NeurIPS MeetUp '19

DeepGCNs.org

DeepGCNs for Representation Learning on Graphs

Guohao Li

KAUST

VCC VISUAL COMPUTING CENTER IVUL

NEURIPS MEETUP '19

DeepGCNs.org

① **DeepGCNs: Can GCNs Go as Deep as CNNs? (*ICCV'2019 Oral*)**
Guohao Li*, Matthias Müller*, Ali Thabet, Bernard Ghanem

② **DeepGCNs: Making GCNs Go as Deep as CNNs (*arXiv'2019*)**
Guohao Li*, Matthias Müller*, Guocheng Qian, Itzel C. Delgadillo, Abdulellah Abualshour, Ali Thabet, Bernard Ghanem

③ **SGAS: Sequential Greedy Architecture Search (*arXiv'2019*)**
Guohao Li*, Guocheng Qian*, Itzel C. Delgadillo*, Matthias Müller, Ali Thabet, Bernard Ghanem


Guohao Li


Matthias Müller


Guocheng Qian

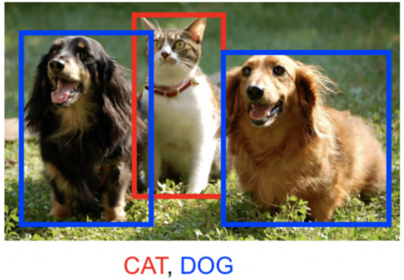

Itzel C. Delgadillo


Abdulellah Abualshour


Ali Thabet


Bernard Ghanem

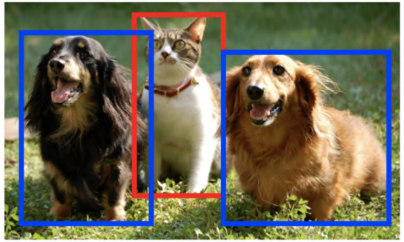
Grid data vs. General graphs



Grid Data :

- Image

Grid data vs. General graphs



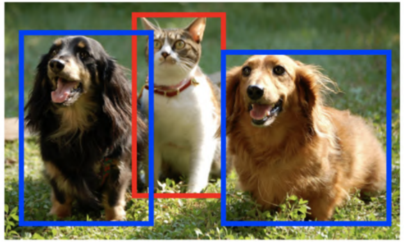
CAT, DOG

Grid Data :

- Image
- Video

Grid data vs. General graphs

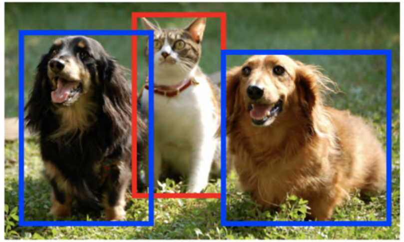
DeepGCNs.org


CAT, DOG

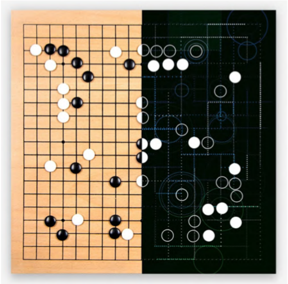
Grid Data：
- Image
- Video
- Audio
- Text

Grid data vs. General graphs

DeepGCNs.org



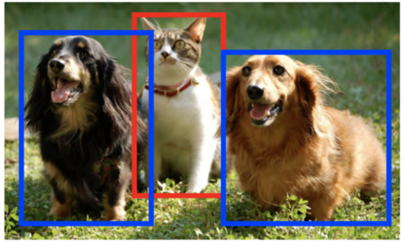
CAT, DOG



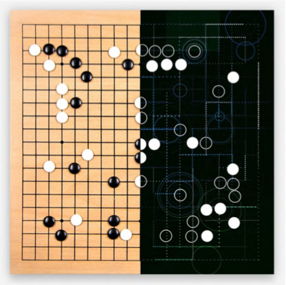
Grid Data：

- 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

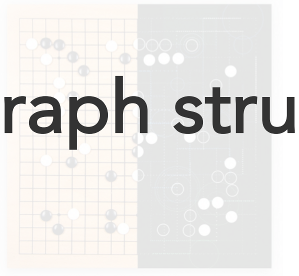
Why do we need graph convolutional networks?

Grid data vs. General graphs

DeepGCNs.org

Why we need graph convolutional networks?

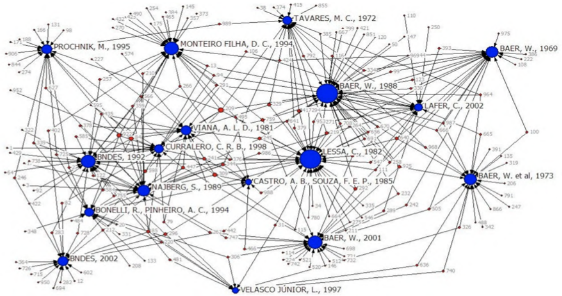
Tremendous non-grid graph structured data



Grid data vs. General graphs

DeepGCNs.org

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

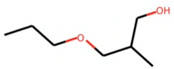


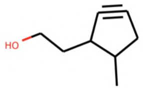
General Graphs :

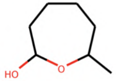
- Social Networks
- Citation Networks

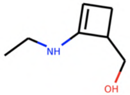
Grid data vs. General graphs

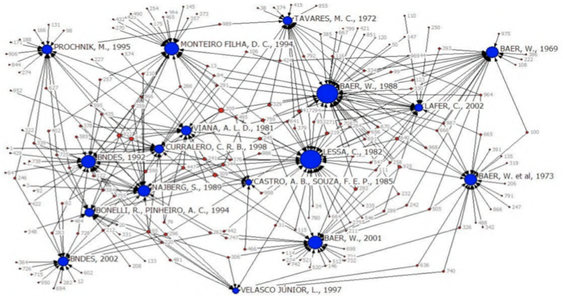
Lots of real-world applications need to deal with Non-Grid data











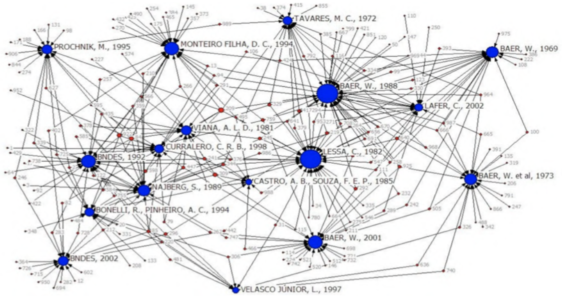
General Graphs :

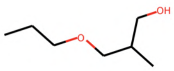
- Social Networks
- Citation Networks
- Molecules

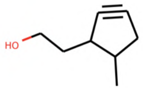
Grid data vs. General graphs

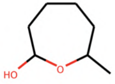
DeepGCNs.org

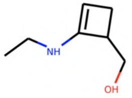
Lots of real-world applications need to deal with Non-Grid data

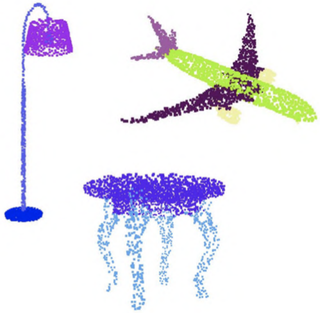


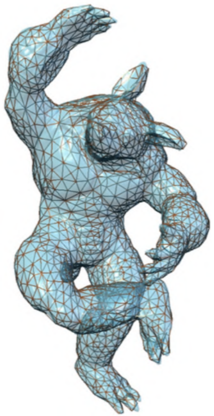










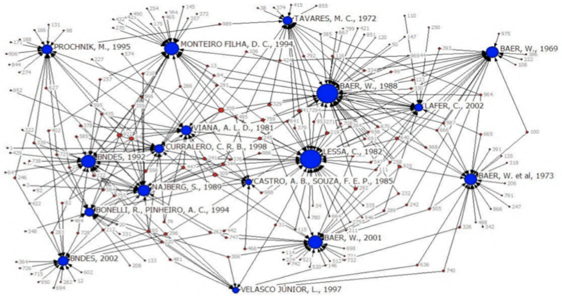


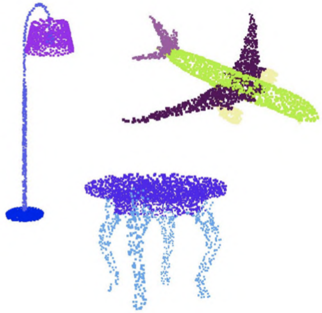
General Graphs :

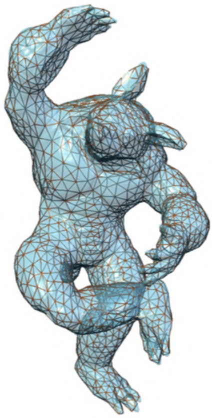
- Social Networks
- Citation Networks
- Molecules
- Point Clouds
- 3D Meshes
- ...

Grid data vs. General graphs

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







General Graphs :

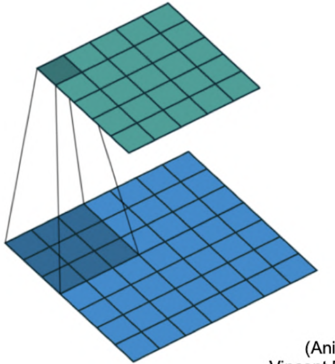
- Social Networks
- Citation Networks
- Molecules
- Point Clouds
- 3D Meshes
- ...

CNN doesn't work
GCN to rescue

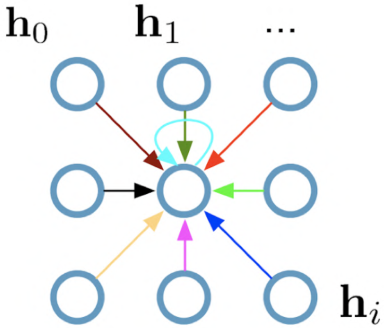
CNN vs. GCN - Recap: CNN

DeepGCNs.org

Single CNN layer with 3x3 filter:



(Animation by Vincent Dumoulin)

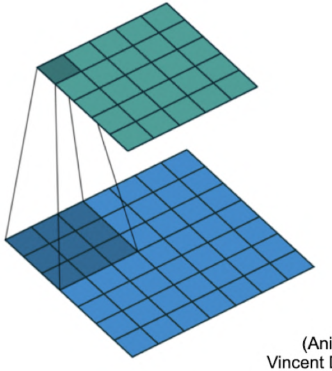


$\mathbf{h}_i \in \mathbb{R}^F$ are (hidden layer) activations of a pixel/node

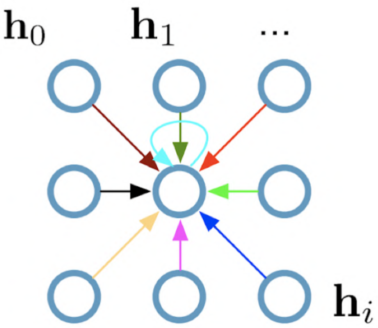
Slides by Thomas Kipf

CNN vs. GCN - Recap: CNN

Single CNN layer with 3x3 filter:



(Animation by Vincent Dumoulin)



Update for a single pixel:

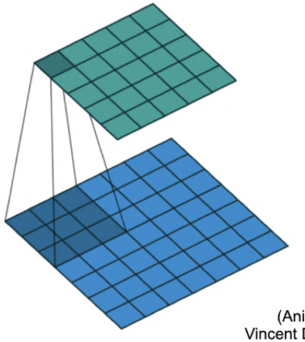
- Transform messages individually $\mathbf{W}_i \mathbf{h}_i$
- Add everything up $\sum_i \mathbf{W}_i \mathbf{h}_i$

$\mathbf{h}_i \in \mathbb{R}^F$ are (hidden layer) activations of a pixel/node

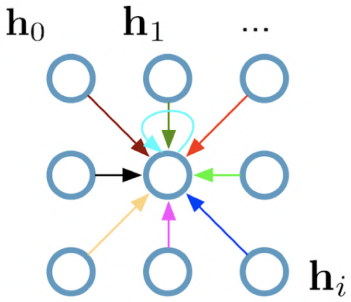
Slides by Thomas Kipf

CNN vs. GCN - Recap: CNN

Single CNN layer with 3x3 filter:



(Animation by Vincent Dumoulin)



Update for a single pixel:

- Transform messages individually $\mathbf{W}_i \mathbf{h}_i$
- Add everything up $\sum_i \mathbf{W}_i \mathbf{h}_i$

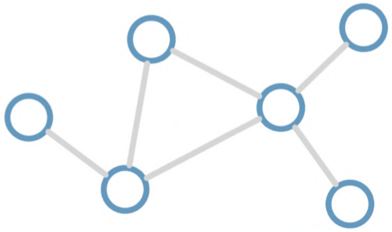
$\mathbf{h}_i \in \mathbb{R}^F$ are (hidden layer) activations of a pixel/node

Full update:

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

Slides by Thomas Kipf

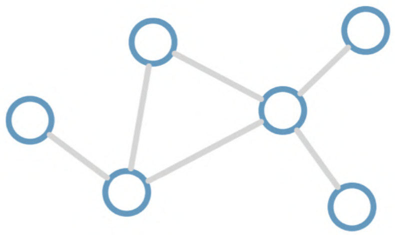
Consider this undirected graph:



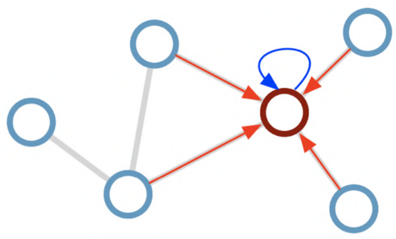
Slides by Thomas Kipf

CNN vs. GCN - Introduction: GCN

Consider this undirected graph:



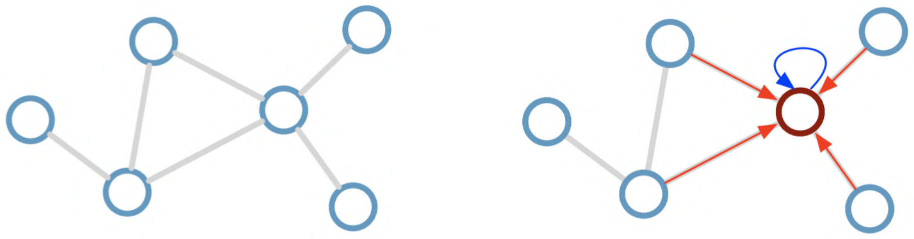
Calculate update for node in red:



