DeepGCNs: Can GCNs Go as Deep as CNNs?

Authors: Guohao Li, Matthias Muller, Ali Thabet, Bernard Ghanem.

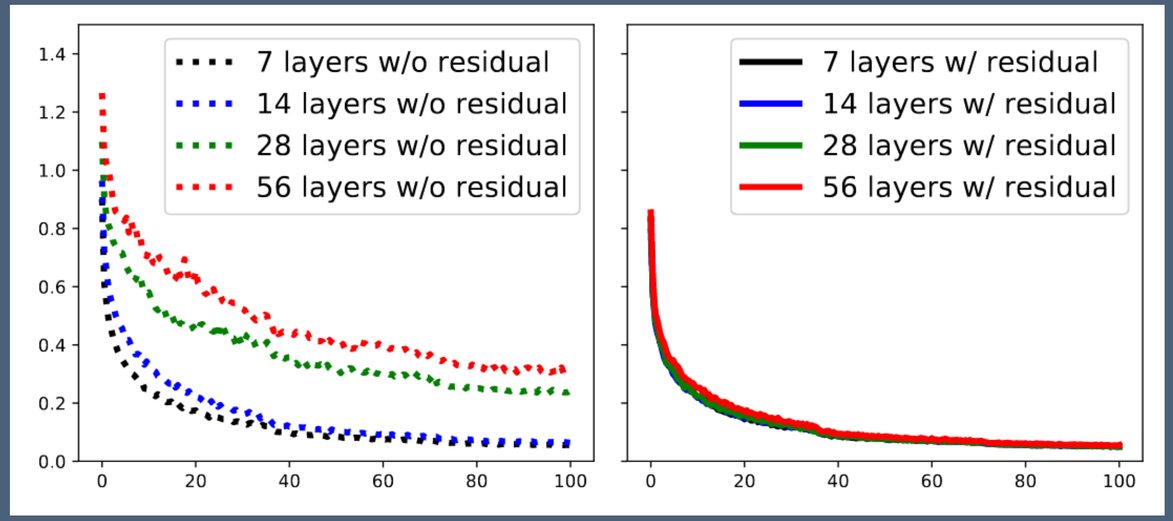
Presented by: Lucas Fenaux

Overview

- Problem: Vanishing gradients
- Previous Work
- Residual connections
- Dense connections
- Dilated convolutions
- Empirical evaluation

The vanishing gradient problem

- Depth of the network shrinks the magnitude of the gradients of earlier layers. -> Unstable training.
- Graph Neural Networks are not the only models to suffer from this problem:
  - Convolution Neural Networks (CNNs)
  - Recurrent Neural Networks (RNNs)



Small Scale Example

(For an MLP with sigmoid activation σ)

