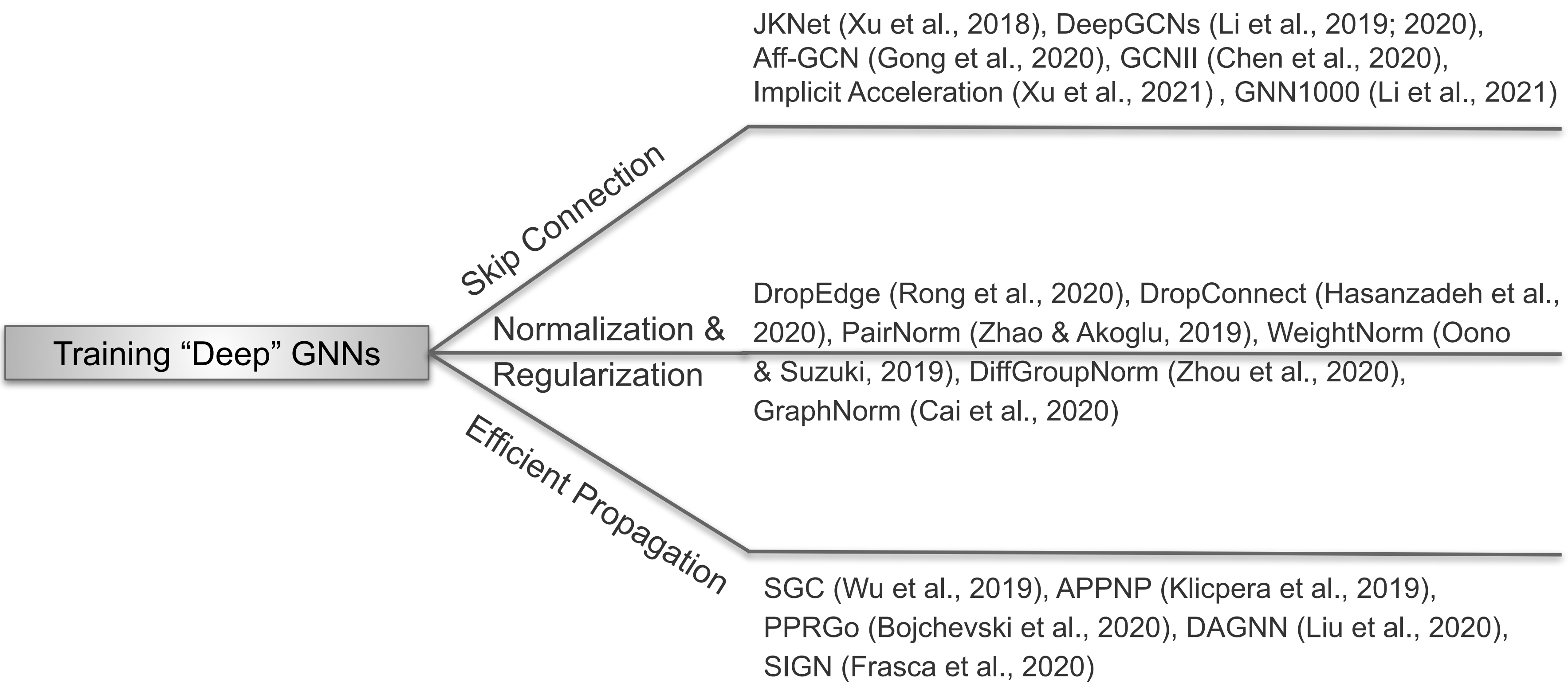
جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

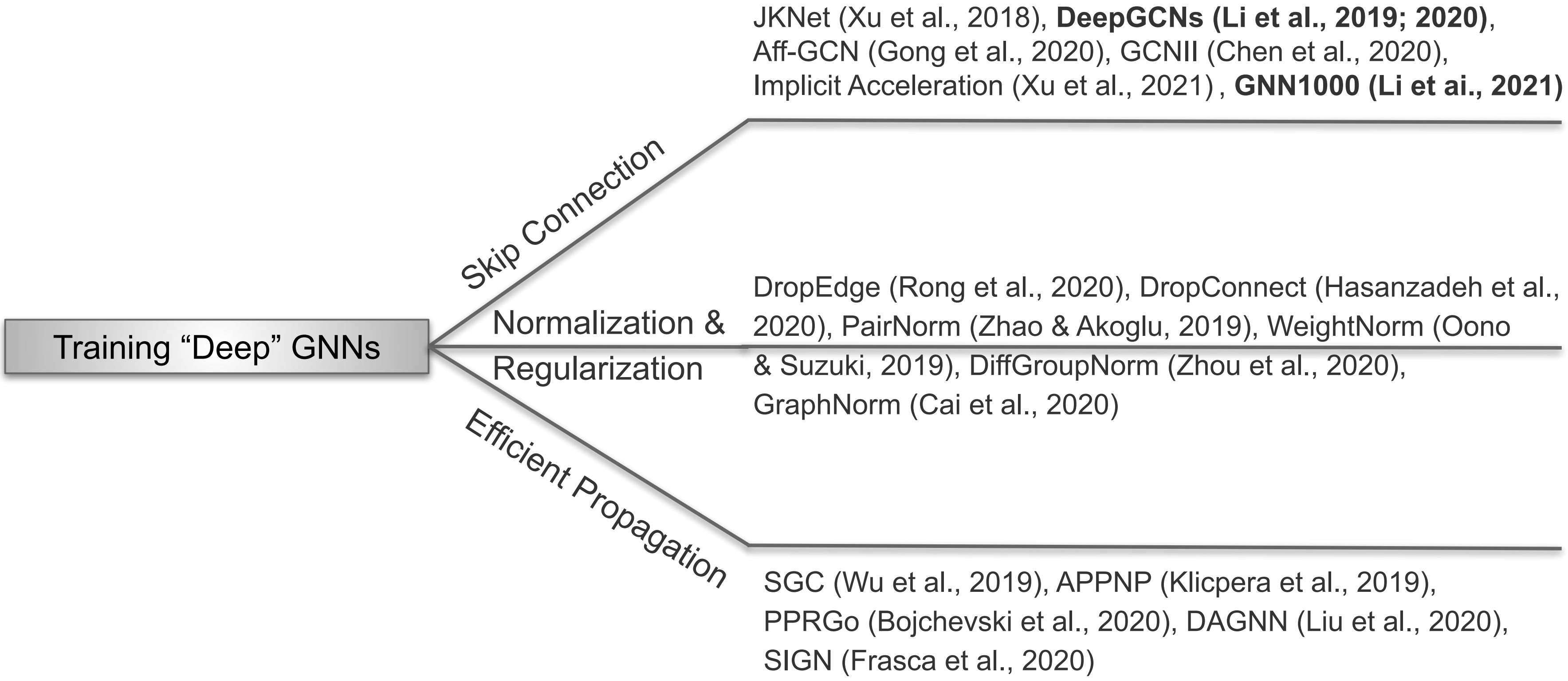
DeepGCNs.org

# DeepGCNs

Guohao Li, KAUST

IVUL


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 (Li et al., 2021)
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)



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 (Li et ai., 2021)**
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)

DeepGCNs.org

# DeepGCNs for Representation Learning on Graphs

- DeepGCNs: Can GCNs Go as Deep as CNNs? (ICCV’2019, TPAMI)
- DeeperGCN: All You Need to Train Deeper GCNs (arXiv’2020)
- Training Graph Neural Networks with 1000 Layers (ICML’2021)

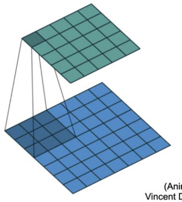
جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

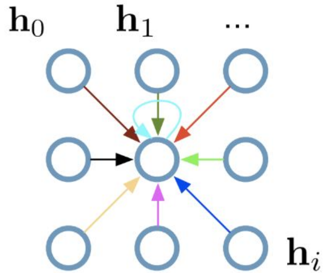
IVUL Guohao Li

# CNN vs. GCN - Comparison

DeepGCNs.org

**Single CNN layer with 3x3 filter:**



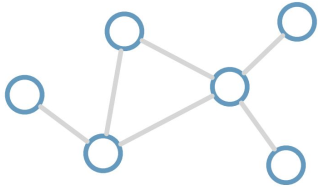




**Full update:**

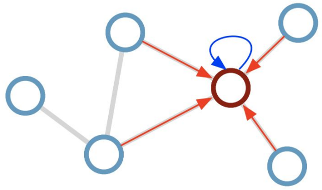
$$\mathbf{h}_4^{(l+1)} = \sigma\left(\mathbf{W}_0^{(l)}\mathbf{h}_0^{(l)} + \mathbf{W}_1^{(l)}\mathbf{h}_1^{(l)} + \cdots + \mathbf{W}_8^{(l)}\mathbf{h}_8^{(l)}\right)$$

Convolutional Neural Network (CNN)

Consider this undirected graph:



Calculate update for node in red:



**Update rule:**

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j \in \mathcal{N}_i} \frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

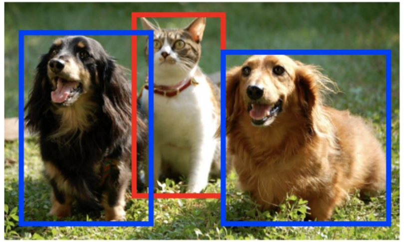
Graph Convolutional Network (GCN)

Slides by Thomas Kipf

Why do we need graph convolutional networks?