Slides by Thomas Kipf

Consider this undirected graph:

Calculate update for node in red:



$\mathcal{N}_i$: neighbor indices

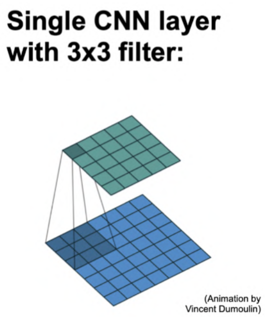
c_{ij} : norm. constant (fixed/trainable)

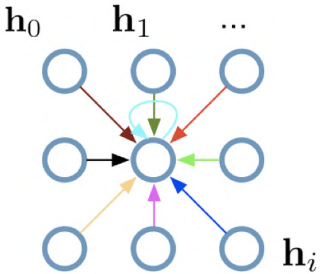
Update rule:

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

Slides by Thomas Kipf

CNN vs. GCN - Comparison



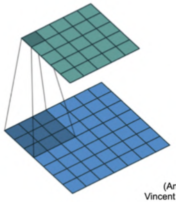


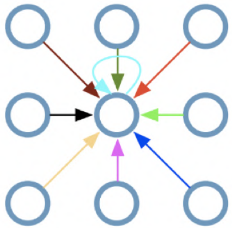
Convolutional Neural Network (CNN)

Slides by Thomas Kipf

CNN vs. GCN - Comparison

Single CNN layer with 3x3 filter:





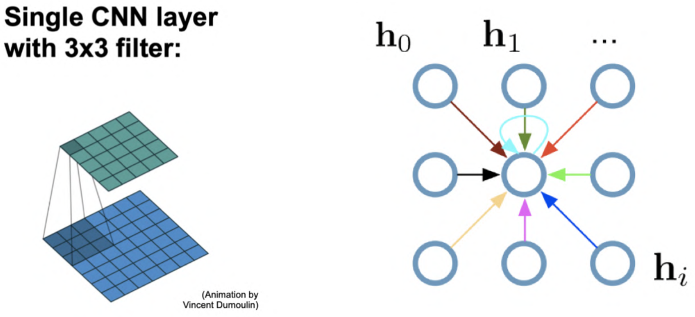
Full update:

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

Convolutional Neural Network (CNN)

Slides by Thomas Kipf

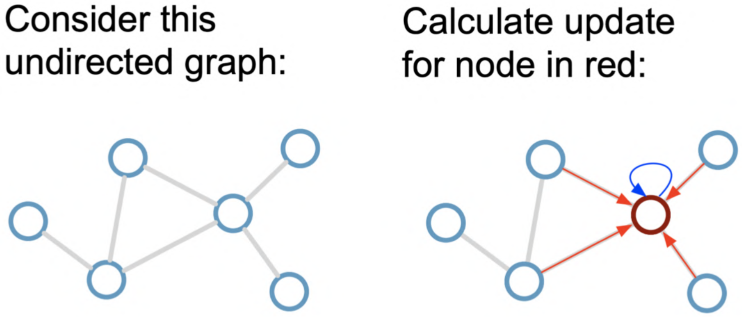
CNN vs. GCN - Comparison



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)

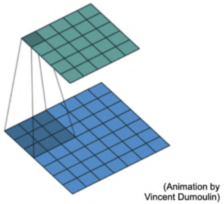


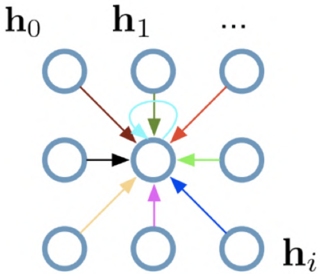
Graph Convolutional Network (GCN)

Slides by Thomas Kipf

CNN vs. GCN - Comparison

Single CNN layer with 3x3 filter:



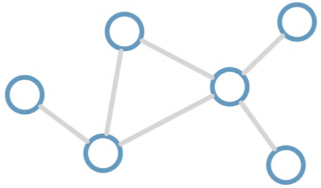


Full update:

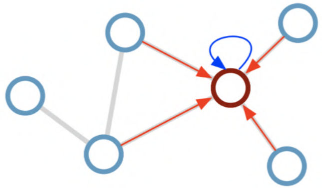
$$\mathbf{h}_4^{(l+1)} = \sigma\left(\mathbf{W}_0^{(l)}\mathbf{h}_0^{(l)} + \mathbf{W}_1^{(l)}\mathbf{h}_1^{(l)} + \cdots + \mathbf{W}_8^{(l)}\mathbf{h}_8^{(l)}\right)$$

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

CNN vs. GCN - Message Passing

Node Features

Neighbor’s Features

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

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

CNN vs. GCN - Message Passing

DeepGCNs.org

Node Features

Neighbor's Features

Edge Features

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

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

CNN vs. GCN - 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

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

CNN vs. GCN - Message Passing

DeepGCNs.org

Node Features

Neighbor's Features

Edge Features

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

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

CNN vs. GCN - Message Passing

Node Features

Neighbor's Features

Edge Features

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

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

CNN vs. GCN - Message Passing

DeepGCNs.org

Node Features

Neighbor's Features

Edge Features

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

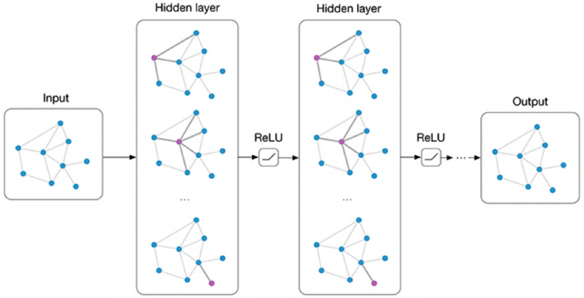
Differentiable (±Learnable) Function e.g., MLPs

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

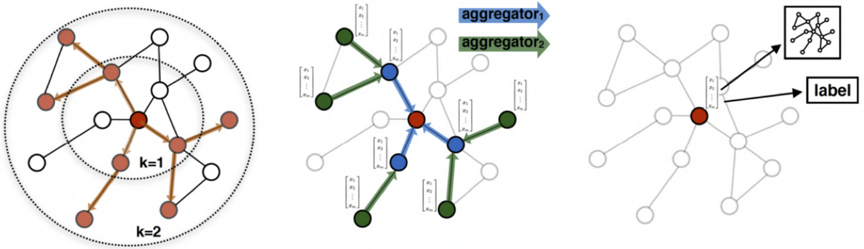
Differentiable (±Learnable) Function e.g., MLPs

$\mathcal{N}_i$: neighbor indices

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

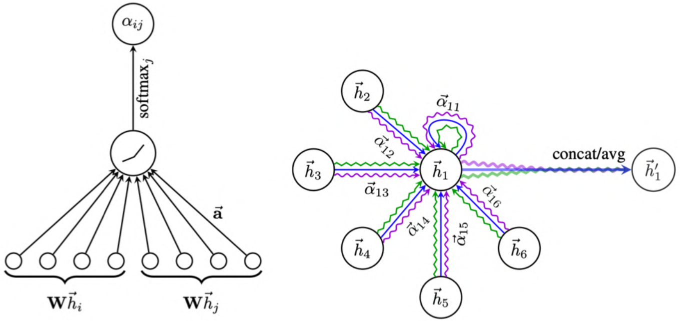


Kipf, T.N. and Welling, M., 2016. Semi-Supervised Classification with Graph Convolutional Networks.

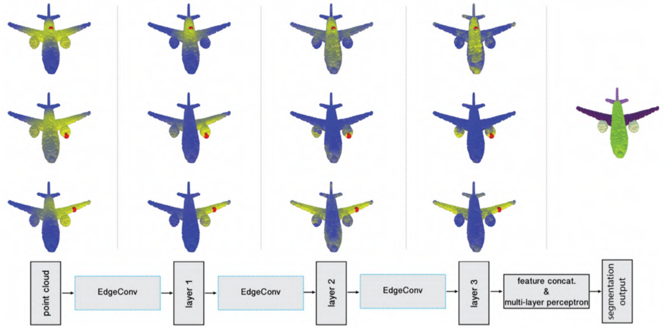


Hamilton, W.L., Ying, R. and Leskovec, J., 2017. Inductive Representation Learning on Large Graphs.

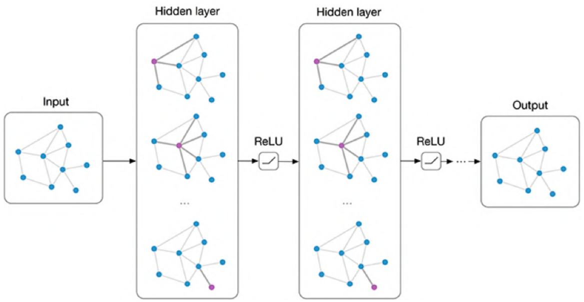
Most SOTA GCN models are no deeper than 3 or 4 layers.



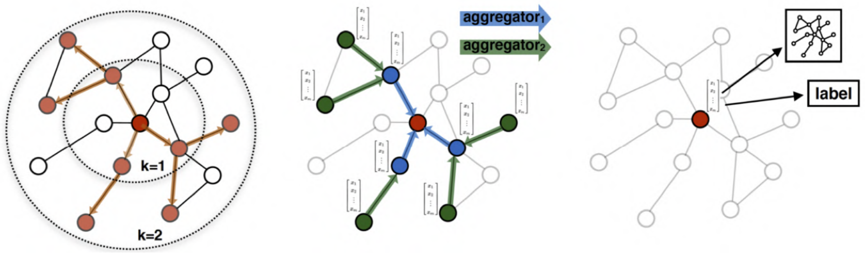
Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P. and Bengio, Y., 2018. Graph Attention Networks.



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.



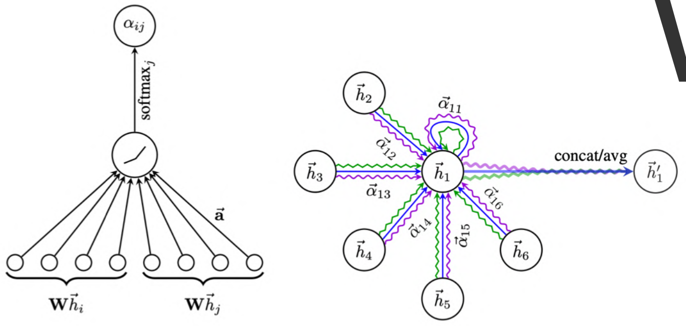
Kipf, T.N. and Welling, M., 2016. Semi-Supervised Classification with Graph Convolutional Networks.



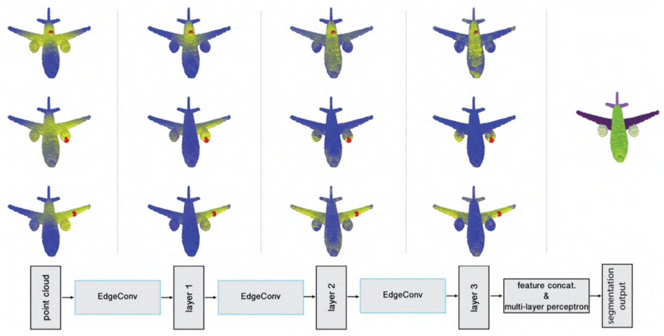
Hamilton, W.L., Ying, R. and Leskovec, J., 2017. Inductive Representation Learning on Large Graphs.

Most SOTA GCN models are no deeper than 3 or 4 layers.

Why?

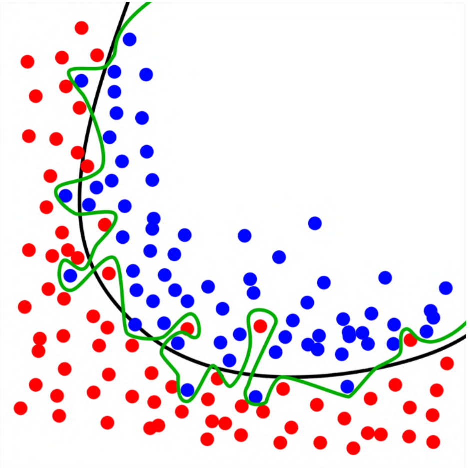


Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P. and Bengio, Y., 2018. Graph Attention Networks.

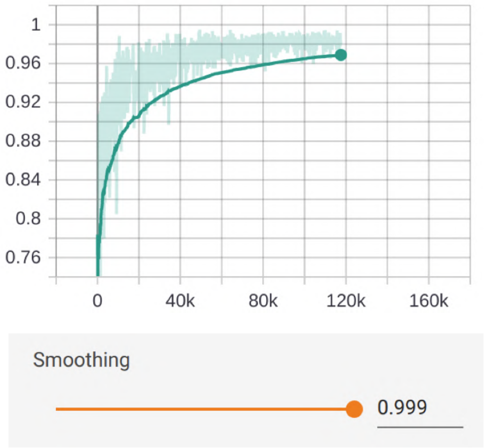


Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M. and Solomon, J.M., 2018. Dynamic Graph CNN for Learning on Point Clouds.

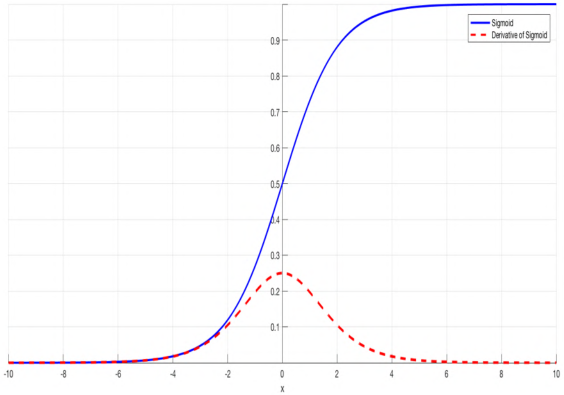
Why GCNs are limited to shallow structures?



