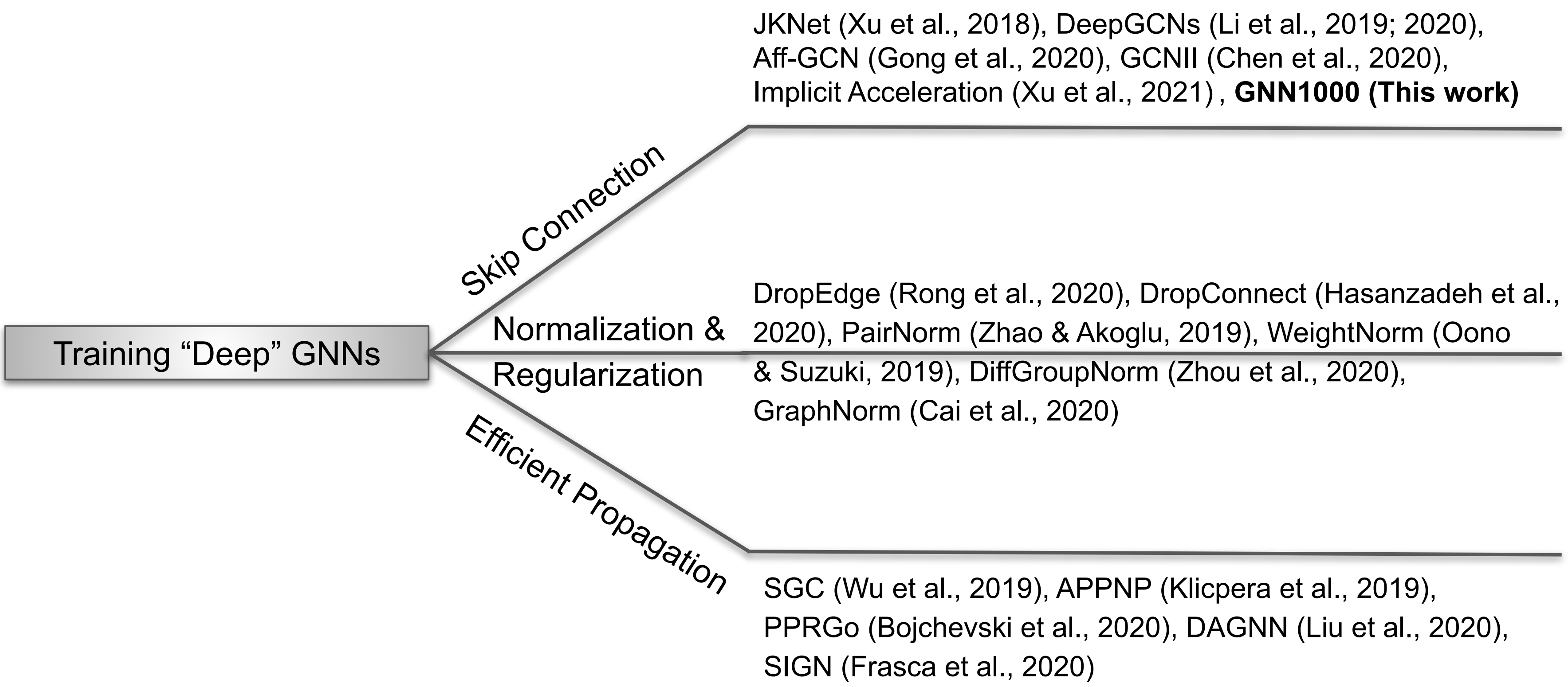
https://www.deepgcns.org/arch/gnn1000

Training Graph Neural Networks with **1000 Layers**

Guohao Li, Matthias Müller, Bernard Ghanem, Vladlen Koltun

intel

KAUST


Training “Deep” GNNs
Skip Connection
JKNet (Xu et al., 2018), DeepGCNs (Li et al., 2019; 2020), Aff-GCN (Gong et al., 2020), GCNII (Chen et al., 2020), Implicit Acceleration (Xu et al., 2021) , GNN1000 (This work)
Normalization & Regularization
DropEdge (Rong et al., 2020), DropConnect (Hasanzadeh et al., 2020), PairNorm (Zhao & Akoglu, 2019), WeightNorm (Oono & Suzuki, 2019), DiffGroupNorm (Zhou et al., 2020), GraphNorm (Cai et al., 2020)
Efficient Propagation
SGC (Wu et al., 2019), APPNP (Klicpera et al., 2019), PPRGo (Bojchevski et al., 2020), DAGNN (Liu et al., 2020), SIGN (Frasca et al., 2020)

Memory complexity of training GNNs

Full batch: **O(LND)**

L - number of layers
N - number of nodes
D - number of features
(assume D is the same for all the layers)

Mini-batch:

Cluster-GCN: **O(LND) - > O(LBD)**

B - number of nodes in subgraphs, B<N

This work:

O(LND) - > O(ND)

- How can we reduce memory complexity?