# Grid Data vs. General Graphs



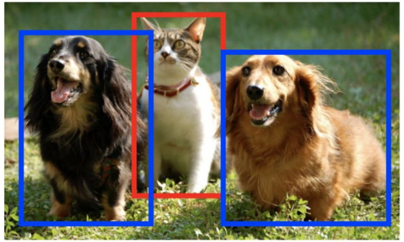


CAT, DOG

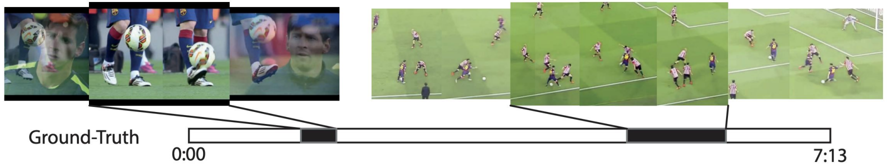
Grid Data :

- Image

# Grid Data vs. General Graphs



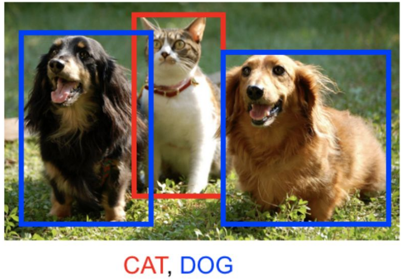
CAT, DOG

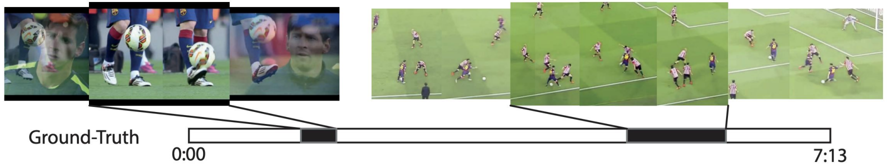



Grid Data :
- Image
- Video

# Grid Data vs. General Graphs

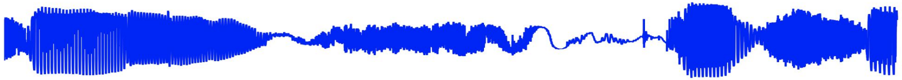




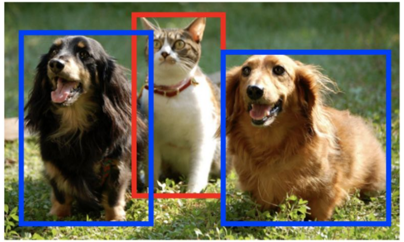

Grid Data :

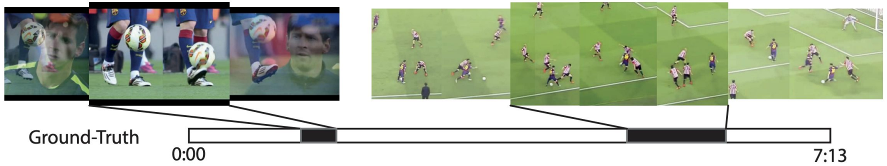
- Image
- Video
- Audio
- Text

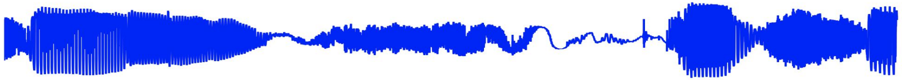


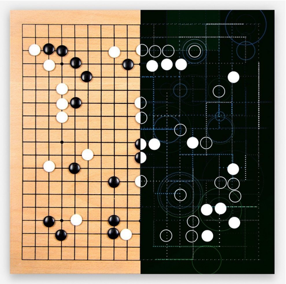
# Grid Data vs. General Graphs

DeepGCNs.org



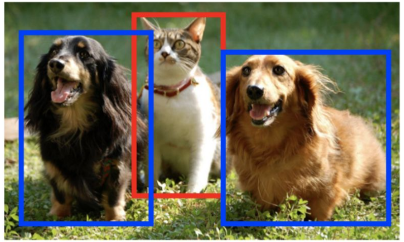



Grid Data :

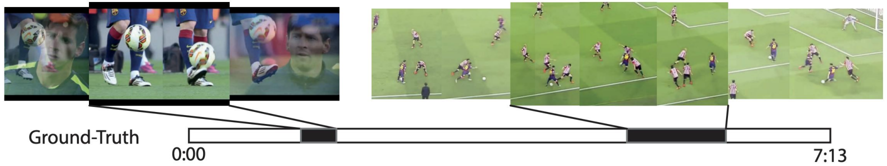
- Image
- Video
- Audio
- Text
- Grid game (Go)
- ...

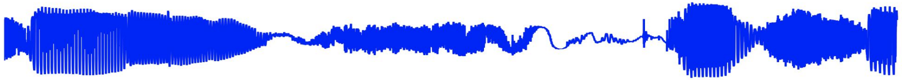
# Grid Data vs. General Graphs

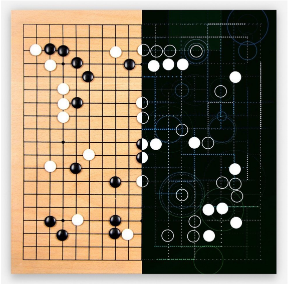




CAT, DOG





Grid Data :

- Image
- Video
- Audio
- Text
- Grid game (Go)
- ...

CNN works well

# Grid Data vs. General Graphs

DeepGCNs.org







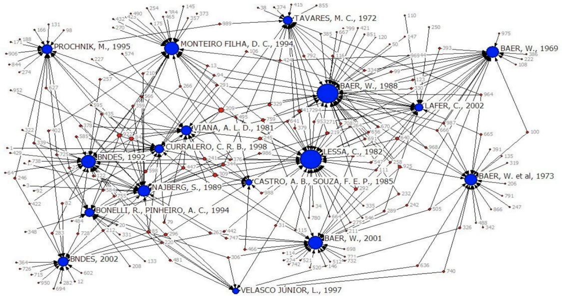
How about non-grid graph structured data?

# Grid Data vs. General Graphs



Lots of real-world applications need to deal with Non-Grid data


General Graphs :
- Social Networks
- Citation Networks

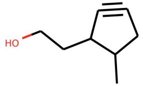


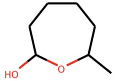
# Grid Data vs. General Graphs

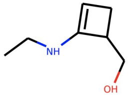
DeepGCNs.org

Lots of real-world applications need to deal with Non-Grid data



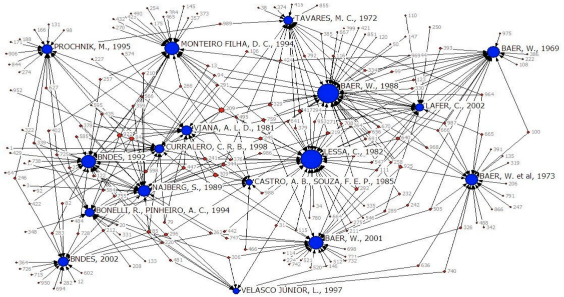



General Graphs :

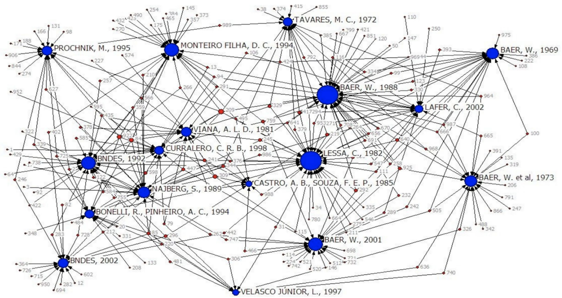
- Social Networks
- Citation Networks
- Molecules

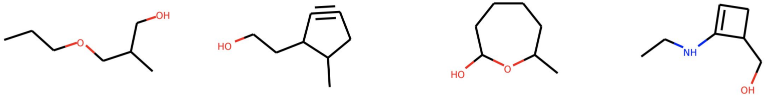


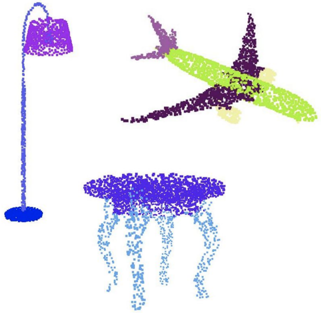
# Grid Data vs. General Graphs

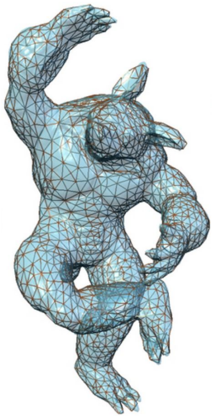


Lots of real-world applications need to deal with Non-Grid data



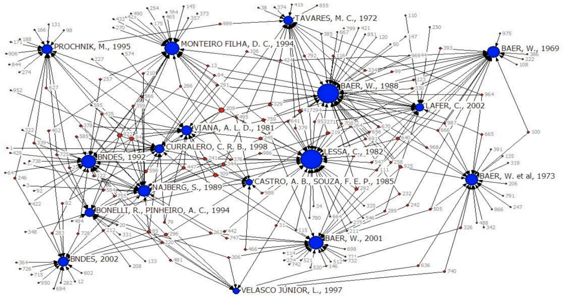
General Graphs :

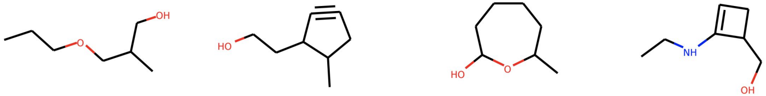
- Social Networks
- Citation Networks
- Molecules
- Point Clouds
- 3D  Meshes
- ...

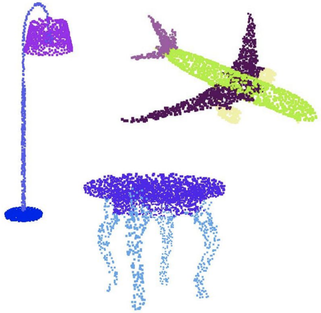
# Grid Data vs. General Graphs

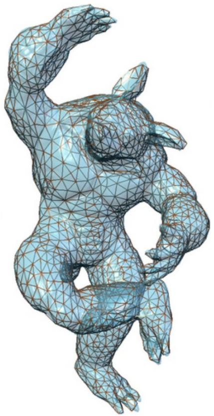
DeepGCNs.org

Lots of real-world applications need to deal with Non-Grid data



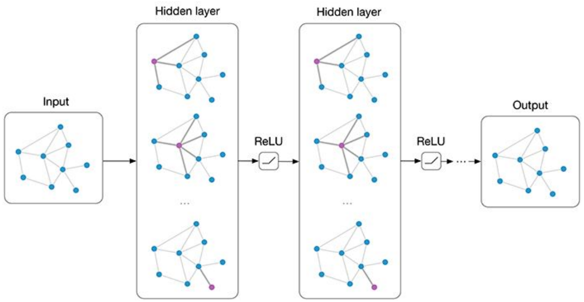




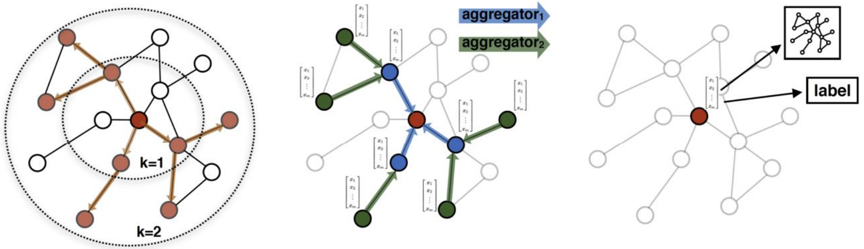
General Graphs:

- Social Networks
- Citation Networks
- Molecules
- Point Clouds
- 3D Meshes
- ...