Over-fitting



Over-smoothing

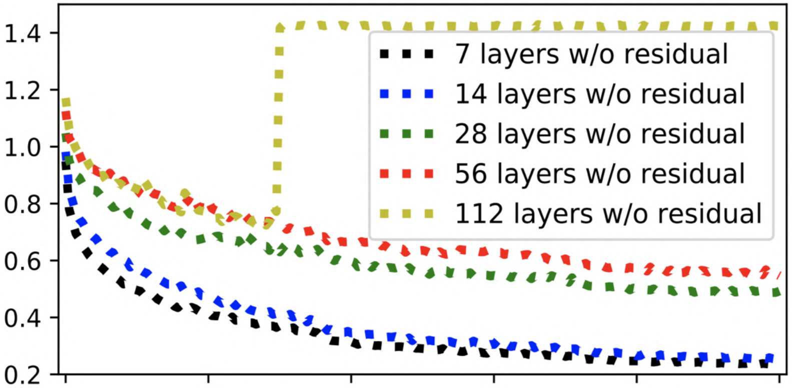


Vanishing Gradient

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

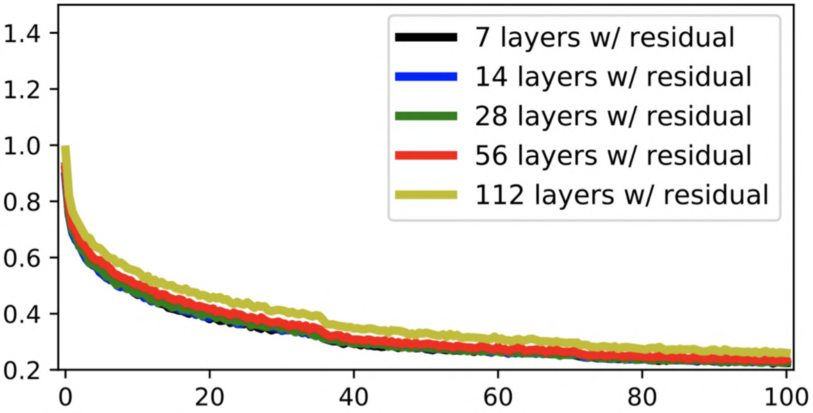
Training Loss of GCNs with varying depth

Deeper GCNs don't converge well.



PlainGCNs

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



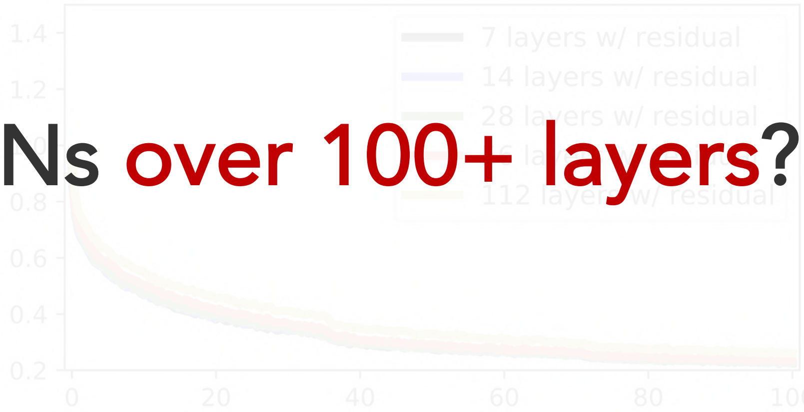
ResGCNs

Training Loss of GCNs with varying depth

Deeper GCNs don't converge well.

PlainGCNs

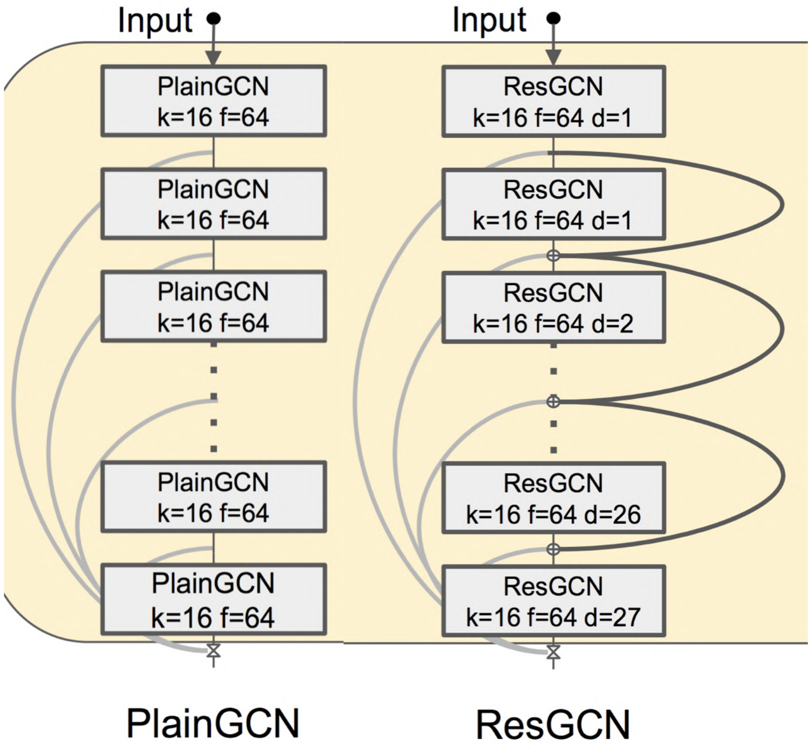
Even a 112-layer deep GCN converges well!!!



ResGCNs

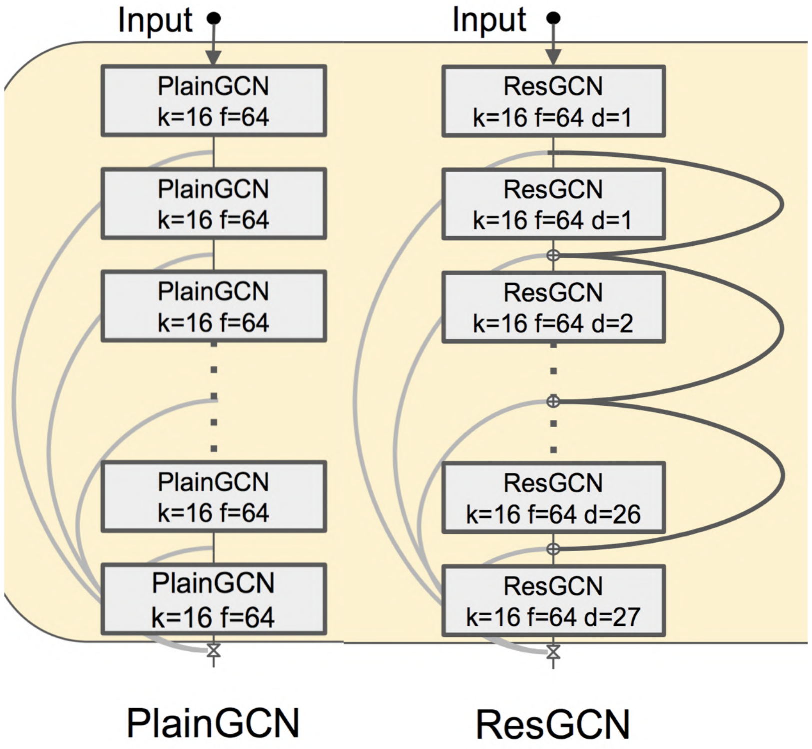
How can we train GCNs **over 100+ layers**?

Residual Graph Connections



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

Residual Graph Connections



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

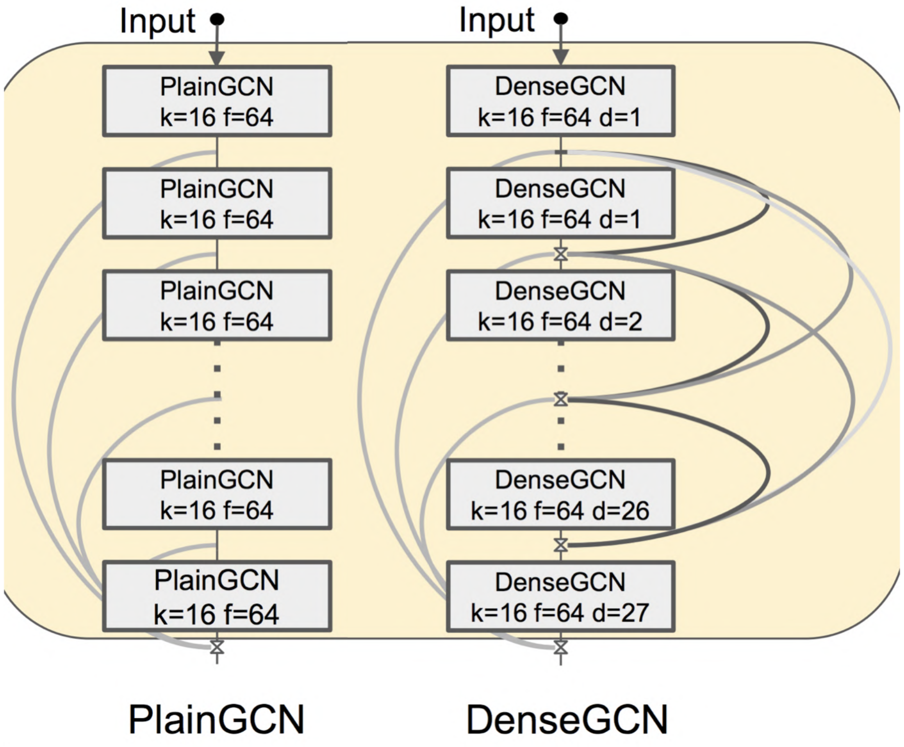
An example: ResMRGCN

$$h^{res}_{\mathcal{N}^{(d)}(v_l)} = max\left(\{h_{u_l} - h_{v_l} | u_l \in \mathcal{N}^{(d)}(v_l)\}\right), \quad \text{Aggregate}$$

$$h^{res}_{v_{l+1}} = mlp\left(concat\left(h_{v_l}, h^{res}_{\mathcal{N}^{(d)}(v_l)}\right)\right), \quad \text{Update}$$

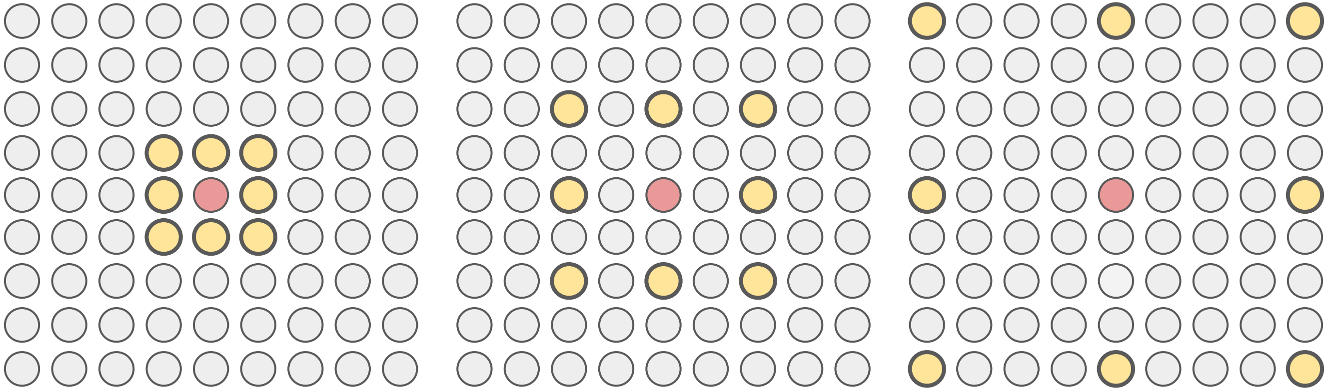
$$h_{v_{l+1}} = h^{res}_{v_{l+1}} + h_{v_l}. \quad \text{Skip connection}$$

Dense Graph Connections

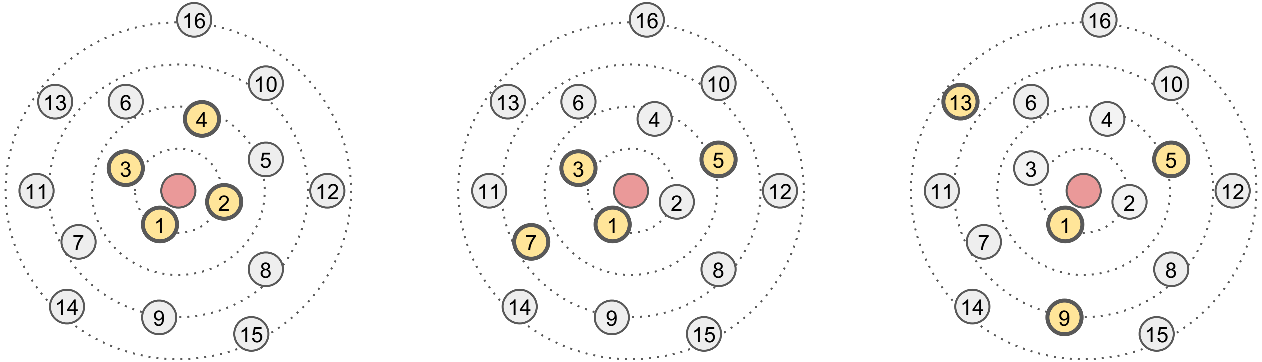


$$\begin{aligned}\mathcal{G}_{l+1} &= \mathcal{H}(\mathcal{G}_l, \mathcal{W}_l) \\ &= \mathcal{T}(\mathcal{F}(\mathcal{G}_l, \mathcal{W}_l), \mathcal{G}_l) \\ &= \mathcal{T}(\mathcal{F}(\mathcal{G}_l, \mathcal{W}_l), \ldots, \mathcal{F}(\mathcal{G}_0, \mathcal{W}_0), \mathcal{G}_0).\end{aligned}$$