- Can we reduce the memory complexity in the L dimension?

Chiang, Wei-Lin, et al. "Cluster-GCN: An efficient algorithm for training deep and large graph convolutional networks." SIGKDD. 2019.

Related work

The Reversible Residual Network: Backpropagation Without Storing Activations

Aidan N. Gomez*[1], Mengye Ren*[1,2,3], Raquel Urtasun[1,2,3], Roger B. Grosse[1,2]
University of Toronto[1]
Vector Institute for Artificial Intelligence[2]
Uber Advanced Technologies Group[3]
{aidan, mren, urtasun, rgrosse}@cs.toronto.edu

Deep Equilibrium Models

Shaojie Bai
Carnegie Mellon University

J. Zico Kolter
Carnegie Mellon University
Bosch Center for AI

Vladlen Koltun
Intel Labs

DNN: **O(L)**

Reversible CNN / DEQ: **O(1)**

*only consider the L dimension

Memory Efficient GNNs

$$\langle X_1, X_2, ..., X_C \rangle \mapsto \langle X'_1, X'_2, ..., X'_C \rangle$$

Reversible GNN:

Forward:

$$X'_0 = \sum_{i=2}^{C} X_i$$

$$X'_i = f_{w_i}(X'_{i-1}, A, U) + X_i,\ i \in \{1, \cdots, C\},$$

Inverse:

$$X_i = X'_i - f_{w_i}(X'_{i-1}, A, U),\ i \in \{2, \cdots, C\}$$

$$X'_0 = \sum_{i=2}^{C} X_i$$

$$X_1 = X'_1 - f_{w_1}(X'_0, A, U).$$

Weight-tied Reversible GNN:

$$f_{w_i}^{(1)} := f_{w_i}^{(2)} \ldots := f_{w_i}^{(L)},\ i \in \{1, \cdots, C\}$$

DEQ-GNN:

$$Z^* = f_w^{\mathrm{DEQ}}(Z^*, X, A, U),$$

Do not need to store the intermediate node features.

O(LND) - > O(ND)