CNN doesn’t work
GCN to rescue

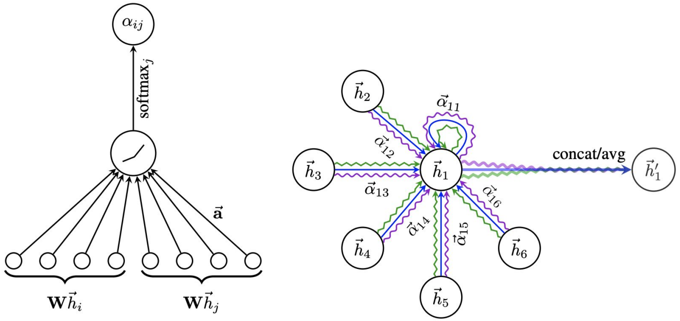



Kipf, T.N. and Welling, M., 2016. Semi-Supervised Classification with Graph Convolutional Networks.

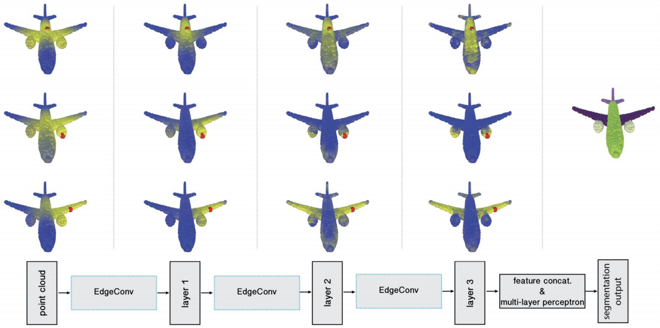



Hamilton, W.L., Ying, R. and Leskovec, J., 2017. Inductive Representation Learning on Large Graphs.

# Most of SOTA GCNs are not deeper than 3 or 4 layers.

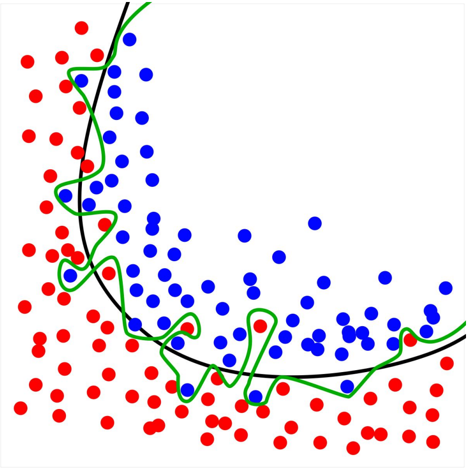



Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P. and Bengio, Y., 2018. Graph Attention Networks.

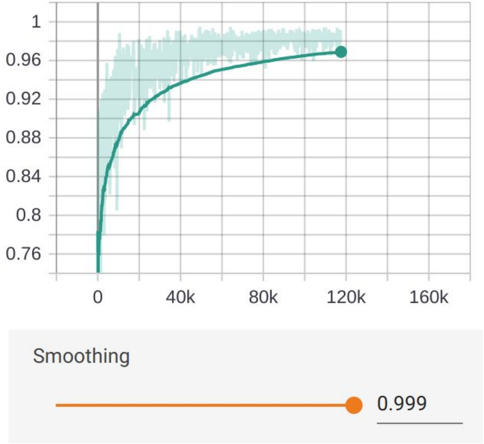



Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M. and Solomon, J.M., 2018. Dynamic Graph CNN for Learning on Point Clouds.

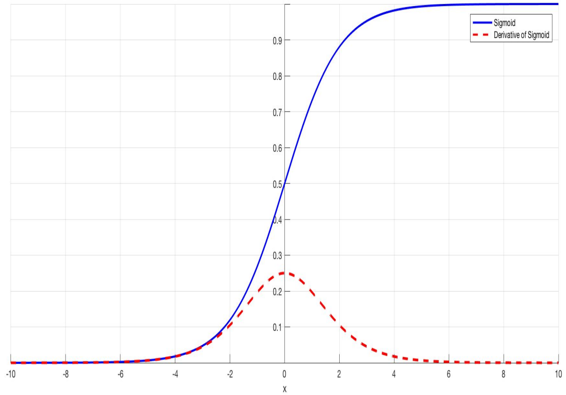
# Why GCNs are limited to shallow structures?



Over-fitting




Over-smoothing




Vanishing Gradient

Figures from https://towardsdatascience.com/the-vanishing-gradient-problem-69bf08b15484

DeepGCNs.org

# DeepGCNs for Representation Learning on Graphs

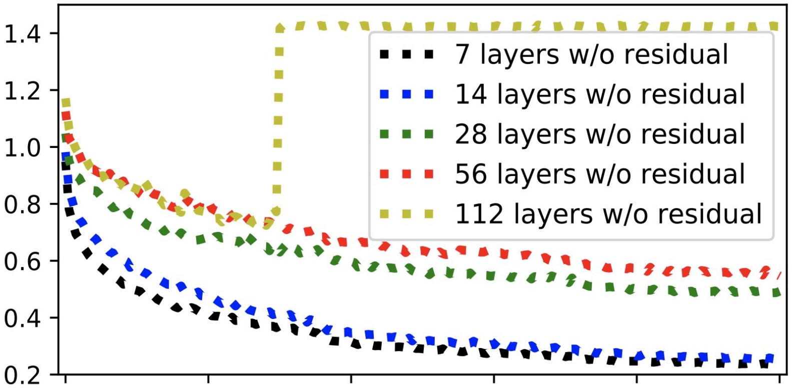
- DeepGCNs: Can GCNs Go as Deep as CNNs? (ICCV'2019, TPAMI)
- DeeperGCN: All You Need to Train Deeper GCNs (arXiv'2020)
- Training Graph Neural Networks with 1000 Layers (ICML'2021)

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

IVUL Guohao Li

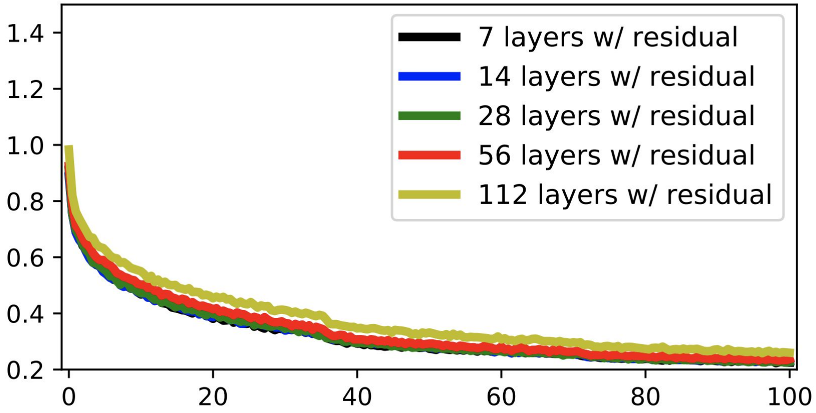
# Training Loss of GCNs with varying depth

Deeper GCNs don't converge well.




PlainGCNs

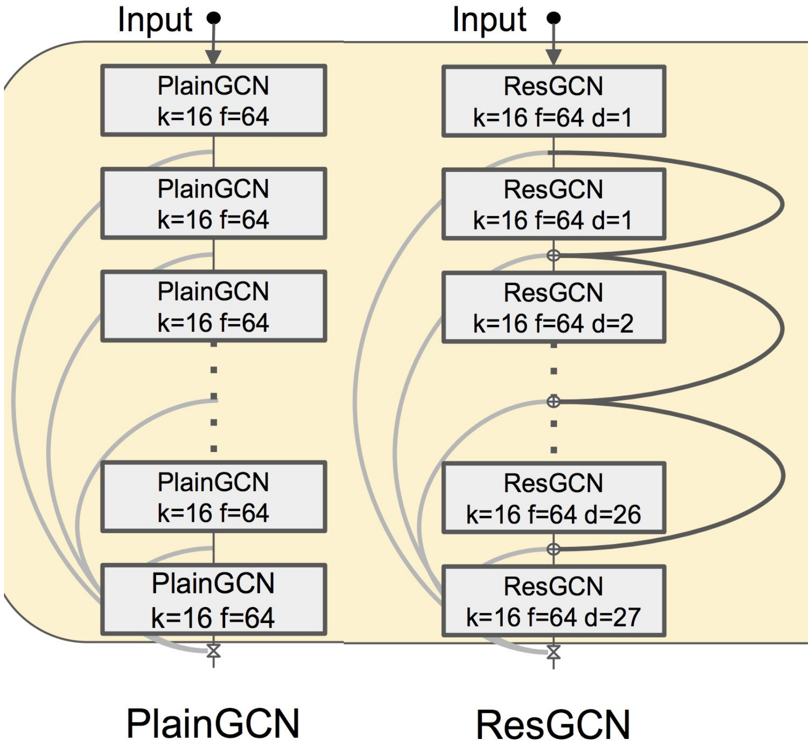
Even a 112-layer deep GCN converges well!!!