Dilated Graph Convolutions



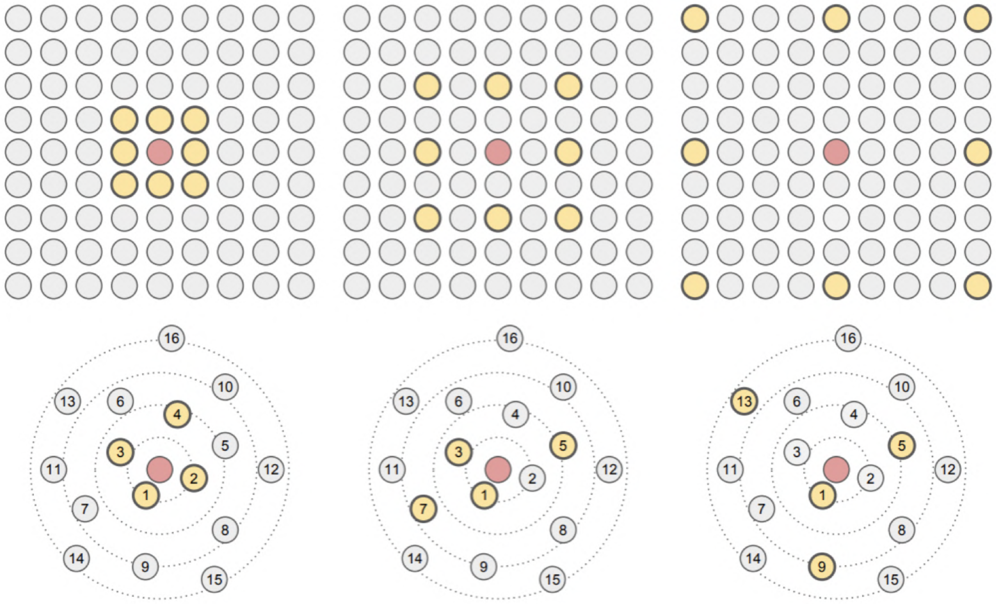
Dilated Convolution on a regular graph, e.g. 2D image



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

Dilated Graph Convolutions

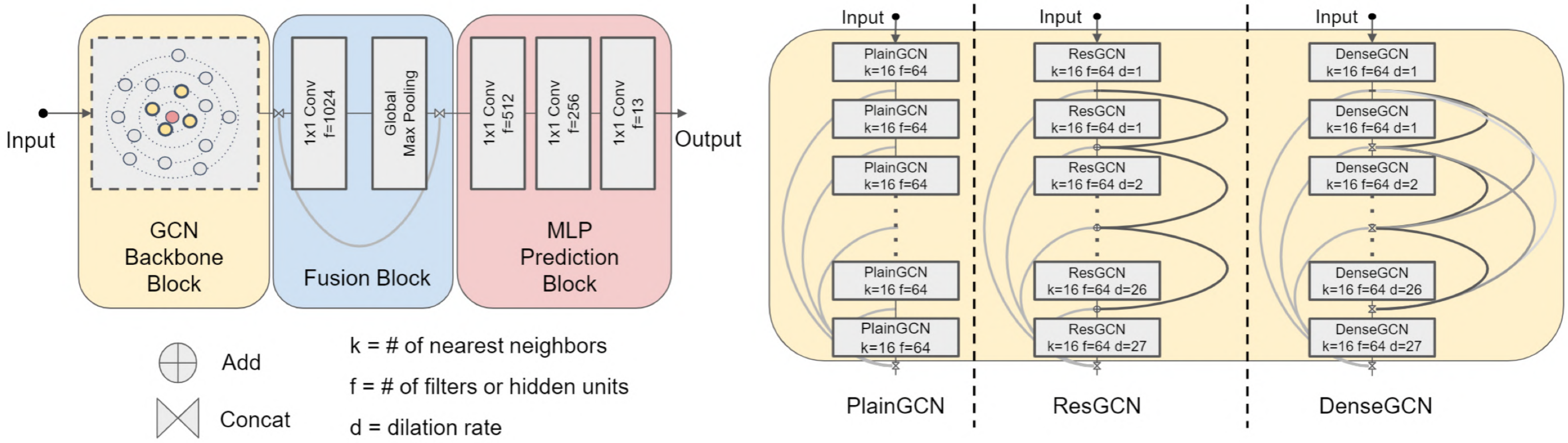
DeepGCNs.org



$$\mathcal{N}^{(d)}(v) = \{u_1, u_{1+d}, u_{1+2d}, \ldots, u_{1+(k-1)d}\}.$$

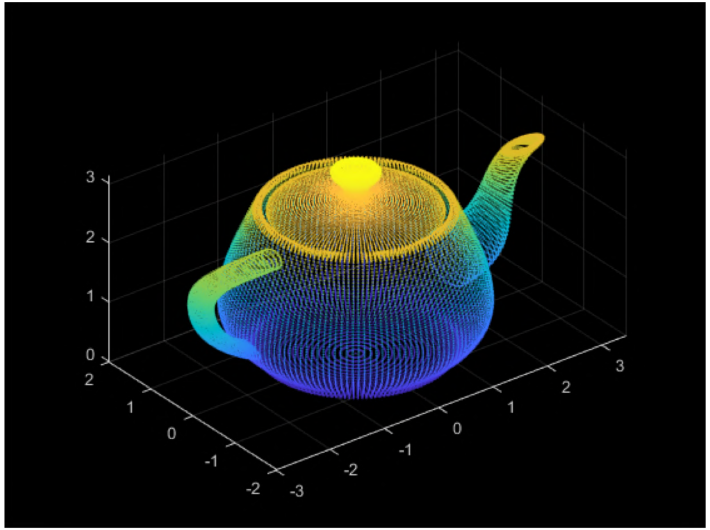
d = dilation rate

Deep Graph Convolutional Networks (GCNs)



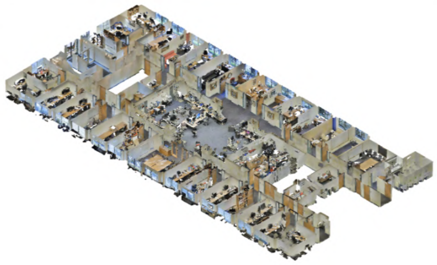
Experiments

Graph Learning on 3D Point Clouds

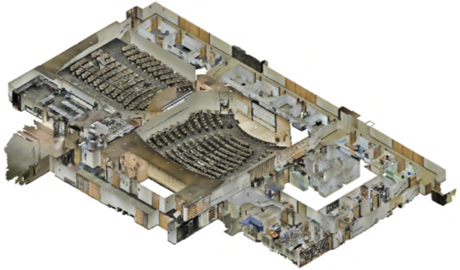


- Point clouds are unordered and irregular
- Represented by 3D coordinates and extra features such as color, surface normal, etc.
- We use k-NN to construct the directed dynamic edges between points at every GCN layer in the feature space.

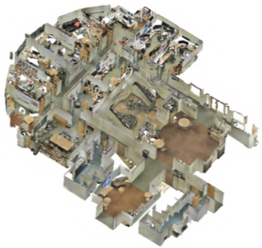
Stanford 3D Large-Scale Indoor Spaces Dataset



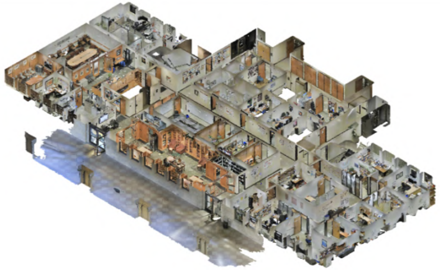
Area 1



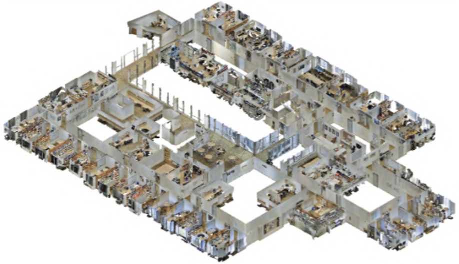
Area 2



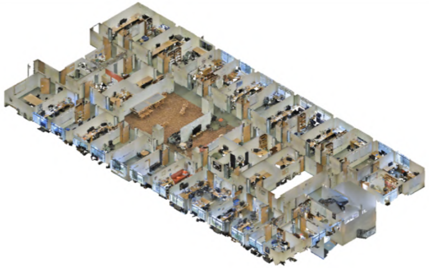
Area 3



Area 4



Area 5



Area 6

http://buildingparser.stanford.edu/dataset.html

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**

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.

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**

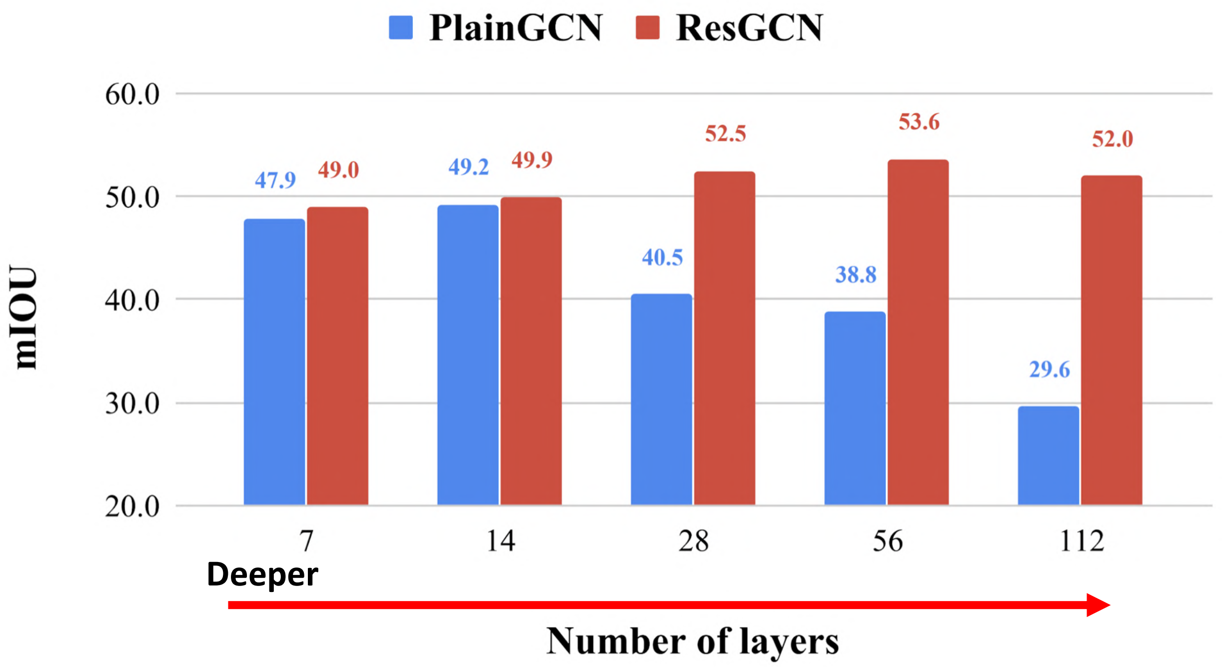
~ 4% boost in mIOU.

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

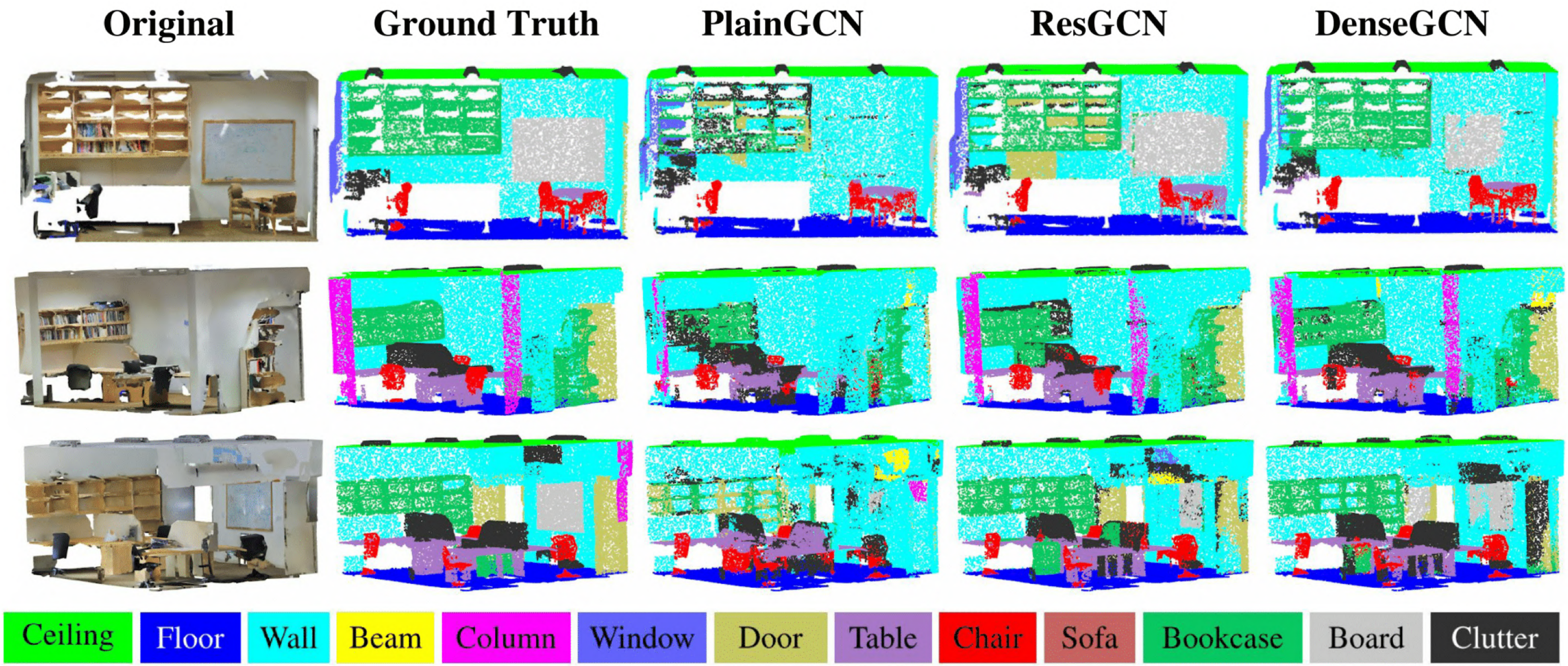
* We reproduced the results of DGCNN on all classes since the results across all classes were not provided in the DGCNN paper.