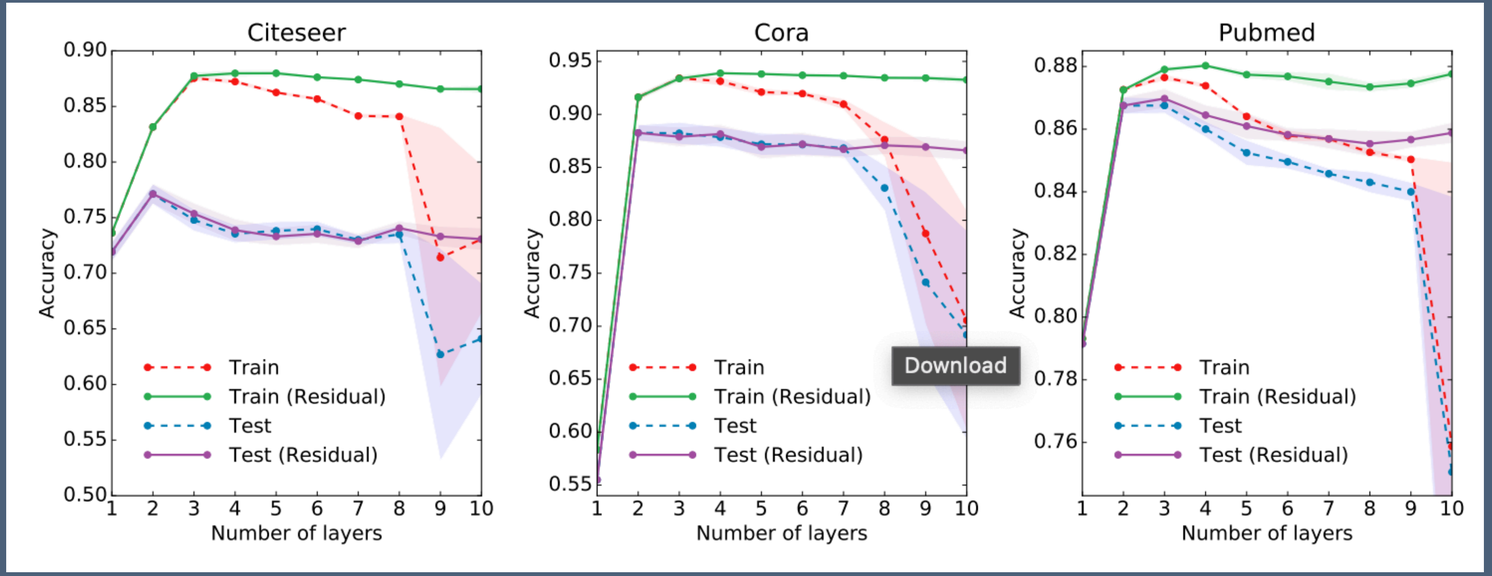
$$\sigma(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}$$

- Let $f(x) = \sigma(A_2(\sigma(A_1(x) + b_1)) + b_2)$
- $\dfrac{\partial f}{\partial A_2} = \sigma(A_2)(1 - \sigma(A_2))$
- $0 < \sigma(A_2) < 1 \implies 0 < 1 - \sigma(A_2) < 1 \implies \sigma(A_2)(1 - \sigma(A_2) < \sigma(A_2)$
- $\dfrac{\partial f}{\partial A_1} = \sigma(A_2)(1 - \sigma(A_2))\sigma(A_1)(1 - \sigma(A_1) < \dfrac{\partial f}{\partial A_2}$
- So as we go towards earlier layers, the gradients get smaller and smaller.

Existing GCN architectures

- GCN: performance degrades after 3 layers. (Kipf et al.)
- Column Network: performance degrades beyond 10 layers. (Pham et al.)
- Highway GCN: 6 layers (Rahimi et al.)
- Jump Knowledge Network: 6 layers (Xu et al.)

Kipf et al.'s GCN



- Layer-wise propagation rule:
  $H^{(l+1)} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)})$
- Where $\tilde{A} = A + I_n$ (identity matrix + self-connection). $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$
- This layer-wise propagation rule is motivated via a first-order approximation of localized spectral filters on graphs.
- They also propose a version with residual connections:
  $H^{(l+1)} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}) + H^{(l)}$

Column Network

- Leverages different relation types for better performance.
- For example, for movies:
  - Movie B being a sequel of movie A.
  - Movies A and B sharing an actor.
  - Movie A and B sharing a director.
- In a way, leverage multiple graphs from the same vertices at the same time.