ResGCNs

# Residual Graph Connections





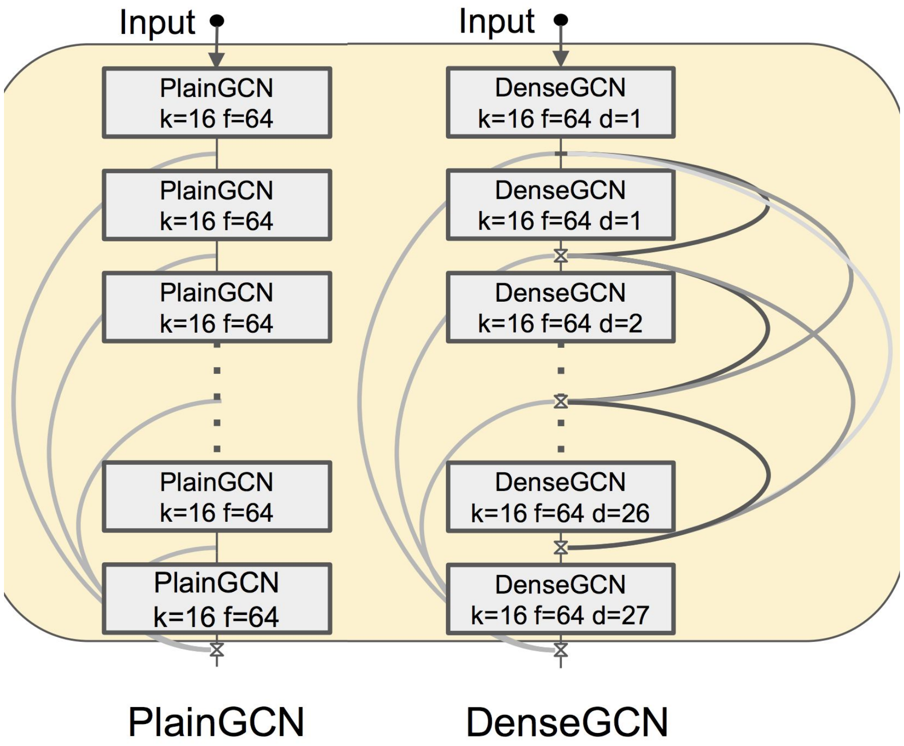

$$\begin{aligned}\mathcal{G}_{l+1} &= \mathcal{H}(\mathcal{G}_l, \mathcal{W}_l) \\ &= \mathcal{F}(\mathcal{G}_l, \mathcal{W}_l) + \mathcal{G}_l.\end{aligned}$$

An example: ResMRGCN

| | |
|---|---|
| $h^{res}_{\mathcal{N}^{(d)}(v_l)} = max\left(\{h_{u_l} - h_{v_l} \mid u_l \in \mathcal{N}^{(d)}(v_l)\}\right),$ | Aggregate |
| $h^{res}_{v_{l+1}} = mlp\left(concat\left(h_{v_l}, h^{res}_{\mathcal{N}^{(d)}(v_l)}\right)\right),$ | Update |
| $h_{v_{l+1}} = h^{res}_{v_{l+1}} + h_{v_l}.$ | Skip connection |

# Dense Graph Connections

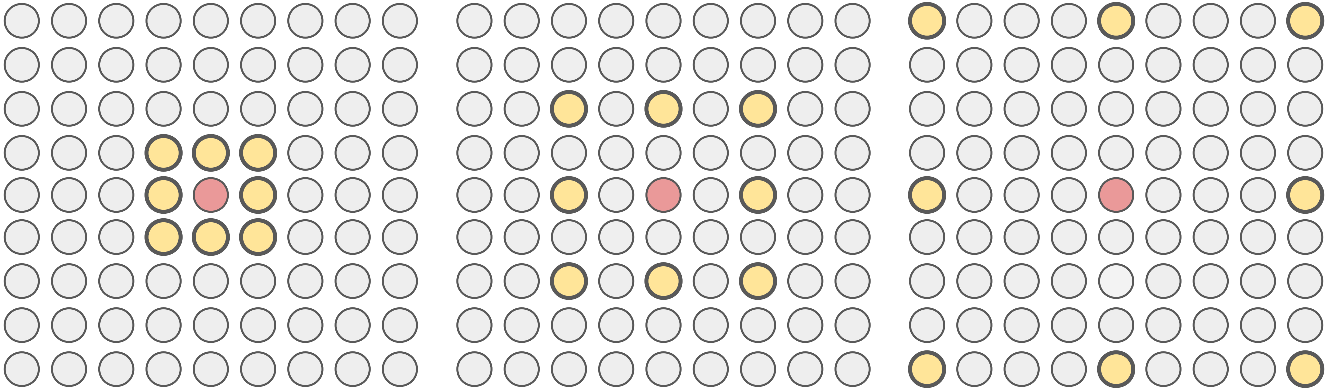
DeepGCNs.org



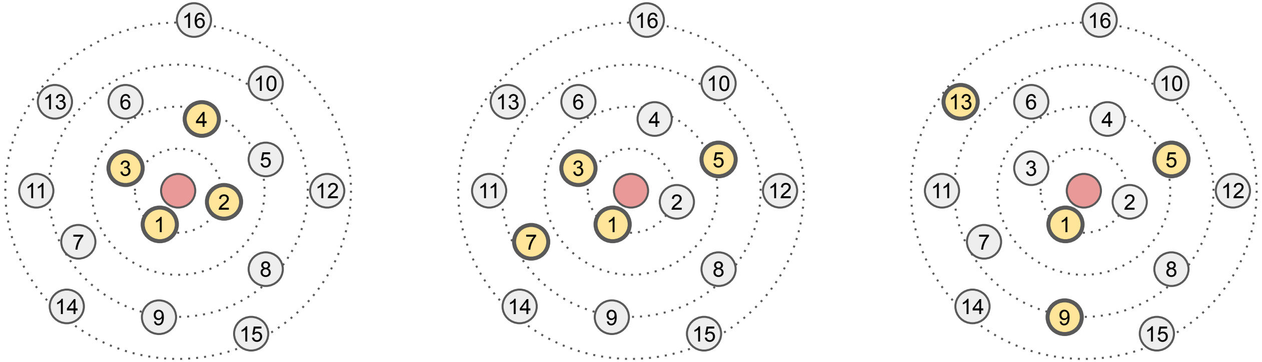

$$
\begin{aligned}
\mathcal{G}_{l+1} &= \mathcal{H}(\mathcal{G}_l, \mathcal{W}_l) \\
&= \mathcal{T}(\mathcal{F}(\mathcal{G}_l, \mathcal{W}_l), \mathcal{G}_l) \\
&= \mathcal{T}(\mathcal{F}(\mathcal{G}_l, \mathcal{W}_l), ..., \mathcal{F}(\mathcal{G}_0, \mathcal{W}_0), \mathcal{G}_0).
\end{aligned}
$$

# Dilated Graph Convolutions

DeepGCNs.org

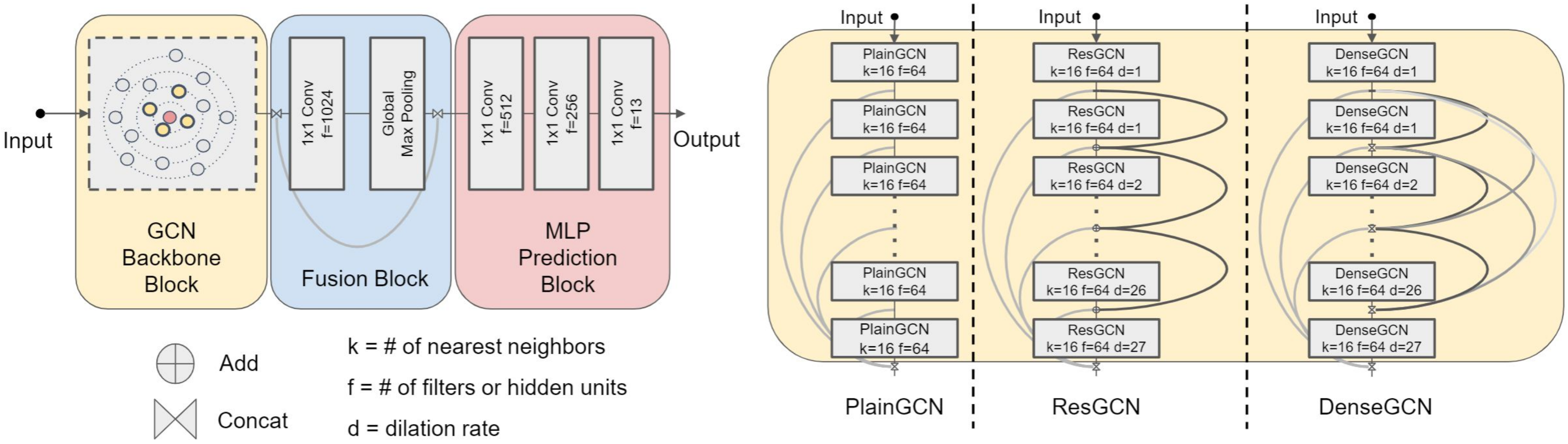


Dilated Convolution on a regular graph, e.g. 2D image



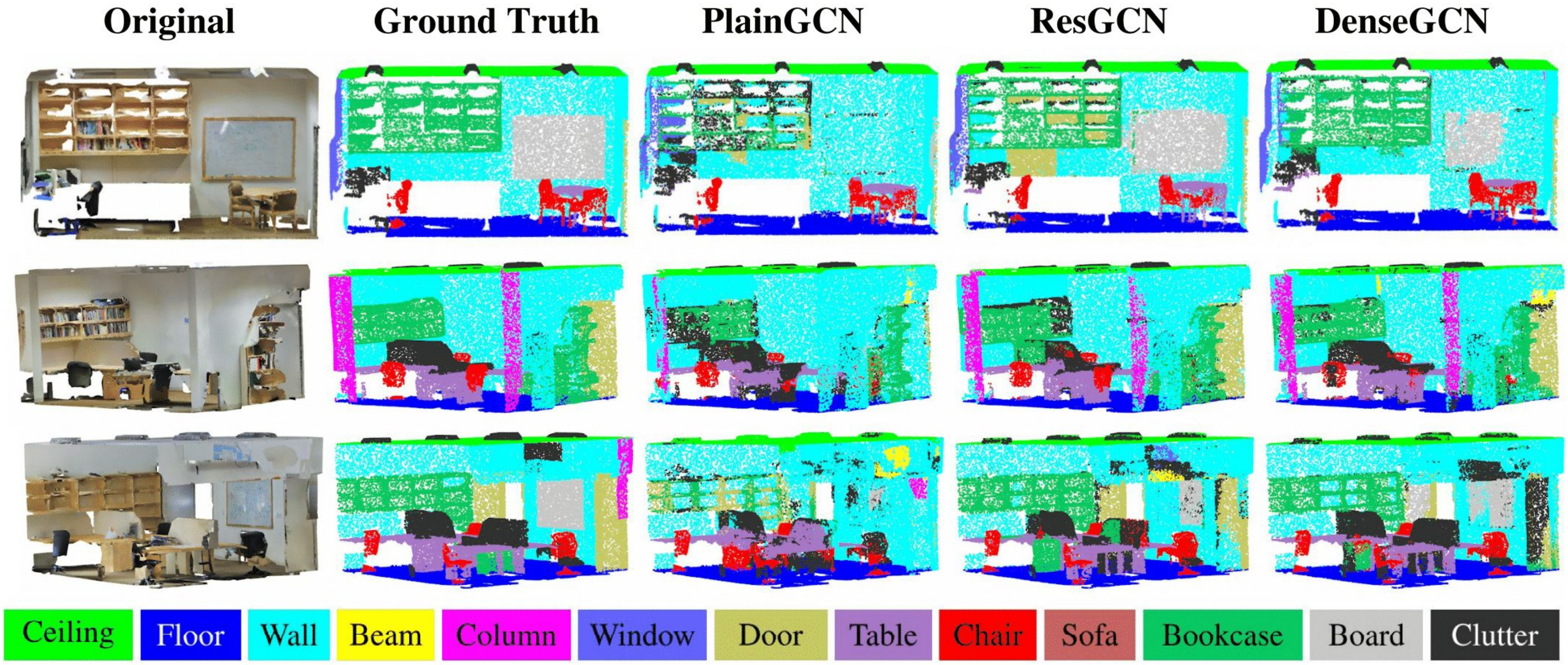
Dilated graph Convolution on an irregular graph, e.g. 3D point cloud