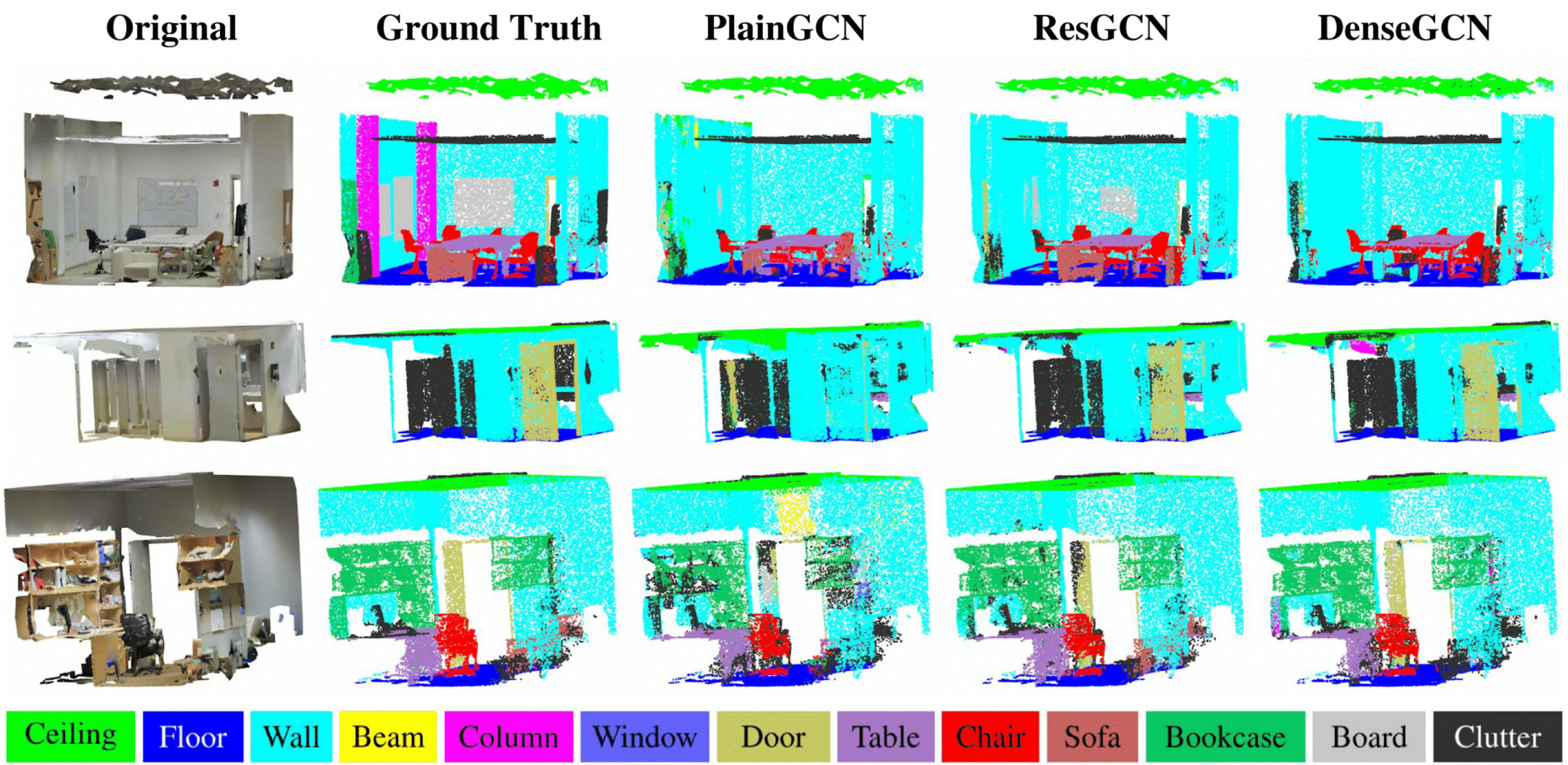
PlainGCN VS. ResGCN

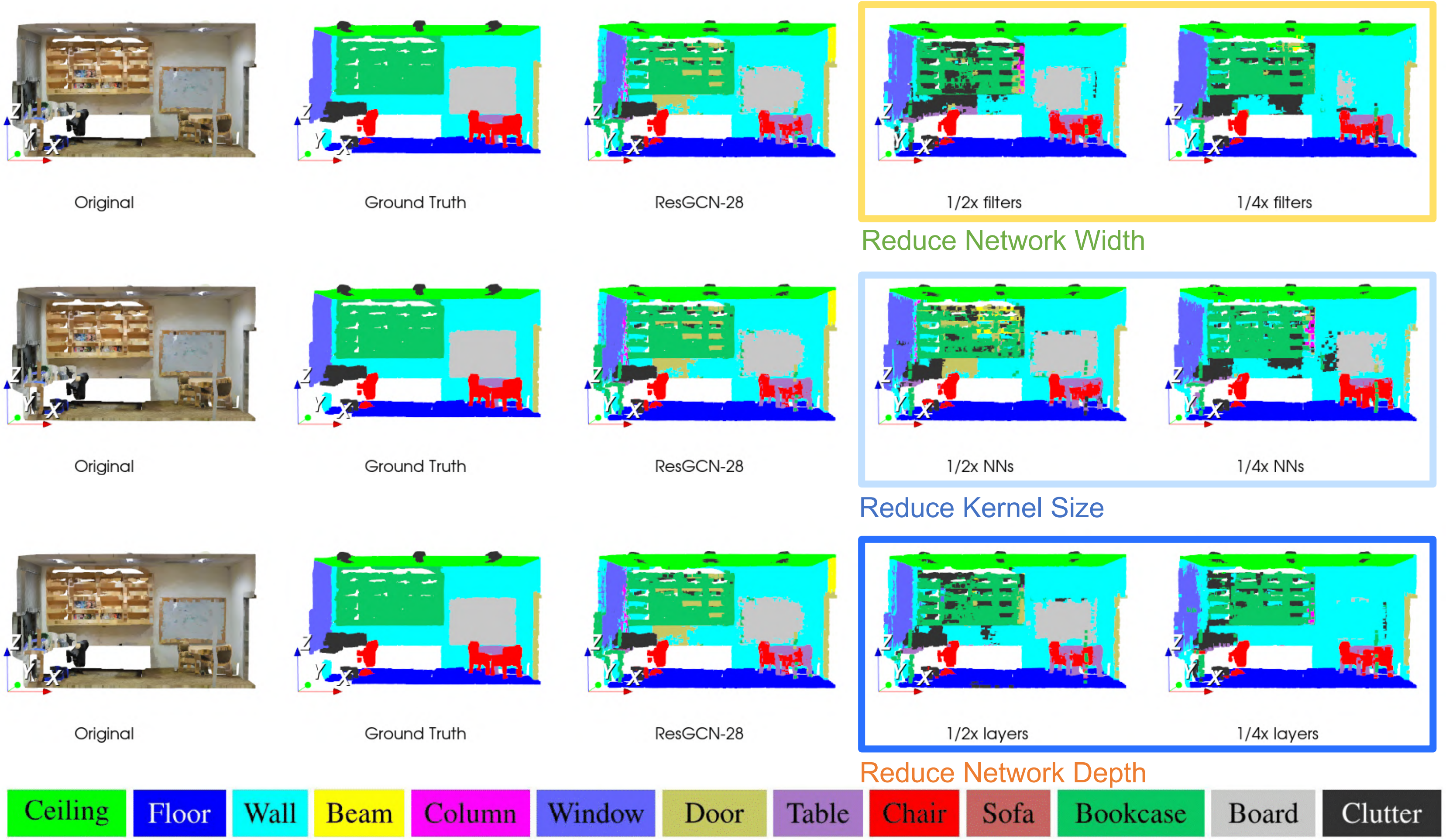
DeepGCNs.org

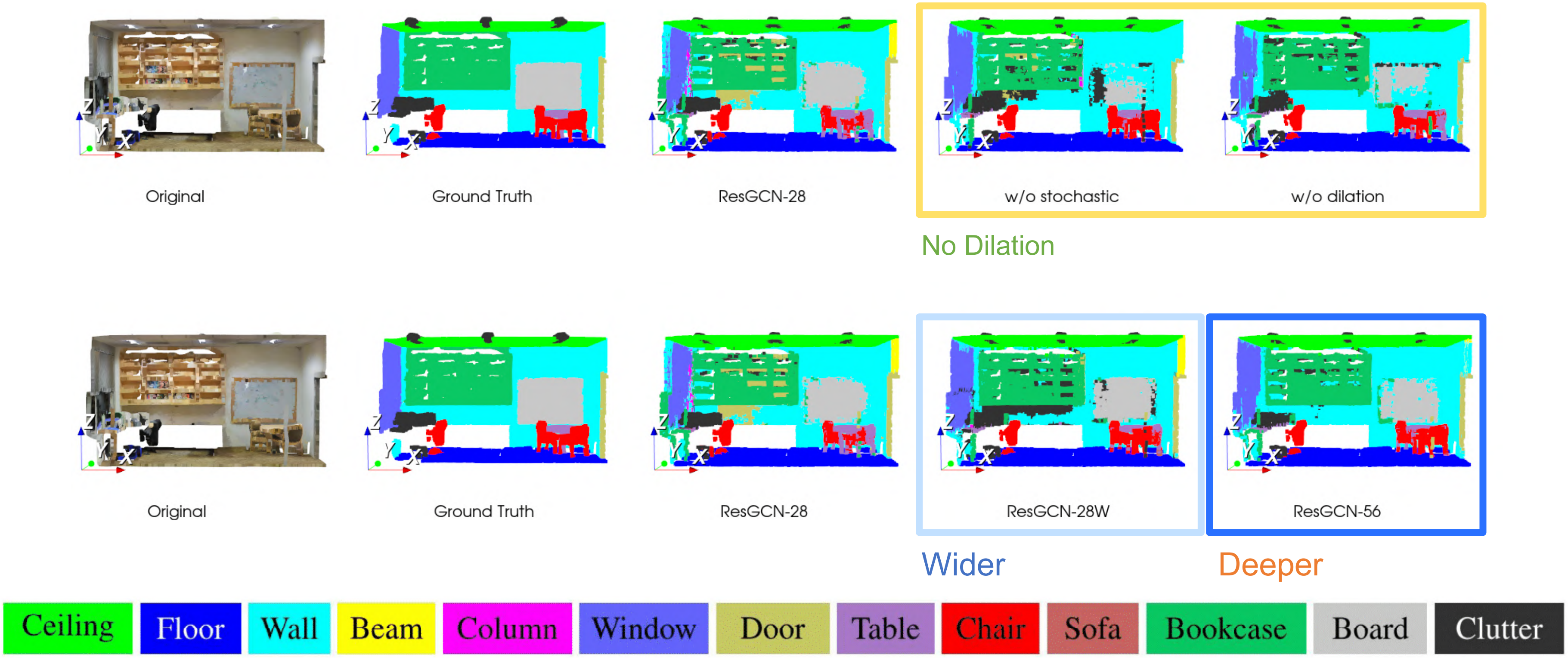


Visualizations on S3DIS


Original
Ground Truth
PlainGCN
ResGCN
DenseGCN
Ceiling
Floor
Wall
Beam
Column
Window
Door
Table
Chair
Sofa
Bookcase
Board
Clutter


Original
Ground Truth
PlainGCN
ResGCN
DenseGCN
Ceiling
Floor
Wall
Beam
Column
Window
Door
Table
Chair
Sofa
Bookcase
Board
Clutter


Original
Ground Truth
ResGCN-28
1/2x filters
1/4x filters
Reduce Network Width
Original
Ground Truth
ResGCN-28
1/2x NNs
1/4x NNs
Reduce Kernel Size
Original
Ground Truth
ResGCN-28
1/2x layers
1/4x layers
Reduce Network Depth
Ceiling
Floor
Wall
Beam
Column
Window
Door
Table
Chair
Sofa
Bookcase
Board
Clutter


Original
Ground Truth
ResGCN-28
w/o stochastic
w/o dilation
No Dilation
Original
Ground Truth
ResGCN-28
ResGCN-28W
ResGCN-56
Wider
Deeper
Ceiling
Floor
Wall
Beam
Column
Window
Door
Table
Chair
Sofa
Bookcase
Board
Clutter

GCN variants

- ResEdgeConv
- ResGraphSAGE
- ResGIN
- ResMRGCN

Model	mIoU	ΔmIoU	dynamic	connection	dilation	stochastic	# NNs	# filters	# layers
ResEdgeConv-28	**52.49**	0.00	✓	$\oplus$	✓	✓	16	64	28
PlainGCN-28	**40.31**	-12.18	✓				16	64	28
ResGraphSAGE-28	**49.20**	-3.29	✓	$\oplus$	✓	✓	16	64	28
ResGraphSAGE-N-28	**49.02**	-3.47	✓	$\oplus$	✓	✓	16	64	28
ResGIN-ϵ-28	**42.81**	-9.68	✓	$\oplus$	✓	✓	16	64	28
ResMRGCN-28	**51.17**	-1.32	✓	$\oplus$	✓	✓	16	64	28

Table 4. Comparisons of Deep GCNs variants on area 5 of S3DIS.

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

ResEdgeConv

$$h^{res}_{v_{l+1}} = mlp\left((1+\epsilon)\cdot h_{v_l} + sum(\{h_{u_l}|u_l \in \mathcal{N}^{(d)}(v_l)\})\right),$$
$$h_{v_{l+1}} = h^{res}_{v_{l+1}} + h_{v_l}.$$

ResGIN

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

ResGraphSAGE

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

ResMRGCN

More Results

DeepGCNs.org

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	-	-	-

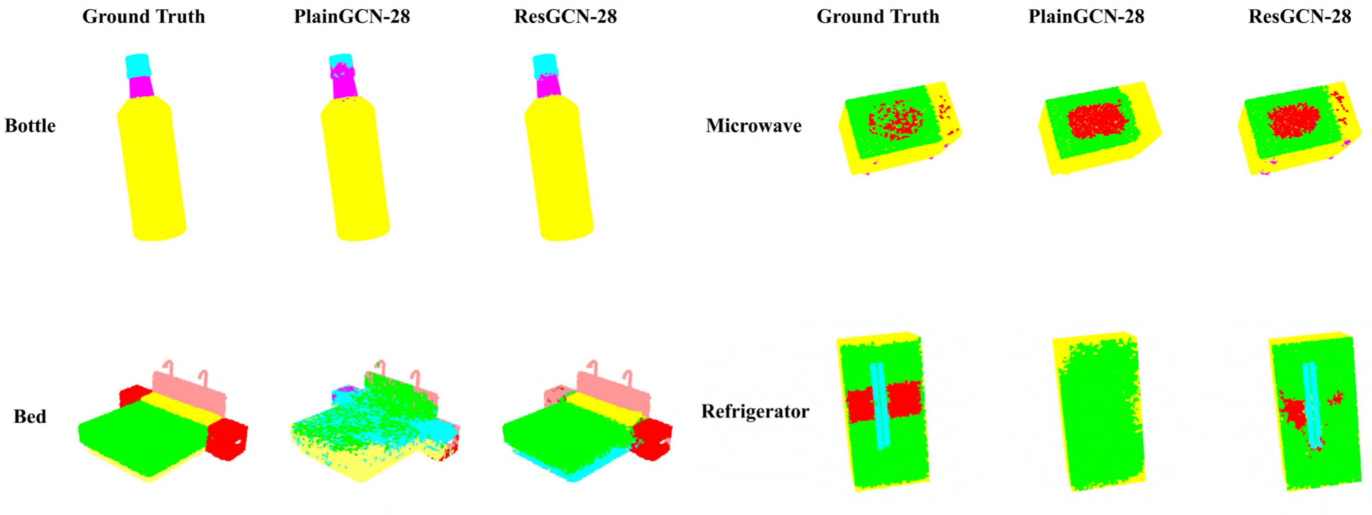
Table 5. Node classification of biological networks.