Results: Summary

1. Regular GNNs quickly run out of memory.

2. We can train huge overparameterized RevGNNs on a single GPU and achieve the best performance.

3. We can train smaller GNNs with weight-tying or DEQ and still reach promising results

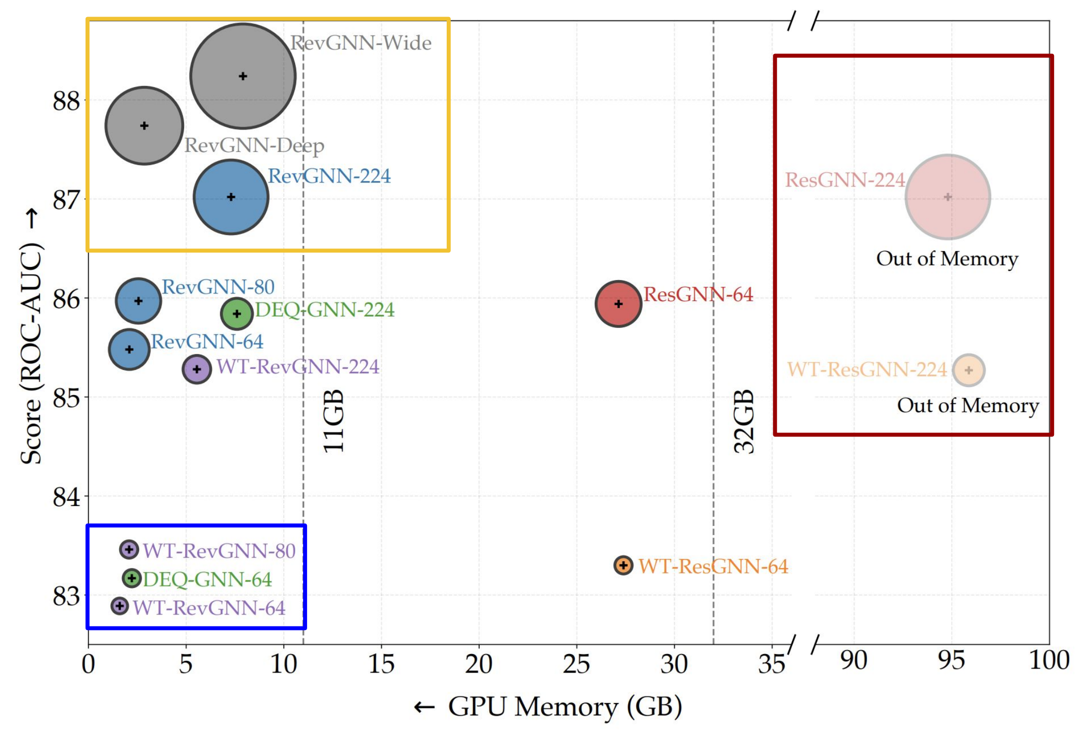
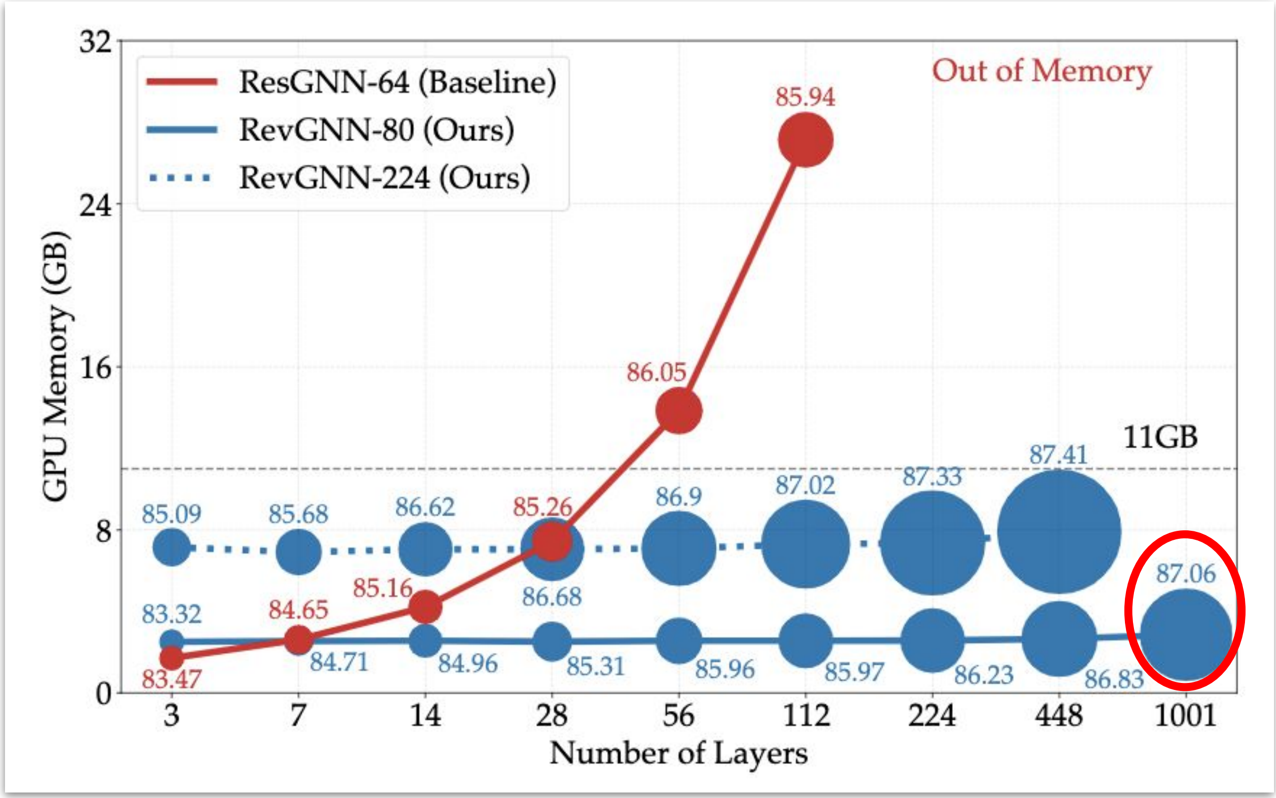


Fig. Performance versus GPU memory consumption on the ogbn-proteins dataset for 112 layer deep networks.