# Deep Graph Convolutional Networks (DeepGCNs)



# Graph Learning on 3D Point Clouds

DeepGCNs.org



# We outperform other SOTA in 9 out of 13 classes

| **Method** | **OA** | **mIOU** | **ceiling** | **floor** | **wall** | **beam** | **column** | **window** | **door** | **table** | **chair** | **sofa** | **bookcase** | **board** | **clutter** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PointNet [29] | 78.5 | 47.6 | 88.0 | 88.7 | 69.3 | 42.4 | 23.1 | 47.5 | 51.6 | 54.1 | 42.0 | 9.6 | 38.2 | 29.4 | 35.2 |
| MS+CU [8] | 79.2 | 47.8 | 88.6 | **95.8** | 67.3 | 36.9 | 24.9 | 48.6 | 52.3 | 51.9 | 45.1 | 10.6 | 36.8 | 24.7 | 37.5 |
| G+RCU [8] | 81.1 | 49.7 | 90.3 | 92.1 | 67.9 | **44.7** | 24.2 | 52.3 | 51.2 | 58.1 | 47.4 | 6.9 | 39.0 | 30.0 | 41.9 |
| PointNet++ [31] | - | 53.2 | 90.2 | 91.7 | 73.1 | 42.7 | 21.2 | 49.7 | 42.3 | 62.7 | 59.0 | 19.6 | 45.8 | 48.2 | 45.6 |
| 3DRNN+CF [51] | **86.9** | 56.3 | 92.9 | 93.8 | 73.1 | 42.5 | 25.9 | 47.6 | 59.2 | 60.4 | **66.7** | 24.8 | **57.0** | 36.7 | 51.6 |
| DGCNN [43] | 84.1 | 56.1 | - | - | - | - | - | - | - | - | - | - | - | - | - |
| **ResGCN-28 (*Ours*)** | 85.9 | **60.0** | **93.1** | 95.3 | **78.2** | 33.9 | **37.4** | **56.1** | **68.2** | **64.9** | 61.0 | **34.6** | 51.5 | **51.1** | **54.4** |

Table 1. Comparison of ResGCN-28 with state-of-the-art.

Consistent improvements across all the classes.

| Class | DGCNN [6] | ResGCN-28 (*Ours*) |
|---|---|---|
| ceiling | 92.7 | **93.1** |
| floor | 93.6 | **95.3** |
| wall | 77.5 | **78.2** |
| beam | 32.0 | **33.9** |
| column | 36.3 | **37.4** |
| window | 52.5 | **56.1** |
| door | 63.7 | **68.2** |
| table | 61.1 | **64.9** |
| chair | 60.2 | **61.0** |
| sofa | 20.5 | **34.6** |
| bookcase | 47.7 | **51.5** |
| board | 42.7 | **51.1** |
| clutter | 51.5 | **54.4** |
| **mIOU** | 56.3 | **60.0** |

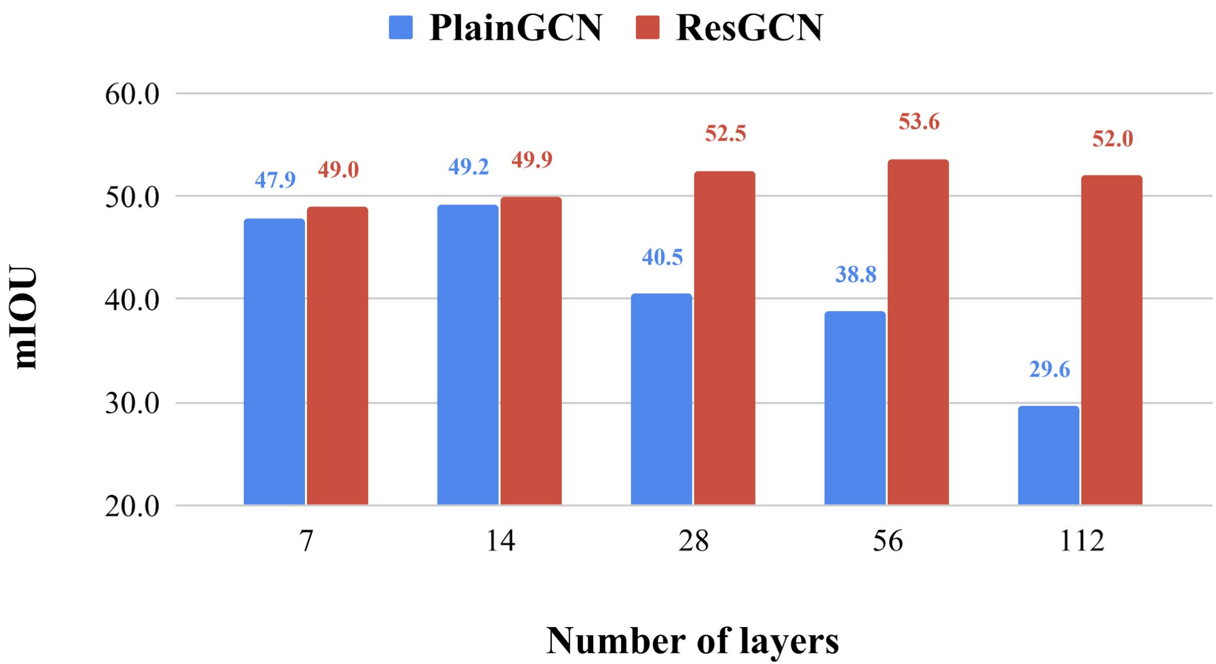
~ 4% boost in mIOU.

Table 2. Comparison of ResGCN-28 with DGCNN* (Our shallow baseline model).

* We reproduced the results of DGCNN on all classes since the results across all classes were not provided in the DGCNN paper.

# PlainGCN VS. ResGCN

DeepGCNs.org



# Application in Biology

By John Morris.

Wider

Deeper

| Number of filters | 32 | 64 | 128 | 256 |
|---|---|---|---|---|
| *PlainMRGCN-3* | 95.84 | 97.60 | 98.58 | 99.13 |
| *PlainMRGCN-7* | 97.35 | 98.69 | 99.22 | **99.38** |
| *PlainMRGCN-14* | 97.55 | 99.02 | 99.31 | 99.34 |
| *PlainMRGCN-28* | 98.09 | 99.00 | 99.02 | 99.31 |
| *PlainMRGCN-56* | 92.70 | 97.43 | 97.31 | 97.61 |
| *PlainMRGCN-112* | 60.75 | 71.97 | 89.69 | 91.50 |
| *ResMRGCN-3* | 96.04 | 97.60 | 98.53 | 99.09 |
| *ResMRGCN-7* | 97.00 | 98.43 | 99.19 | 99.30 |
| *ResMRGCN-14* | 97.75 | 98.88 | 99.26 | **99.38** |
| *ResMRGCN-28* | 98.50 | 99.16 | 99.29 | **99.41** |
| *ResMRGCN-56* | 98.62 | 99.27 | **99.36** | **99.40** |
| *ResMRGCN-112* | 98.41 | 99.34 | **99.38** | **99.39** |
| *DenseMRGCN-3* | 95.96 | 97.85 | 98.66 | 99.11 |
| *DenseMRGCN-7* | 97.87 | 98.47 | 99.31 | **99.36** |
| *DenseMRGCN-14* | 98.93 | 99.00 | 99.01 | **99.43** |
| *DenseMRGCN-28* | 99.16 | 99.29 | **99.42** | - |
| *DenseMRGCN-56* | 99.22 | - | - | - |

Table 2. Node classification of biological networks.

# Application in Biology



| Model | m-F1 score (%) |
|---|---|
| GraphSAGE [42] | 61.20 |
| GATConv [43] | 97.30 |
| VR-GCN [57] | 97.80 |
| GaAN [58] | 98.71 |
| GeniePath [59] | 98.50 |
| Cluster-GCN [56] | 99.36 |
| ***ResMRGCN-28 (Ours)*** | **99.41** |
| ***DenseMRGCN-14 (Ours)*** | **99.43** |

Table 3. Comparison of DeepGCNs with state-of-the-art on PPI node classification.