More Results

By John Morris.

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



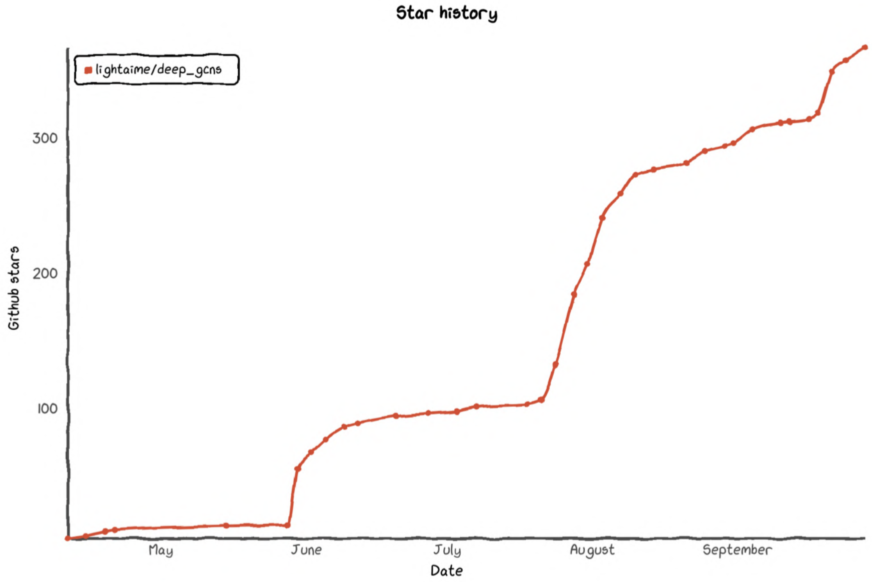
Method	bed	bottle	chair	clock	dishw.	disp.	door	earph.	fauc.	knife	lamp	micro.	fridge	st. furn.	table	tr. can	vase
PointNet [33]	13.4	29.5	27.8	28.4	48.9	76.5	30.4	33.4	47.6	32.9	18.9	37.2	33.5	38.0	29.0	34.8	44.4
PointNet++ [34]	30.3	**41.4**	**39.2**	**41.6**	50.1	80.7	32.6	38.4	**52.4**	34.1	**25.3**	48.5	36.4	40.5	**33.9**	46.7	49.8
SpiderCNN [53]	**36.2**	32.2	30.0	24.8	50.0	80.1	30.5	37.2	44.1	22.2	19.6	43.9	39.1	**44.6**	20.1	42.4	32.4
ResGCN-28 (*Ours*)	35.2	36.8	33.8	32.6	**52.7**	**84.4***	**42.5***	**41.9**	49.7	**35.4***	20.0	**54.3**	**46.1***	42.5	14.8	**49.7**	**50.8**
PointCNN [54]	41.9	41.8	43.9	36.3	58.7	82.5	37.8	48.9	60.5	34.1	20.1	58.2	42.9	49.4	21.3	53.1	58.9

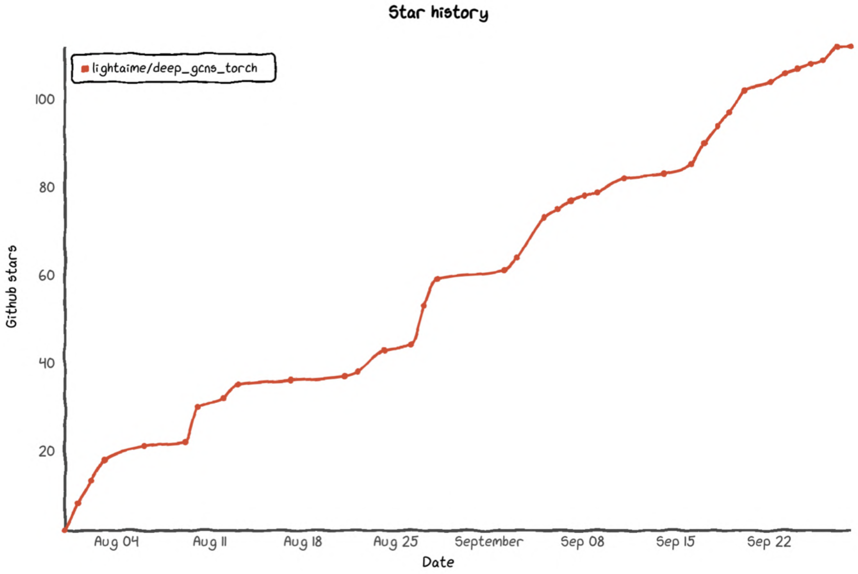
Table 7. Comparison of ResGCN-28 with other methods on PartNet Part Segmentation.

TensorFlow Repo

☆ 750+ Stars

Pytorch Repo





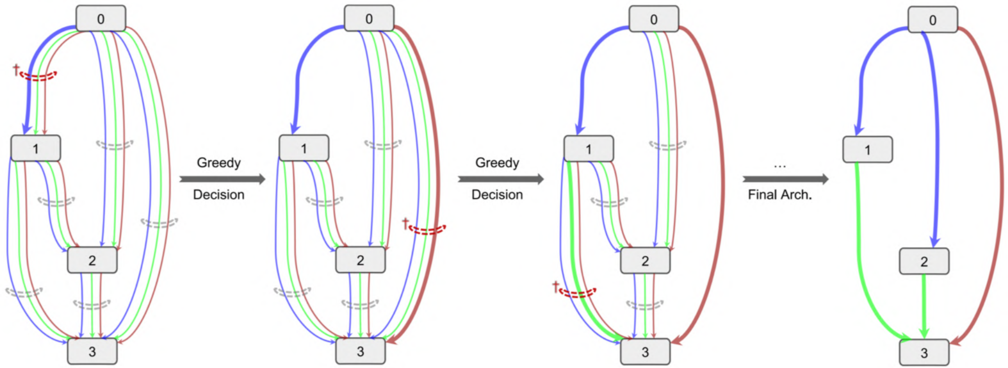
https://www.deepgcns.org

Can we learn how to design Deep GCN architectures automatically?

Can we learn how to design Deep GCN architectures automatically?

Neural Architecture Search!

SGAS: Sequential Greedy Architecture Search



SGAS: Sequential Greedy Architecture Search (arXiv 2019, Guohao Li et.al)

https://sites.google.com/kaust.edu.sa/sgas

SGAS: Sequential Greedy Architecture Search

Aiming to alleviate this common issue, we introduce sequential greedy architecture search (SGAS), an efficient method for neural architecture search.

By dividing the search procedure into sub-problems, SGAS chooses and prunes candidate operations in a greedy fashion.

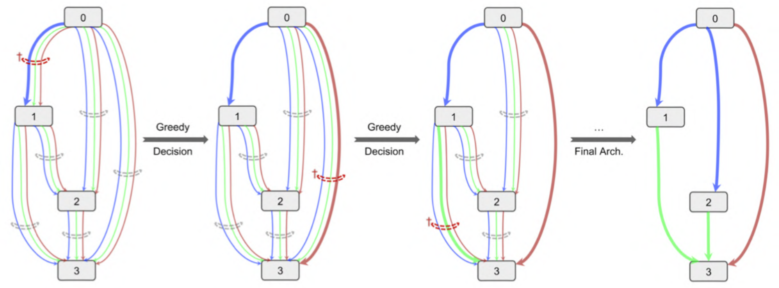


Figure 2. Illustration of Sequential Greedy Architecture Search.

SGAS: Sequential Greedy Architecture Search

We apply SGAS to search architectures for Convolutional Neural Networks (CNN) and Graph Convolutional Networks (GCN).

Extensive experiments show that SGAS is able to find SOTA architectures with minimal computational cost for tasks such as:

- image classification,
- point cloud classification,
- node classification in protein-protein interaction graphs.

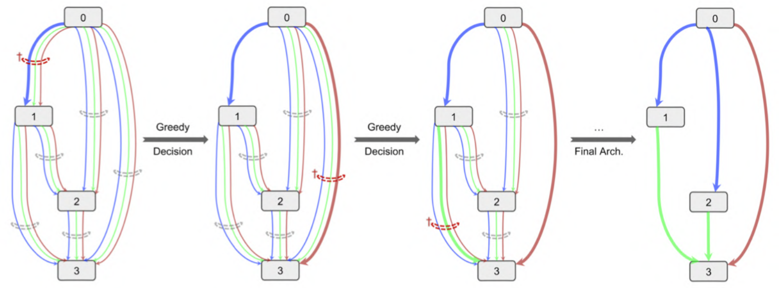


Figure 2. Illustration of Sequential Greedy Architecture Search.

SGAS: Sequential Greedy Architecture Search

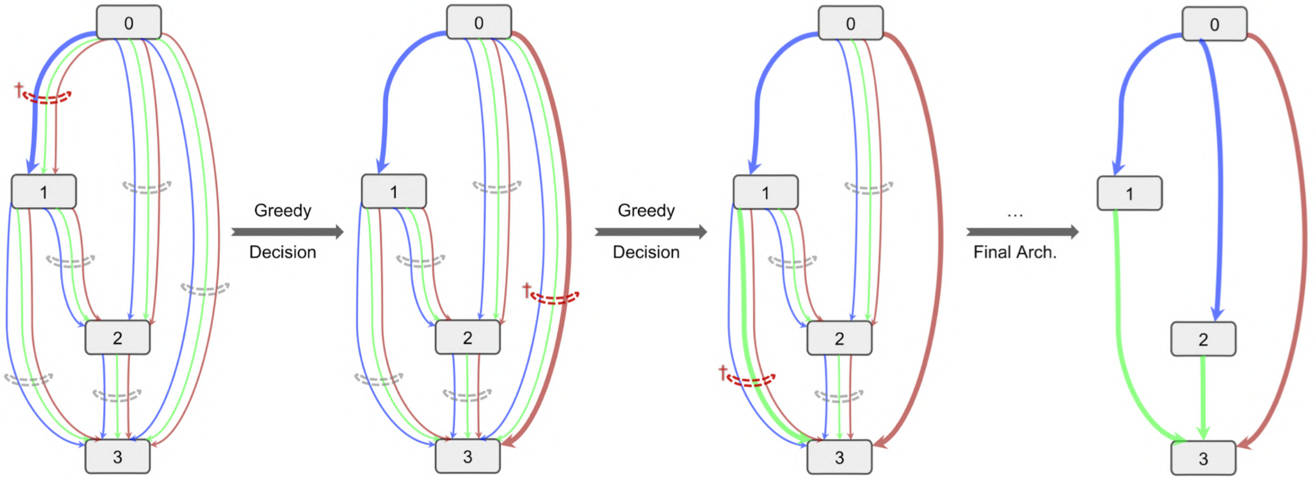


Figure 2. Illustration of Sequential Greedy Architecture Search.

SGAS: Sequential Greedy Architecture Search

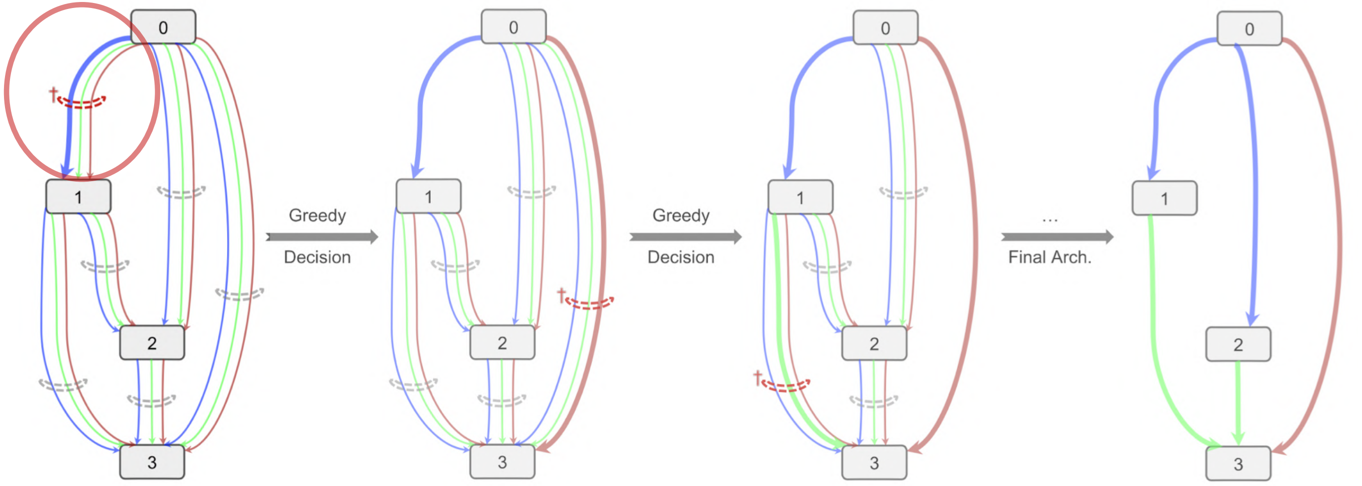


Figure 2. Illustration of Sequential Greedy Architecture Search.