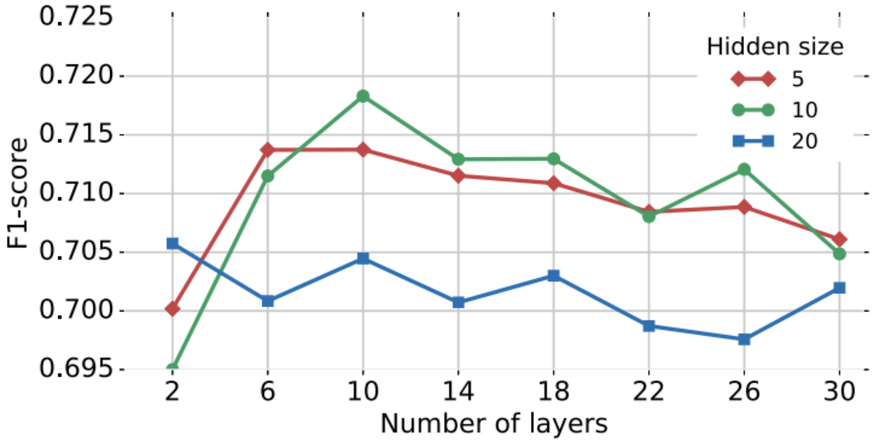


Figure 3: Performance (F1-score) of CLN-HWN on Software delay prediction task with different numbers of layers and hidden sizes

Column Network

- Update rule:

$$h_i^{(t)} = g\left(b^{(t)} + W^{(t)} h_i^{(t-1)} + \frac{1}{z}\sum_{r=1}^{R} V_r^{(t)} \left(\frac{1}{|N_r(i)|}\sum_{j \in N_r(i)} h_j^{(t-1)}\right)\right)$$

- Where g is an activation function, h_i^t is the state of node i at layer t, $W^{(t)}$ and $b^{(t)}$ are the weights and biases for layer t, $\frac{1}{z}$ is a normalization factor, R is the number of relations, $V_r^{(t)}$ are the weight matrices for relational aggregation, and $N_r(i)$ the set of neighbors of e_i that are connected by relation r.

Highway GCN

- Originally for user geolocation, it combines text and network information.
- Text view: Each user is represented by a bag-of-words vector (denoted as X) that captures their tweet content.
- Network View: The social (or @-mention) graph is encoded in an adjacency matrix A. To incorporate self-information, the model forms a modified graph matrix by adding a scaled identity matrix, and then normalizes it: $\hat{A} = \tilde{D}^{-1/2}(A + \lambda I)\tilde{D}^{-1/2}$. Where $\tilde{D}$ is the degree matrix of $A + \lambda I$
- Propagation rule: $H^{(l+1)} = \sigma\left(\hat{A}\, H^{(l)}\, W^{(l)} + b\right)$

Highway GCN

- To allow for more layers: gating mechanism:

$$T(\tilde{h}^{(l)}) = \sigma\left(W_t^{(l)}\tilde{h}^{(l)} + b_t^{(l)}\right)$$

$$0 \leq T(\tilde{h}^{(l)}) \leq 1$$

- Gated update:

$$\tilde{h}^{(l+1)} = \tilde{h}^{(l+1)} \circ T(\tilde{h}^{(l)}) + \tilde{h}^{(l)} \circ \left(1 - T(\tilde{h}^{(l)})\right)$$

- Gate balances how much of the update comes from each layer, limiting the effect of over-smoothing.

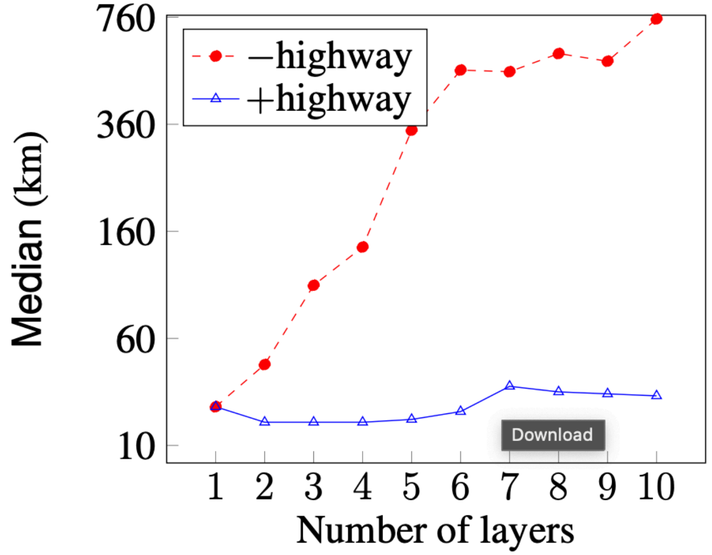


Figure 5: The effect of adding more GCN layers (neighbourhood expansion) to GCN in terms of median error over the development set of GEOTEXT with and without the highway gates, and averaged over 5 runs.