DeepGCNs.org

# DeepGCNs for Representation Learning on Graphs

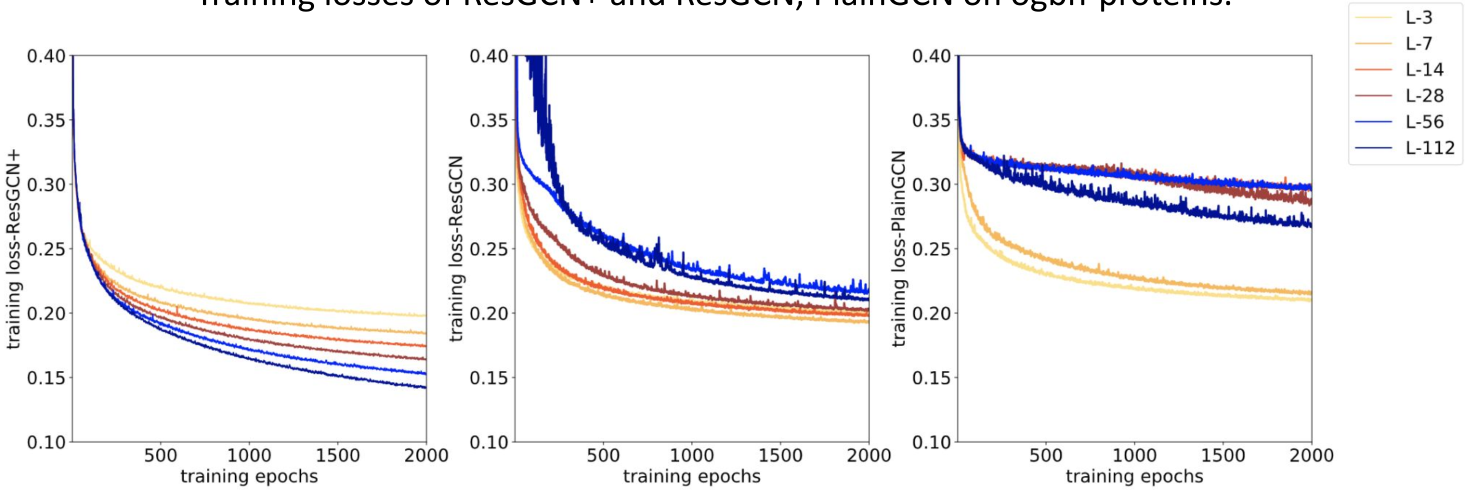
- DeepGCNs: Can GCNs Go as Deep as CNNs? (ICCV’2019, TPAMI)
- DeeperGCN: All You Need to Train Deeper GCNs (arXiv’2020)
- Training Graph Neural Networks with 1000 Layers (ICML’2021)

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

IVUL Guohao Li

# DeeperGCN - Residual Connections

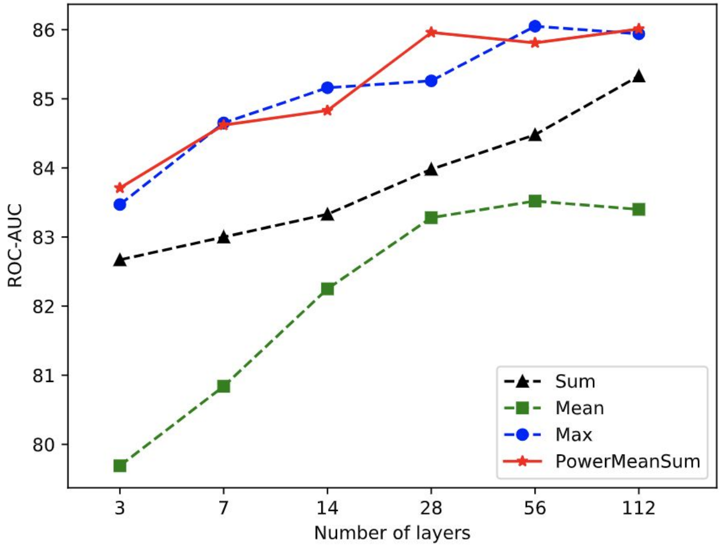
Training losses of ResGCN+ and ResGCN, PlainGCN on ogbn-proteins.



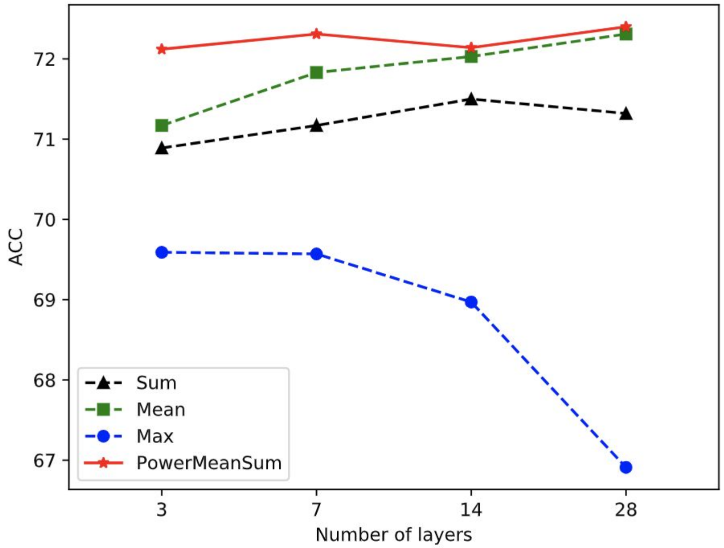

Preactivated residual connections work better.

# DeeperGCN - Aggregation Functions

**DeepGCNs.org**




**(a) different aggregators on the obgn-protein dataset.**




**(b) different aggregations on the obgn-arxiv dataset.**

Aggregation functions perform very differently on different datasets.

# DeeperGCN - Message Passing

Node Features

Neighbor's Features

Edge Features

$$\mathbf{x}_i^{(k)} = \gamma^{(k)} \left( \mathbf{x}_i^{(k-1)}, \square_{j \in \mathcal{N}(i)} \, \phi^{(k)} \left( \mathbf{x}_i^{(k-1)}, \mathbf{x}_j^{(k-1)}, \mathbf{e}_{i,j} \right) \right),$$

**Differentiable (±Learnable) Function e.g., MLPs**

**Permutation Invariant Function e.g., sum, mean or max**

**Differentiable (±Learnable) Function e.g., MLPs**

$\mathcal{N}_i$ : neighbor indices

By https://pytorch-geometric.readthedocs.io

# DeeperGCN - Aggregation Functions

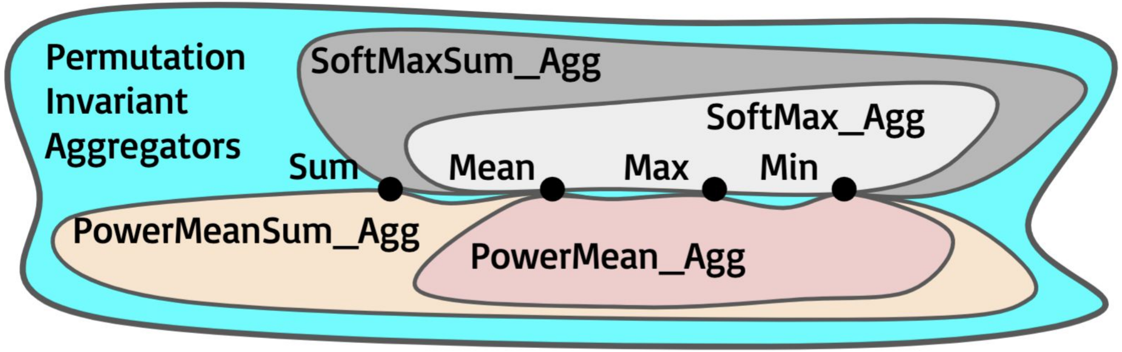



Illustration of Generalized Message Aggregation Functions

Generalized mean-max aggregation function:

$$SoftMax\_Agg_{\beta}(\cdot) = \sum_{u \in \mathcal{N}(v)} \frac{\exp(\beta \mathbf{m}_{vu})}{\sum_{i \in \mathcal{N}(v)} \exp(\beta \mathbf{m}_{vi})} \cdot \mathbf{m}_{vu}.$$

$$PowerMean\_Agg_{p}(\cdot) = \left(\frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} \mathbf{m}_{vu}^{p}\right)^{1/p}.$$

Generalized mean-max-sum aggregation function:

$$\left|\mathcal{N}(v)\right|^{y} \cdot \zeta_{x}(\cdot)$$

Differentiable aggregation functions

# DeeperGCN - Results

DeepGCNs.org

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | GraphSAGE | GCN | GaAN | | | | Ours |
| ogbn-proteins | 77.68 ± 0.20 | 72.51 ± 0.35 | 78.03 ± 0.73 | | | | **86.16 ± 0.16** |
| | GraphSAGE | GCN | GaAN | GCNII | JKNet | DAGNN | |
| ogbn-arxiv | 71.49 ± 0.27 | 71.74 ± 0.29 | 71.97 ± 0.24 | **72.74 ± 0.16** | 72.19 ± 0.21 | 72.09 ± 0.25 | 72.32 ± 0.27 |
| | GraphSAGE | GCN | ClusterGCN | GraphSAINT | GAT | | |
| ogbn-products | 78.29 ± 0.16 | 75.64 ± 0.21 | 78.97 ± 0.33 | 80.27 ± 0.26 | 79.45 ± 0.59 | | **81.64 ± 0.30** |
| | GIN | GCN | GIN* | GCN* | HIMP | | |
| ogbg-molhiv | 75.58 ± 1.40 | 76.06 ± 0.97 | 77.07 ± 1.49 | 75.99 ± 1.19 | 78.80 ± 0.82 | | **78.87 ± 1.24** |
| ogbg-molpcba | 22.66 ± 0.28 | 20.20 ± 0.24 | 27.03 ± 0.23 | 24.24 ± 0.34 | | | **27.81 ± 0.38*** |
| ogbg-ppa | 68.92 ± 1.00 | 68.39 ± 0.84 | 70.37 ± 1.07 | 68.57 ± 0.61 | | | **77.12 ± 0.71** |
| | GraphSAGE | GCN | DeepWalk | | | | |
| ogbl-collab | 48.10 ± 0.81 | 44.75 ± 1.07 | 50.37 ± 0.34 | | | | **52.73 ± 0.47** |

Table 4. DeeperGCN achieves SOTA results on 6 OGB datasets.

# DeeperGCN - Results

DeepGCNs.org

## Leaderboard for ogbn-products

The classification accuracy on the test and validation sets. The higher, the better.