1 If a decision epoch, select an edge $(i^\dagger, j^\dagger)$ based on the greedy *Selection Criterion*

SGAS: Sequential Greedy Architecture Search

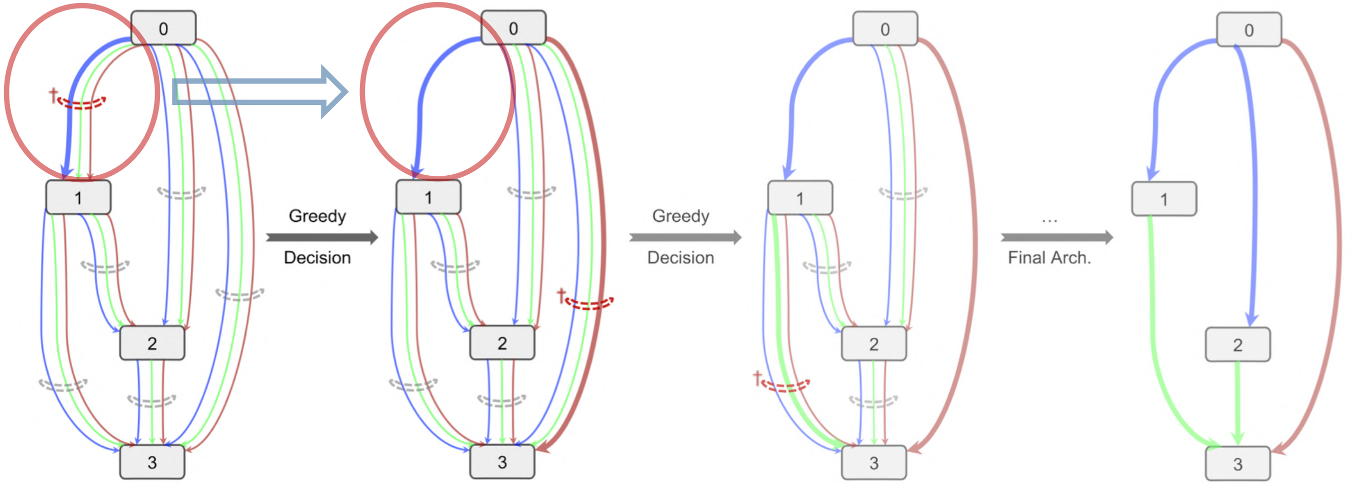


Figure 2. Illustration of Sequential Greedy Architecture Search.

1. If a decision epoch, select an edge $(i^\dagger, j^\dagger)$ based on the greedy *Selection Criterion*
2. Determine the operation by replacing $\bar{o}^{(i^\dagger,j^\dagger)}$ with $o^{(i^\dagger,j^\dagger)} = \text{argmax}_{o \in \mathcal{O}} \ \alpha_o^{(i^\dagger,j^\dagger)}$

SGAS: Sequential Greedy Architecture Search

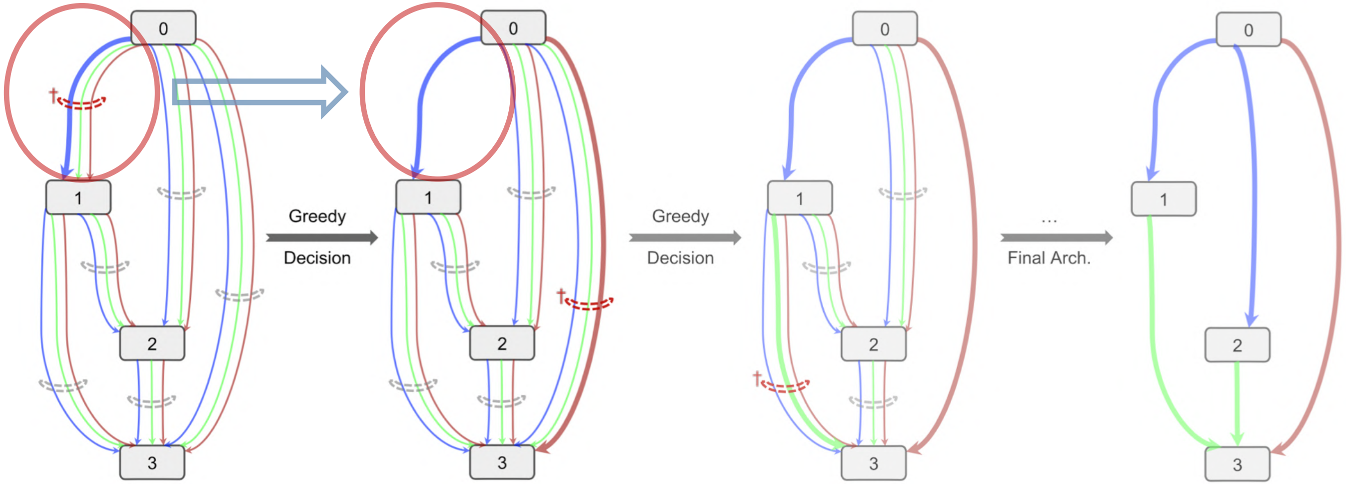


Figure 2. Illustration of Sequential Greedy Architecture Search.

1. If a decision epoch, select an edge $(i^\dagger, j^\dagger)$ based on the greedy *Selection Criterion*
2. Determine the operation by replacing $\bar{o}^{(i^\dagger, j^\dagger)}$ with $o^{(i^\dagger, j^\dagger)} = \text{argmax}_{o \in \mathcal{O}} \ \alpha_o^{(i^\dagger, j^\dagger)}$
3. Prune unchosen weights from $\mathcal{W}$, Remove $\alpha^{(i^\dagger, j^\dagger)}$ from $\mathcal{A}$

SGAS: Sequential Greedy Architecture Search

Repeat…

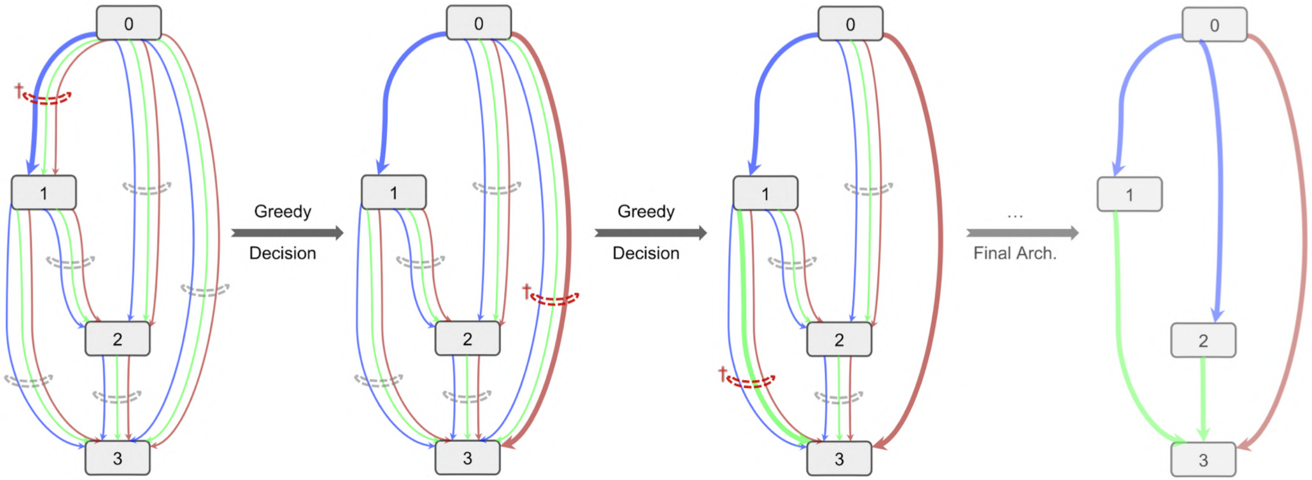


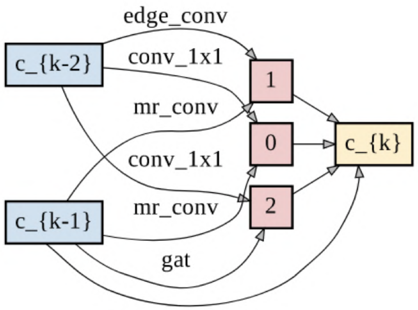
Figure 2. Illustration of Sequential Greedy Architecture Search.

1. If a decision epoch, select an edge $(i^\dagger, j^\dagger)$ based on the greedy *Selection Criterion*
2. Determine the operation by replacing $\bar{o}^{(i^\dagger, j^\dagger)}$ with $o^{(i^\dagger, j^\dagger)} = \text{argmax}_{o \in \mathcal{O}} \ \alpha_o^{(i^\dagger, j^\dagger)}$
3. Prune unchosen weights from $\mathcal{W}$, Remove $\alpha^{(i^\dagger, j^\dagger)}$ from $\mathcal{A}$

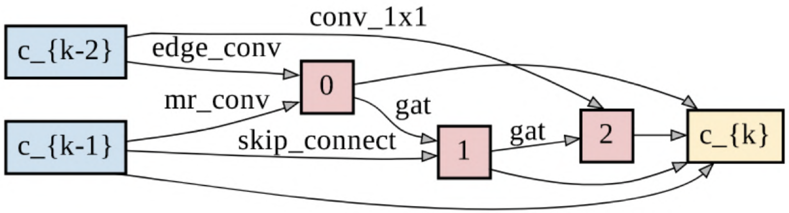
Results – SGAS for GCN on ModelNet

Architecture	OA (%)	Params (M)	Search Cost (GPU-days)
3DmFV-Net [4]	91.6	45.77	manual
SpecGCN [54]	91.5	2.05	manual
PointNet++ [42]	90.7	1.48	manual
PCNN [3]	92.3	8.2	manual
PointCNN [31]	92.2	0.6	manual
DGCNN [55]	92.2	1.84	manual
KPConv [51]	92.9	14.3	manual
Random Search	92.65±0.33	8.77	random
SGAS (Cri.1 avg.)	92.69±0.20	8.78	0.19
SGAS (Cri.1 best)	92.87	8.63	0.19
SGAS (Cri.2 avg.)	92.93±0.19	8.87	0.19
SGAS (Cri.2 best)	**93.23**	8.49	0.19
SGAS (Cri.2 small best)	93.07	3.86	0.19

Table 3. Comparison with state-of-the-art architectures for 3D object classification on ModelNet40.



(a) Normal cell of the best model with SGAS (Cri. 1) on ModelNet

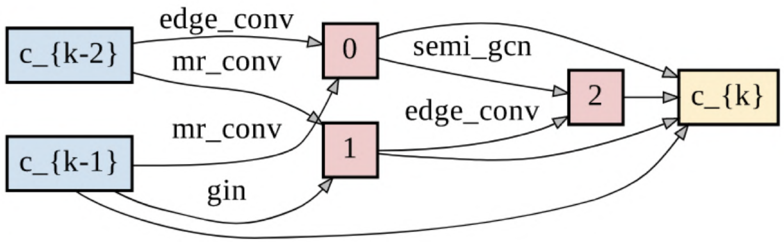


(b) Normal cell of the best model with SGAS (Cri. 2) on ModelNet

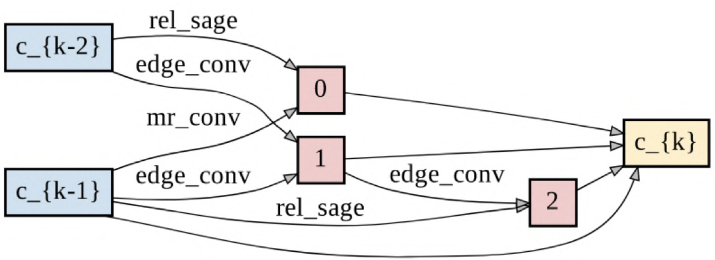
Results – SGAS for GCN on PPI

Architecture	micro-F1 (%)	Params (M)	Search Cost (GPU-days)
GraphSAGE (LSTM) [14]	61.2	0.26	manual
GeniePath [30]	97.9	1.81	manual
GAT [44]	97.3±0.2	3.64	manual
DenseMRGCN-14 [23]	99.43	53.42	manual
ResMRGCN-28 [23]	99.41	14.76	manual
Random Search	99.36±0.04	23.70	random
SGAS (Cri.1 avg.)	99.38±0.17	25.01	0.003
SGAS (Cri.1 best)	**99.46**	23.18	0.003
SGAS (Cri.2 avg.)	99.40±0.09	25.93	0.003
SGAS (Cri.2 best)	**99.46**	29.73	0.003
SGAS (small)	98.89	0.40	0.003

Table 4. Comparison with state-of-the-art architectures for node classification on PPI.



(a) Normal cell of the best model with SGAS (Cri. 1) on PPI



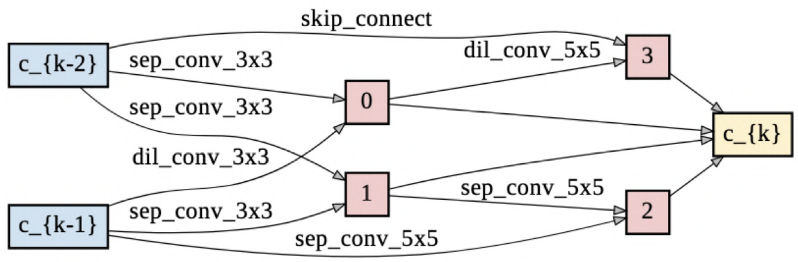
(b) Normal cell of the best model with SGAS (Cri. 2) on PPI

Results – SGAS for CNN on CIFAR-10

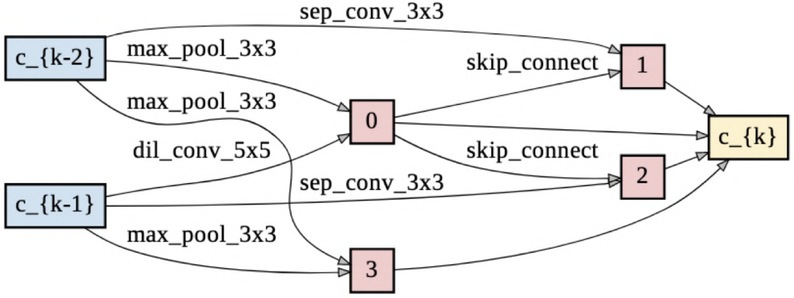
Architecture	Test Err. (%)	Params (M)	Search Cost (GPU-days)	Search Method
DenseNet-BC [18]	3.46	25.6	-	manual
NASNet-A [55]	2.65	3.3	1800	RL
AmoebaNet-A [36]	3.34±0.06	3.2	3150	evolution
AmoebaNet-B [36]	2.55±0.05	2.8	3150	evolution
Hier-Evolution [28]	3.75±0.12	15.7	300	evolution
PNAS [27]	3.41±0.09	3.2	225	SMBO
ENAS [34]	2.89	4.6	0.5	RL
NAONet-WS [31]	3.53	3.1	0.4	NAO
DARTS (1st order) [29]	3.00±0.14	3.3	0.4	gradient
DARTS (2nd order) [29]	2.76±0.09	3.3	1	gradient
SNAS (mild) [49]	2.98	2.9	1.5	gradient
ProxylessNAS [7]	2.08	-	4	gradient
P-DARTS [8]	2.5	3.4	0.3	gradient
BayesNAS [52]	2.81±0.04	3.4	0.2	gradient
PC-DARTS [50]	2.57±0.07	3.6	0.1	gradient
SGAS (Cri.1 avg.)	2.66±0.24*	3.7	0.25	gradient
SGAS (Cri.1 best)	2.39	3.8	0.25	gradient
SGAS (Cri.2 avg.)	2.67±0.21*	3.9	0.25	gradient
SGAS (Cri.2 best)	2.44	4.1	0.25	gradient

Table 1. Performance comparison with state-of-the-art image classifiers on CIFAR-10.