Results: Complexity Analysis

Method	Memory	Params	Time
Full-batch GNN	$\mathcal{O}(LND)$	$\mathcal{O}(LD^2)$	$\mathcal{O}(L\left\|A\right\|_0 D + LND^2)$
GraphSAGE	$\mathcal{O}(R^L BD)$	$\mathcal{O}(LD^2)$	$\mathcal{O}(R^L ND^2)$
VR-GCN	$\mathcal{O}(LND)$	$\mathcal{O}(LD^2)$	$\mathcal{O}(L\left\|A\right\|_0 D + LND^2 + R^L ND^2)$
FastGCN	$\mathcal{O}(LRBD)$	$\mathcal{O}(LD^2)$	$\mathcal{O}(RLND^2)$
Cluster-GCN	$\mathcal{O}(LBD)$	$\mathcal{O}(LD^2)$	$\mathcal{O}(L\left\|A\right\|_0 D + LND^2)$
GraphSAINT	$\mathcal{O}(LBD)$	$\mathcal{O}(LD^2)$	$\mathcal{O}(L\left\|A\right\|_0 D + LND^2)$
Weight-tied GNN	$\mathcal{O}(LND)$	$\mathcal{O}(D^2)$	$\mathcal{O}(L\left\|A\right\|_0 D + LND^2)$
RevGNN	$\mathcal{O}(ND)$	$\mathcal{O}(LD^2)$	$\mathcal{O}(L\left\|A\right\|_0 D + LND^2)$
WT-RevGNN	$\mathcal{O}(ND)$	$\mathcal{O}(D^2)$	$\mathcal{O}(L\left\|A\right\|_0 D + LND^2)$
DEQ-GNN	$\mathcal{O}(ND)$	$\mathcal{O}(D^2)$	$\mathcal{O}(K\left\|A\right\|_0 D + KND^2)$
RevGNN + Subgraph Sampling	$\mathcal{O}(BD)$	$\mathcal{O}(LD^2)$	$\mathcal{O}(L\left\|A\right\|_0 D + LND^2)$
WT-RevGNN + Subgraph Sampling	$\mathcal{O}(BD)$	$\mathcal{O}(D^2)$	$\mathcal{O}(L\left\|A\right\|_0 D + LND^2)$
DEQ-GNN + Subgraph Sampling	$\mathcal{O}(BD)$	$\mathcal{O}(D^2)$	$\mathcal{O}(K\left\|A\right\|_0 D + KND^2)$

Results: Constant Memory with RevGNN



Train 1001-layer GNN with only 2.86G peak GPU memory!

The deepest GNN by one order of magnitude.

Results: SOTA with RevGNN (ogbn-proteins)

Rank	Method	Test ROC-AUC	Validation ROC-AUC	Contact	References	#Params	Hardware	Date
1	**RevGNN-Wide**	0.8824 ± 0.0015	0.9450 ± 0.0008	Guohao Li - DeepGCNs.org	Paper, Code	68,471,608	NVIDIA RTX 6000 (48G)	Jun 16, 2021
2	**RevGNN-Deep**	0.8774 ± 0.0013	0.9326 ± 0.0006	Guohao Li - DeepGCNs.org	Paper, Code	20,031,384	NVIDIA RTX 6000 (48G)	Jun 16, 2021
3	**GAT+BoT**	0.8765 ± 0.0008	0.9280 ± 0.0008	Yangkun Wang (DGL Team)	Paper, Code	2,484,192	Tesla A100 (40GB GPU)	Jun 16, 2021
4	GAT + labels + node2vec	0.8711 ± 0.0007	0.9217 ± 0.0011	Huixuan Chi	Paper, Code	6,360,470	Tesla V100 (32GB)	Jun 7, 2021
5	**GIPA**	0.8700 ± 0.0010	0.9187 ± 0.0003	Qinkai Zheng (GeaLearn Team)	Paper, Code	4,831,056	GeForce Titan RTX (24GB GPU)	May 13, 2021
6	**UniMP+CrossEdgeFeat**	0.8691 ± 0.0018	0.9258 ± 0.0009	Yelrose (PGL Team)	Paper, Code	1,959,984	Tesla V100 (32GB)	Nov 24, 2020
7	GAT+EdgeFeatureAtt	0.8682 ± 0.0021	0.9194 ± 0.0003	Yangkun Wang (DGL Team)	Paper, Code	2,475,232	p3.8xlarge (15GB GPU)	Nov 6, 2020
8	**UniMP**	0.8642 ± 0.0008	0.9175 ± 0.0006	Yunsheng Shi (PGL team)	Paper, Code	1,909,104	Tesla V100 (32GB)	Sep 8, 2020
9	**DeeperGCN+FLAG**	0.8596 ± 0.0027	0.9132 ± 0.0022	Kezhi Kong	Paper, Code	2,374,568	GeForce RTX 2080 Ti (11GB GPU)	Oct 20, 2020