Package: >=1.1.1

| Rank | Method | Test Accuracy | Validation Accuracy | Contact | References | #Params | Hardware | Date |
|---|---|---|---|---|---|---|---|---|
| 1 | **MLP + C&S** | 0.8418 ± 0.0007 | 0.9147 ± 0.0009 | Horace He (Cornell) | Paper, Code | 96,247 | GeForce RTX 2080 (11GB GPU) | Oct 27, 2020 |
| 2 | **Linear + C&S** | 0.8301 ± 0.0001 | 0.9134 ± 0.0001 | Horace He (Cornell) | Paper, Code | 10,763 | GeForce RTX 2080 (11GB GPU) | Oct 27, 2020 |
| 3 | **UniMP** | 0.8256 ± 0.0031 | 0.9308 ± 0.0017 | Yunsheng Shi (PGL team) | Paper, Code | 1,475,605 | Tesla V100 (32GB) | Sep 8, 2020 |
| 4 | **Plain Linear + C&S** | 0.8254 ± 0.0003 | 0.9103 ± 0.0001 | Horace He (Cornell) | Paper, Code | 4,747 | GeForce RTX 2080 (11GB GPU) | Oct 27, 2020 |
| 5 | **DeeperGCN+FLAG** | 0.8193 ± 0.0031 | 0.9221 ± 0.0037 | Kezhi Kong | Paper, Code | 253,743 | NVIDIA Tesla V100 (32GB GPU) | Oct 20, 2020 |
| 6 | **GAT+FLAG** | 0.8176 ± 0.0045 | 0.9251 ± 0.0006 | Kezhi Kong | Paper, Code | 751,574 | GeForce RTX 2080 Ti (11GB GPU) | Oct 20, 2020 |
| 7 | GraphSAGE + C&S + node2vec | 0.8154 ± 0.0050 | 0.9238 ± 0.0006 | HuiXuan Chi | Paper, Code | 103,983 | Tesla V100 (32GB) | Apr 6, 2021 |
| 8 | **DeeperGCN** | 0.8098 ± 0.0020 | 0.9238 ± 0.0009 | Guohao Li - DeepGCNs.org | Paper, Code | 253,743 | NVIDIA Tesla V100 (32GB GPU) | Jun 28, 2020 |

## Leaderboard for ogbn-proteins

The ROC-AUC score on the test and validation sets. The higher, the better.

Package: >=1.1.1

| Rank | Method | Test ROC-AUC | Validation ROC-AUC | Contact | References | #Params | Hardware | Date |
|---|---|---|---|---|---|---|---|---|
| 1 | **UniMP+CrossEdgeFeat** | 0.8691 ± 0.0018 | 0.9258 ± 0.0009 | Yelrose (PGL Team) | Paper, Code | 1,959,984 | Tesla V100 (32GB) | Nov 24, 2020 |
| 2 | GAT | 0.8682 ± 0.0021 | 0.9194 ± 0.0003 | Yangkun Wang (DGL Team) | Paper, Code | 2,475,232 | p3.8xlarge (15GB GPU) | Nov 6, 2020 |
| 3 | **UniMP** | 0.8642 ± 0.0008 | 0.9175 ± 0.0006 | Yunsheng Shi (PGL team) | Paper, Code | 1,909,104 | Tesla V100 (32GB) | Sep 8, 2020 |
| 4 | **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 |
| 5 | **DeeperGCN** | 0.8580 ± 0.0017 | 0.9106 ± 0.0016 | Guohao Li - DeepGCNs.org | Paper, Code | 2,374,568 | NVIDIA Tesla V100 (32GB GPU) | Jun 16, 2020 |
| 6 | **DeepGCN** | 0.8496 ± 0.0028 | 0.8971 ± 0.0011 | Guohao Li - DeepGCNs.org | Paper, Code | 2,374,456 | NVIDIA Tesla V100 (32GB GPU) | Jun 20, 2020 |

DeeperGCN ranked top 1 on several datasets from Jun. 2020 to Sep 2020.

DeepGCNs.org

# DeepGCNs for Representation Learning on Graphs

- DeepGCNs: Can GCNs Go as Deep as CNNs? (ICCV'2019, TPAMI)
- DeeperGCN: All You Need to Train Deeper GCNs (arXiv'2020)
- Training Graph Neural Networks with 1000 Layers (ICML'2021)

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

IVUL Guohao Li

# 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)**

# 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:

$$X_i = X_i' - f_{w_i}(X_{i-1}', A, U), \; i \in \{2, \cdots, C\}$$
$$X_0' = \sum_{i=2}^{C} X_i$$
$$X_1 = X_1' - f_{w_1}(X_0', A, U).$$

When #group =2:

$$\langle X_1, X_2 \rangle \mapsto \langle X_1', X_2' \rangle$$

Forward:

$$X_0' = X_2$$
$$X_1' = f_{w_1}(X_0', A, U) + X_1$$
$$X_2' = f_{w_2}(X_1', A, U) + X_2$$

Inverse:

$$X_2 = X_2' - f_{w_2}(X_1', A, U)$$
$$X_0' = X_2$$
$$X_1 = X_1' - f_{w_1}(X_0', A, U).$$

# Memory Efficient GNNs

DEQ-GNN:

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

$$\frac{\partial \ell}{\partial (\cdot)} = -\frac{\partial \ell}{\partial \mathbf{z}^\star_{1:T}} \left( J^{-1}_{g_\theta} \big|_{\mathbf{z}^\star_{1:T}} \right) \frac{\partial f_\theta(\mathbf{z}^\star_{1:T}; \mathbf{x}_{1:T})}{\partial (\cdot)} = -\frac{\partial \ell}{\partial h} \frac{\partial h}{\partial \mathbf{z}^\star_{1:T}} \left( J^{-1}_{g_\theta} \big|_{\mathbf{z}^\star_{1:T}} \right) \frac{\partial f_\theta(\mathbf{z}^\star_{1:T}; \mathbf{x}_{1:T})}{\partial (\cdot)},$$

# 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

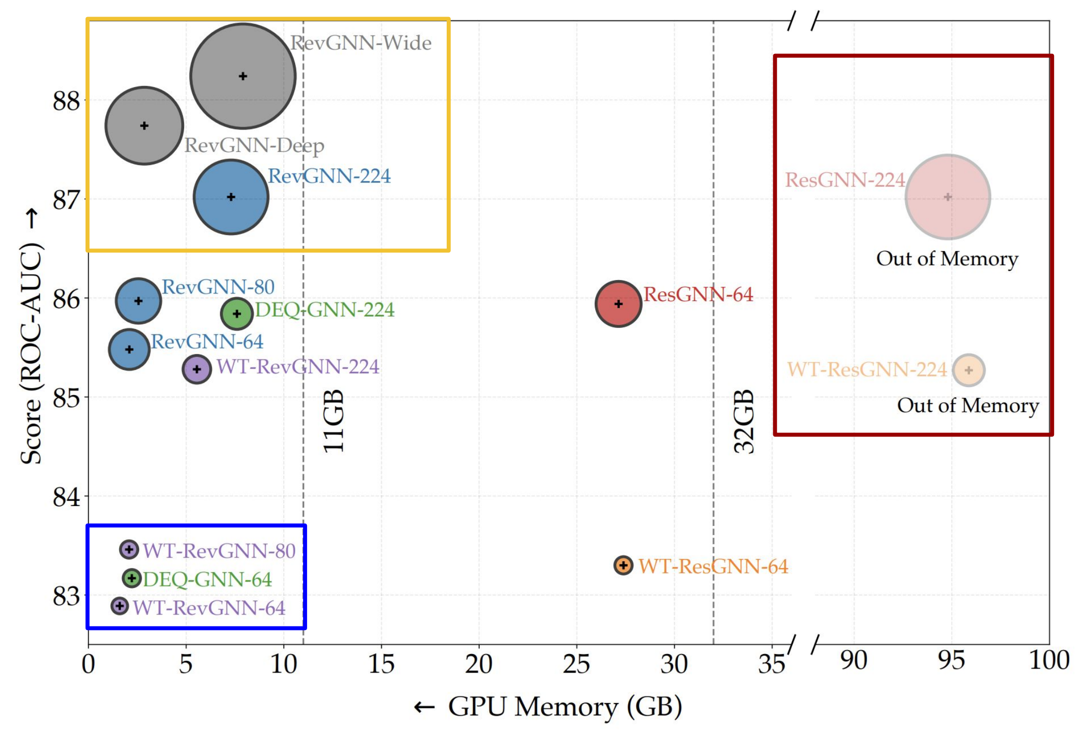
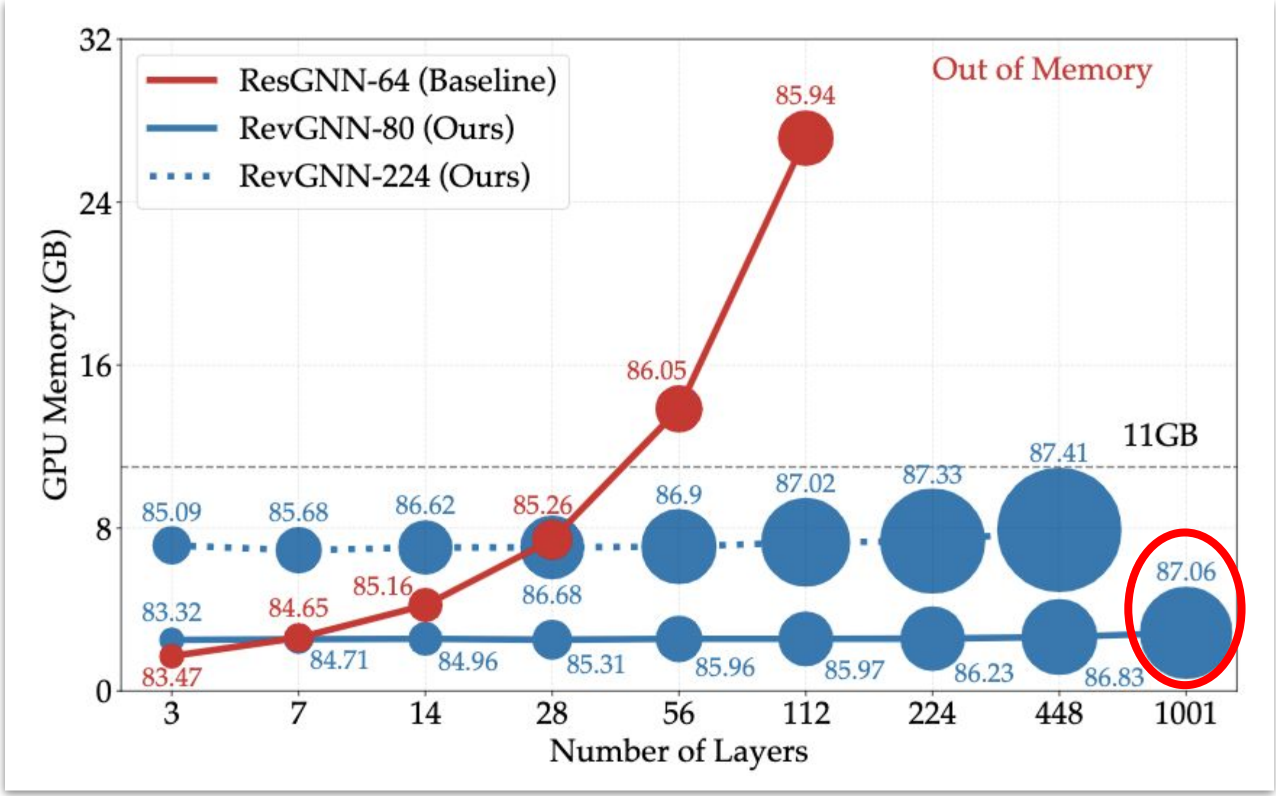



Fig. Performance versus GPU memory consumption on the ogbn-proteins dataset for 112 layer deep networks.