Jump Knowledge Network

- Conceptually similar to DenseNet.
- Final feature representation doesn't rely solely on the last layer. Instead, relies on all previous intermediate representations: allows for a better receptive field.

Model	Citeseer	Model	Cora
GCN (2)	77.3 (1.3)	GCN (2)	88.2 (0.7)
GAT (2)	76.2 (0.8)	GAT (3)	87.7 (0.3)
JK-MaxPool (1)	77.7 (0.5)	JK-Maxpool (6)	**89.6** (0.5)
JK-Concat (1)	**78.3** (0.8)	JK-Concat (6)	89.1 (1.1)
JK-LSTM (2)	74.7 (0.9)	JK-LSTM (1)	85.8 (1.0)

Table 2. Results of GCN-based JK-Nets on Citeseer and Cora. The baselines are GCN and GAT. The number in parentheses next to the model name indicates the best-performing number of layers among 1 to 6. Accuracy and standard deviation are computed from 3 random data splits.

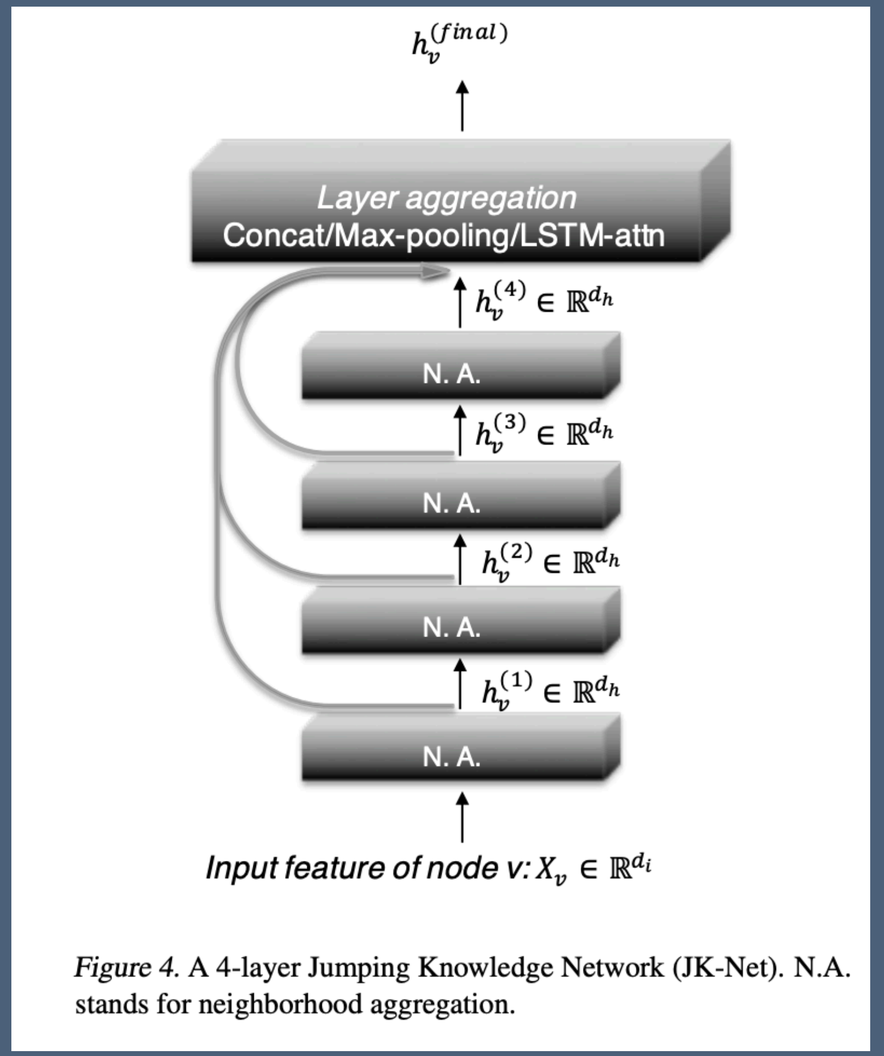


Figure 4. A 4-layer Jumping Knowledge Network (JK-Net). N.A. stands for neighborhood aggregation.