68M parameters
(about a half of GPT)

Results: SOTA with RevGNN (ogbn-arxiv)

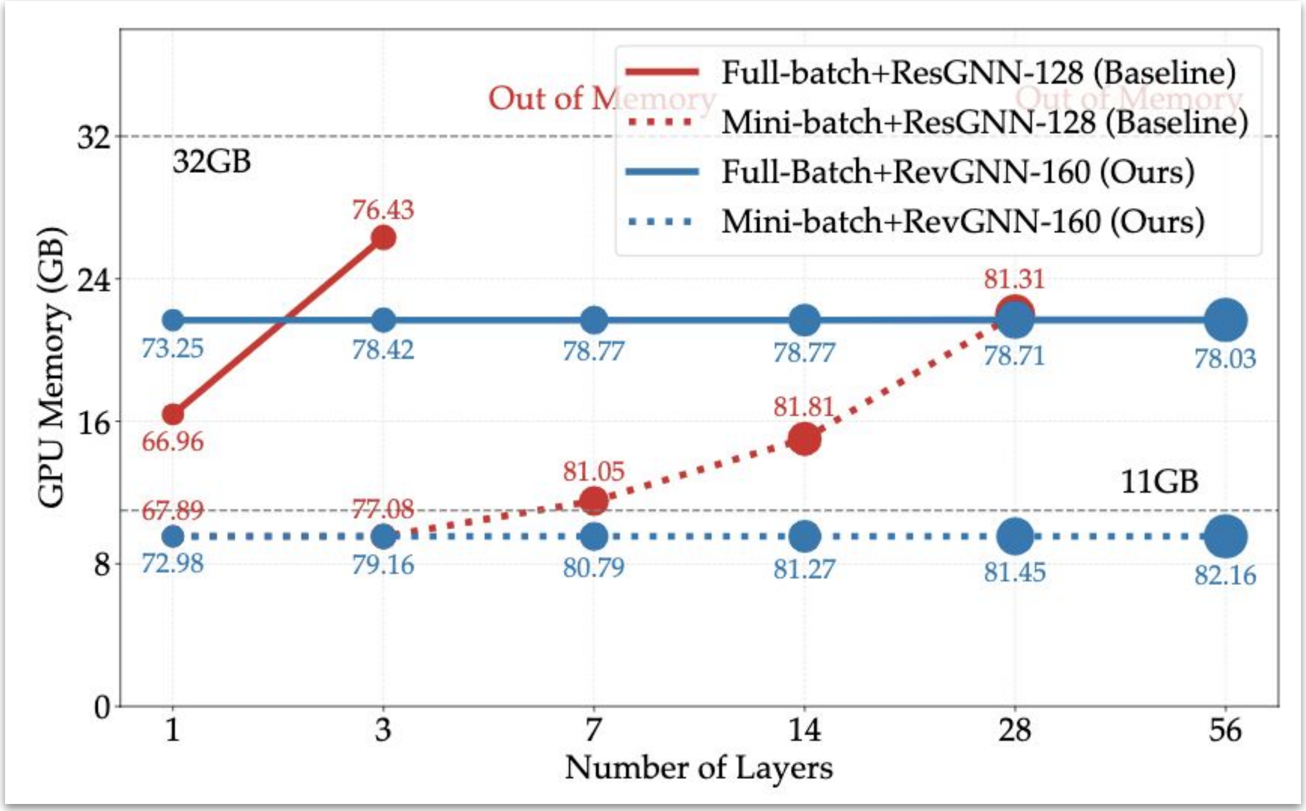
Rank	Method	Test Accuracy	Validation Accuracy	Contact	References	#Params	Hardware	Date
1	**RevGAT+N.Adj+LabelReuse+SelfKD**	0.7426 ± 0.0017	0.7497 ± 0.0008	Guohao Li - DeepGCNs.org	Paper, Code	2,098,256	NVIDIA Tesla V100 (32GB GPU)	Jun 21, 2021
2	GAT+label reuse+self KD	0.7416 ± 0.0008	0.7514 ± 0.0004	Shunli Ren(CMIC@SJTU)	Paper, Code	1,441,580	GeForce RTX 1080Ti (11GB GPU)	Dec 15, 2020
3	**RevGAT+NormAdj+LabelReuse**	0.7402 ± 0.0018	0.7501 ± 0.0010	Guohao Li - DeepGCNs.org	Paper, Code	2,098,256	NVIDIA Tesla V100 (32GB GPU)	Jun 21, 2021
4	GAT+label+reuse+topo loss	0.7399 ± 0.0012	0.7513 ± 0.0009	Mengyang Niu (DAMO DI)	Paper, Code	1,441,580	Tesla V100 (16GB)	Dec 10, 2020
5	**AGDN (GAT-HA+3_heads+labels)**	0.7398 ± 0.0009	0.7519 ± 0.0009	Chuxiong Sun	Paper, Code	1,508,555	Tesla V100 (32GB GPU)	Jan 3, 2021
6	**UniMP_v2**	0.7397 ± 0.0015	0.7506 ± 0.0009	Weiyue Su (PGL Team)	Paper, Code	687,377	Tesla V100 (32GB)	Nov 24, 2020
7	GAT(norm.adj.)+label reuse+C&S	0.7395 ± 0.0012	0.7519 ± 0.0008	Yangkun Wang (DGL Team)	Paper, Code	1,441,580	p3.8xlarge (15GB GPU)	Nov 24, 2020
8	GAT+norm. adj.+label reuse	0.7391 ± 0.0012	0.7516 ± 0.0008	Yangkun Wang (DGL Team)	Paper, Code	1,441,580	p3.8xlarge (15GB GPU)	Nov 11, 2020
9	**GAT + C&S**	0.7386 ± 0.0014	0.7484 ± 0.0007	Horace He (Cornell)	Paper, Code	1,567,000	GeForce RTX 2080 (11GB GPU)	Oct 27, 2020