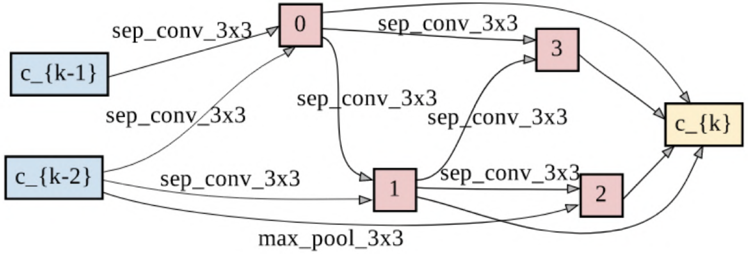
Results – SGAS for CNN on CIFAR-10



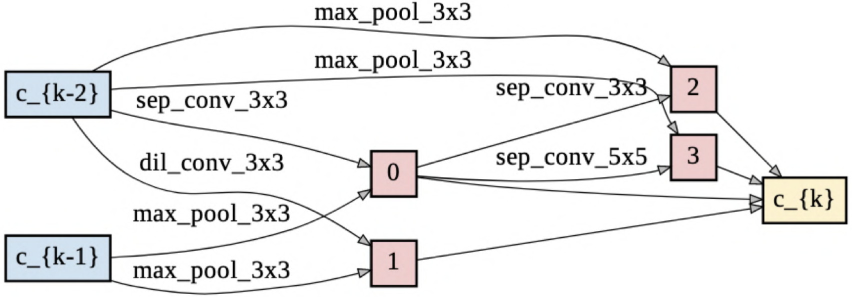
(a) Normal cell of the best model with SGAS (Cri. 1) on CIFAR-10



(b) Reduction cell of the best model with SGAS (Cri. 1) on CIFAR-10



(c) Normal cell of the best model with SGAS (Cri. 2) on CIFAR-10



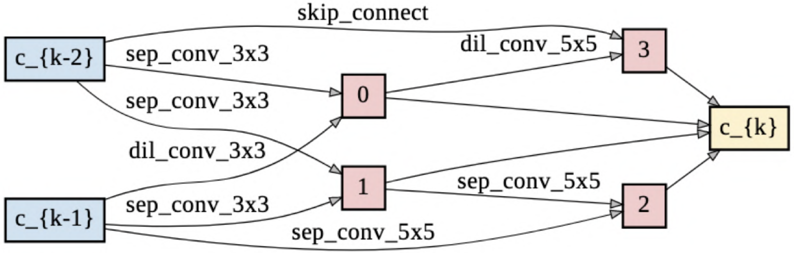
(d) Reduction cell of the best model with SGAS (Cri. 2) on CIFAR-10

Results – SGAS for CNN on ImageNet

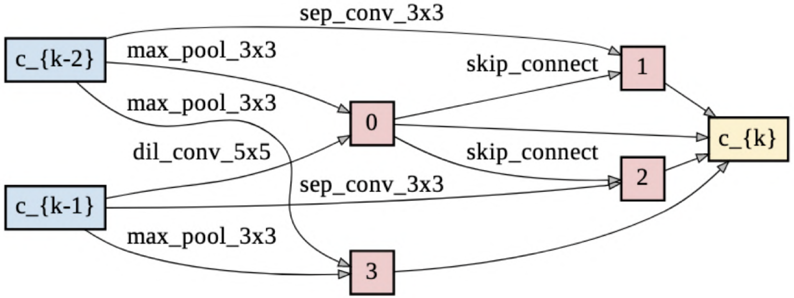
Architecture	Test Err. (%) top-1	Test Err. (%) top-5	Params (M)	×+ (M)	Search Cost (GPU-days)	Search Method
Inception-v1 [41]	30.2	10.1	6.6	1448	-	manual
MobileNet [16]	29.4	10.5	4.2	569	-	manual
ShuffleNet 2x (v1) [51]	26.4	10.2	∼5	524	-	manual
ShuffleNet 2x (v2) [32]	25.1	-	∼5	591	-	manual
NASNet-A [55]	26	8.4	5.3	564	1800	RL
NASNet-B [55]	27.2	8.7	5.3	488	1800	RL
NASNet-C [55]	27.5	9	4.9	558	1800	RL
AmoebaNet-A [36]	25.5	8	5.1	555	3150	evolution
AmoebaNet-B [36]	26	8.5	5.3	555	3150	evolution
AmoebaNet-C [36]	24.3	7.6	6.4	570	3150	evolution
PNAS [27]	25.8	8.1	5.1	588	225	SMBO
MnasNet-92 [42]	25.2	8	4.4	388	-	RL
DARTS (2^{nd} order) [29]	26.7	8.7	4.7	574	4.0	gradient
SNAS (mild) [49]	27.3	9.2	4.3	522	1.5	gradient
ProxylessNAS [7]	24.9	7.5	7.1	465	8.3	gradient
P-DARTS [8]	24.4	7.4	4.9	557	0.3	gradient
BayesNAS [52]	26.5	8.9	3.9	-	0.2	gradient
PC-DARTS [50]	25.1	7.8	5.3	586	0.1	gradient
SGAS (Cri.1 avg.)	24.4±0.2	7.3±0.1	5.3	579	0.25	gradient
SGAS (Cri.1 best)	24.2	7.2	5.3	585	0.25	gradient
SGAS (Cri.2 avg.)	24.4±0.2	7.4±0.1	5.4	597	0.25	gradient
SGAS (Cri.2 best)	**24.1**	7.3	5.4	598	0.25	gradient

Table 2. Performance comparison with state-of-the-art image classifiers on ImageNet.

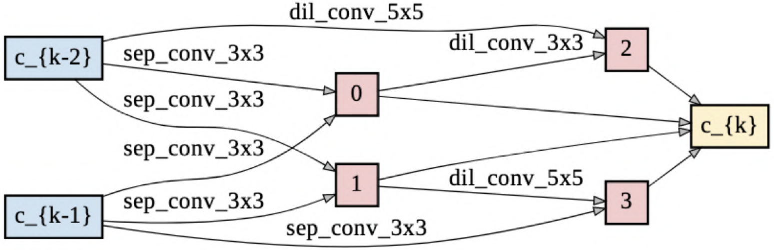
Results – SGAS for CNN on ImageNet



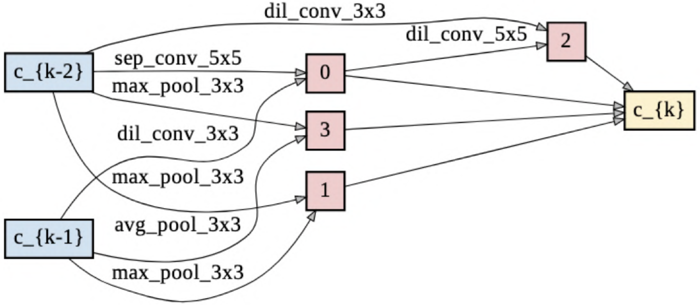
(a) Normal cell of the best model with SGAS (Cri. 1) on ImageNet



(b) Reduction cell of the best model with SGAS (Cri. 1) on ImageNet



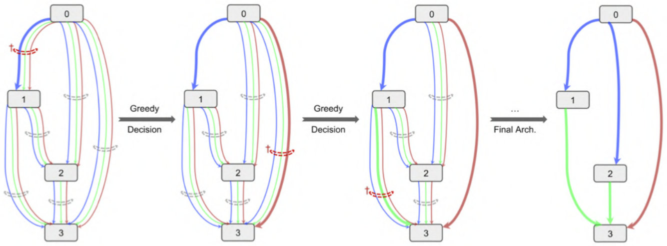
(c) Normal cell of the best model with SGAS (Cri. 2) on ImageNet



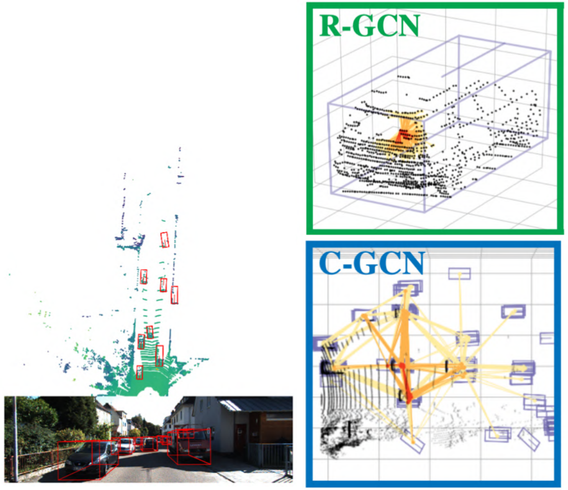
(d) Reduction cell of the best model with SGAS (Cri. 2) on ImageNet

Follow-up works

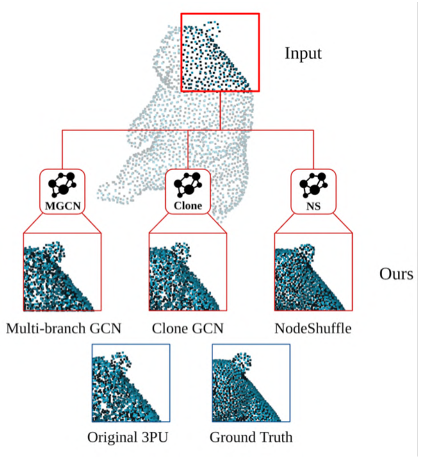
DeepGCNs.org



SGAS: Sequential Greedy Architecture Search. Guohao Li. et al.



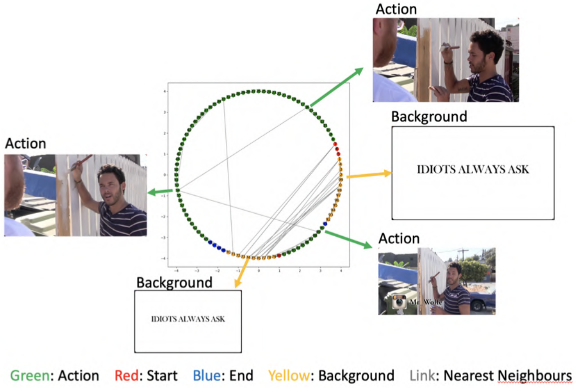
PointRGCN: Graph Convolution Networks for 3D Vehicles Detection Refinement. Jesue Zarzar. et al.



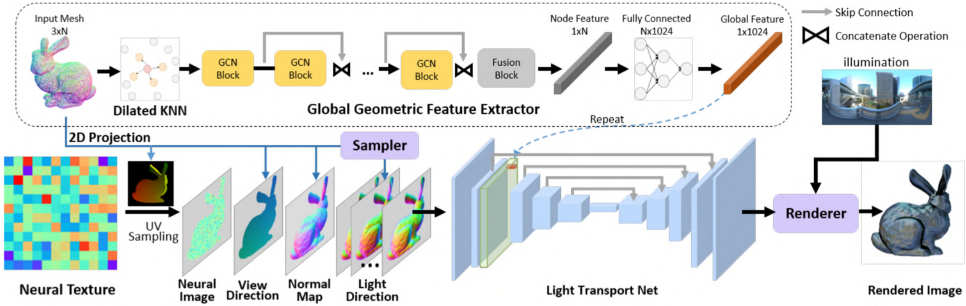
PU-GCN: Point Cloud Upsampling via Graph Convolutional Network. Guocheng Qian. et al.

Follow-up works

DeepGCNs.org



G-TAD: Sub-Graph Localization for Temporal Action Detection. Mengmeng xu. et al.



A Neural Rendering Framework for Free-Viewpoint Relighting. Zhang Chen. et al.

Useful materials or tools

DeepGCNs.org

- Thomas Kipf: http://tkipf.github.io/misc/SlidesCambridge.pdf
- Stanford SNAP: http://snap.stanford.edu/proj/embeddings-www/files/nrltutorial-part2-gnns.pdf

Useful materials or tools

- Thomas Kipf: http://tkipf.github.io/misc/SlidesCambridge.pdf
- Stanford SNAP: http://snap.stanford.edu/proj/embeddings-www/files/nrltutorial-part2-gnns.pdf

- Pytorch Geometric: https://pytorch-geometric.readthedocs.io
- Deep Graph Library: https://www.dgl.ai/
- TensorFlow Graphics: https://github.com/tensorflow/graphics

NeurIPS MeetUp '19

DeepGCNs.org

THANK YOU

DeepGCNs for Representation Learning on Graphs

DeepGCNs: Can GCNs Go as Deep as CNNs? (*ICCV'2019 Oral*)
Guohao Li*, Matthias Müller*, Ali Thabet, Bernard Ghanem

DeepGCNs: Making GCNs Go as Deep as CNNs (*arXiv'2019*)
Guohao Li*, Matthias Müller*, Guocheng Qian, Itzel C. Delgadillo, Abdulellah Abualshour, Ali Thabet, Bernard Ghanem

SGAS: Sequential Greedy Architecture Search (*arXiv'2019*)
Guohao Li*, Guocheng Qian*, Itzel C. Delgadillo*, Matthias Müller, Ali Thabet, Bernard Ghanem

KAUST

Contact: guohao.li@kaust.edu.sa
About me: https://ghli.org

VCC VISUAL COMPUTING CENTER IVUL