# 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)

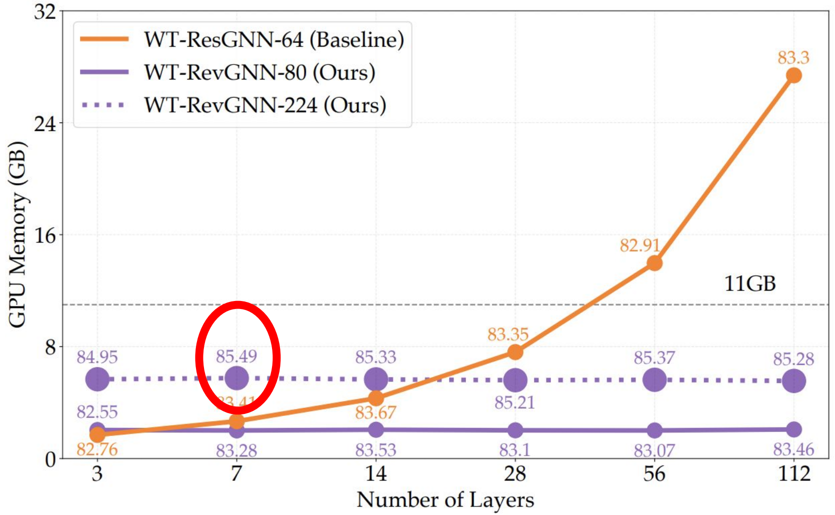
| Rank | Method | Test ROC-AUC | Validation ROC-AUC | Contact | References | #Params | Hardware | Date |
|---|---|---|---|---|---|---|---|---|
| 1 | **RevGNN-Wide** | 0.8824 ± 0.0015 | 0.9450 ± 0.0008 | Guohao Li - DeepGCNs.org | Paper, Code | 68,471,608 | NVIDIA RTX 6000 (48G) | Jun 16, 2021 |
| 2 | **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)

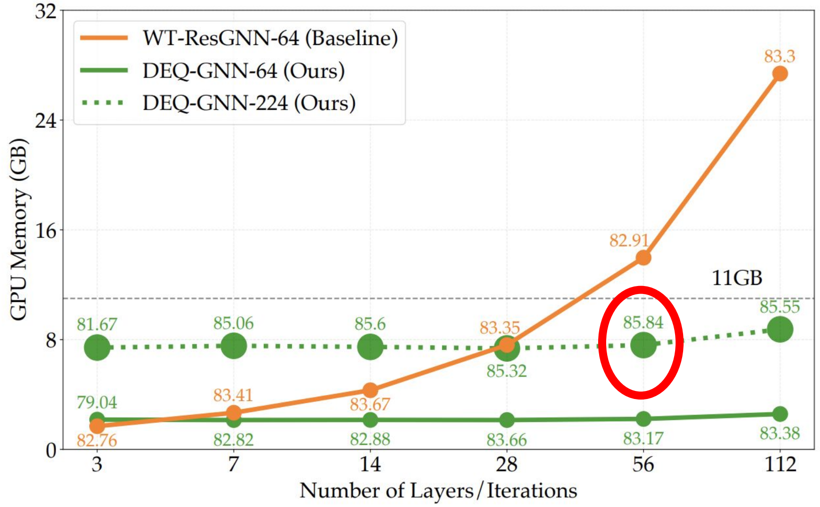
# Results: SOTA with RevGNN (ogbn-arxiv)

| 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 |

# Results: Constant Memory and Parameter Complexities




WT-RevGNN



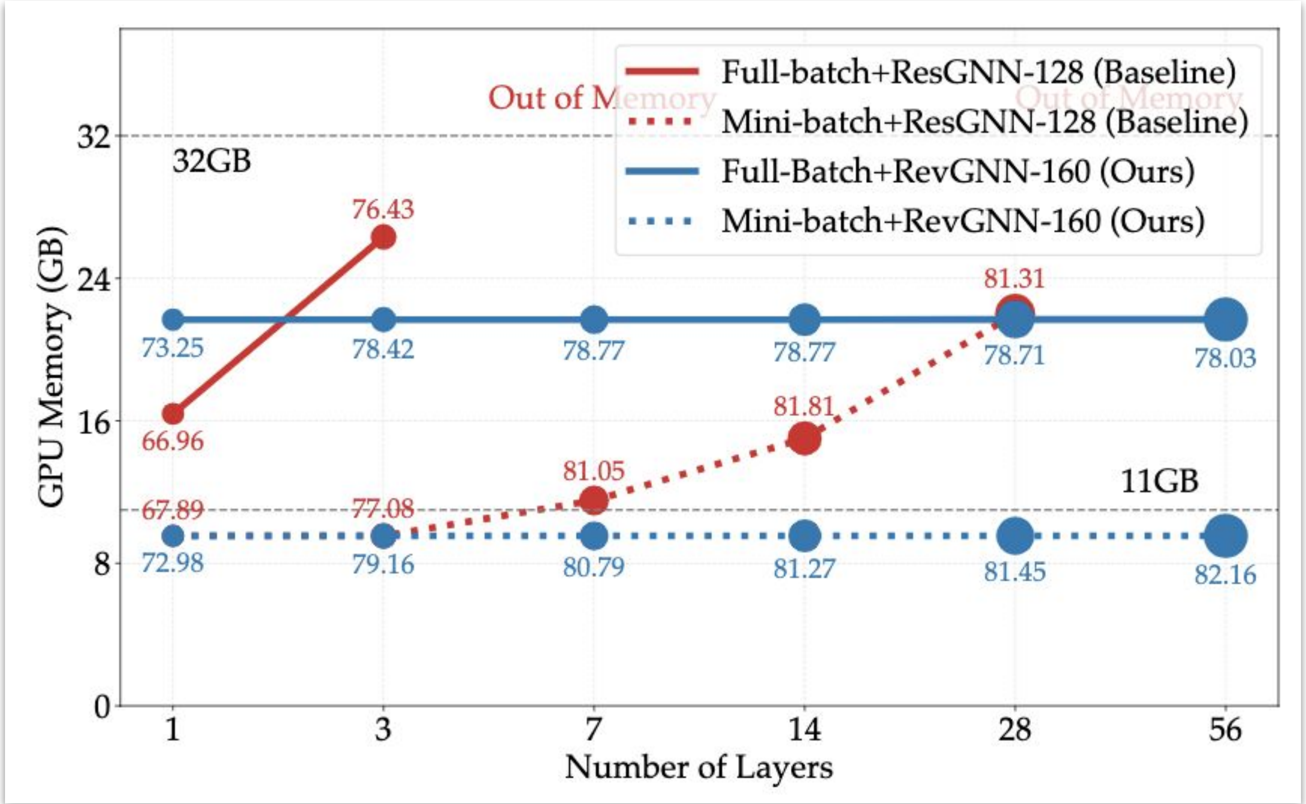

DEQ-RevGNN

# 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.

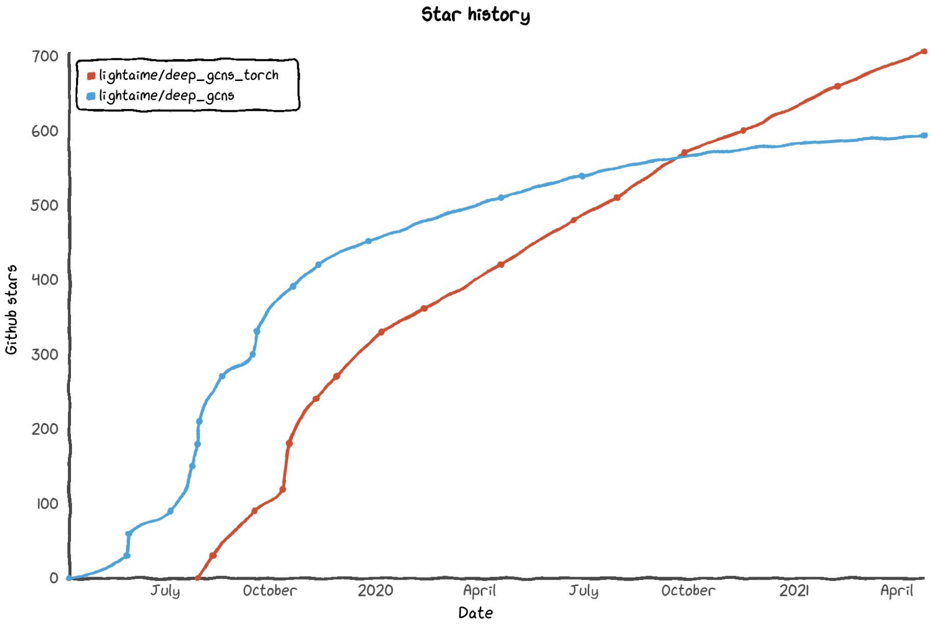
# Ablation: Mini-batch Training (ogbn-products)




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

# Open Source

DeepGCNs.org



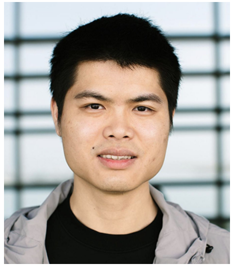

> 1300 Stars (Pytorch + Tensorflow)

PyTorch geometric

DEEP GRAPH LIBRARY

# Team

Bernard Ghanem



Guohao Li

Matthias Müller

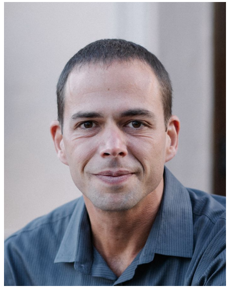
Ali Thabet

Guocheng Qian

Itzel C. Delgadillo

Abdulellah Abualshour

Chenxin Xiong



Vladlen koltun

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

DeepGCNs.org

# Thanks
# Q & A