Ablation: Different GNN operators (ogbn-arxiv)

Model	#L	#Ch	ACC ↑	Mem ↓	Params
ResGCN	28	128	72.46 ± 0.29	11.15	491k
RevGCN	28	128	73.01 ± 0.31	**1.84**	262k
RevGCN	28	180	**73.22** ± 0.19	2.73	500k
ResSAGE	28	128	72.46 ± 0.29	8.93	950k
RevSAGE	28	128	72.69 ± 0.23	**1.17**	491k
RevSAGE	28	180	**72.73** ± 0.10	1.57	953k
ResGEN	28	128	72.32 ± 0.27	21.63	491k
RevGEN	28	128	72.34 ± 0.18	**4.08**	262k
RevGEN	28	180	**72.93** ± 0.10	5.67	500k
ResGAT	5	768	73.76 ± 0.13	9.96	3.87M
RevGAT	5	768	74.02 ± 0.18	**6.30**	2.10M
RevGAT	5	1068	**74.05** ± 0.11	8.49	3.88M

RevGNNs are generic and can be applied to different operators.

Ablation: Mini-batch Training (ogbn-products)



Mini-batch training further reduces the memory consumption of RevGNN and improves its accuracy.

Conclusion

- We study reversible connections, group convolutions, weight tying, and equilibrium models to advance the memory and parameter efficiency of GNNs

- We train overparameterized GNNs with reversible residual connections (RevGNN) which outperform SOTA models on several OGB datasets

- Implementation with PyG and DGL is available at: https://www.deepgcns.org/arch/gnn1000 https://www.deepgcns.org/arch/gnn1000

Training Graph Neural Networks with **1000 Layers**

Guohao Li, Matthias Müller, Bernard Ghanem, Vladlen Koltun

intel

KAUST