Proposed Solutions

Inspired by recent CNN advancements

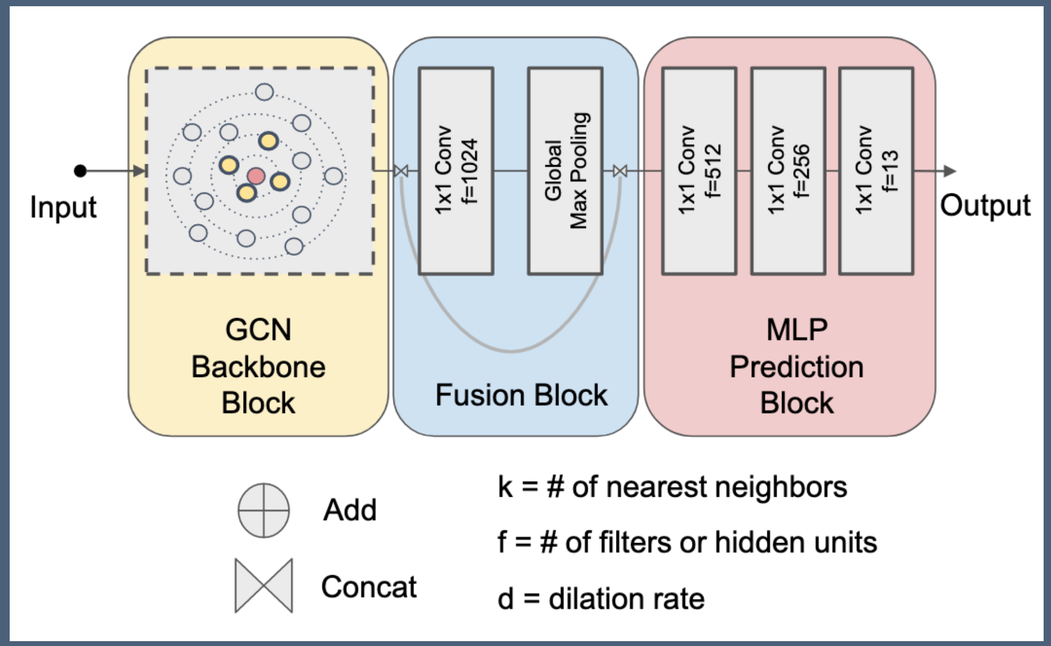
- ResNet -> Residual connections
- DenseNet -> Dense connections
- Dilated convolutions -> dilated graph convolutions

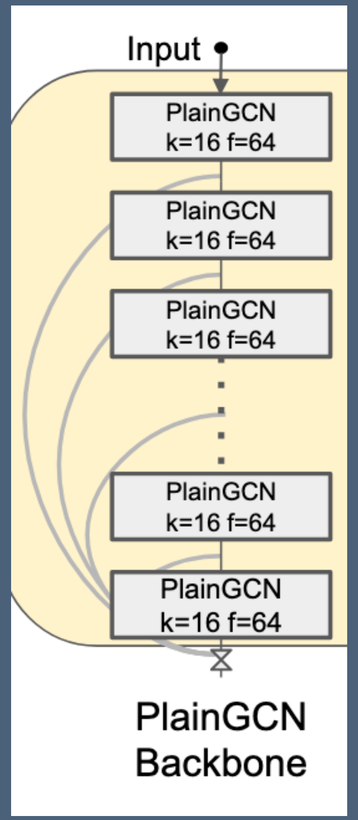
Notation

- Graph: $G(V, E)$
- Convolution at layer l: $G_{l+1} = F(G_l, W_l)$.
- $F(G_l, W_l) = Update(Aggregate(G_l, W^{agg})W_l^{update})$.

Overall Architecture

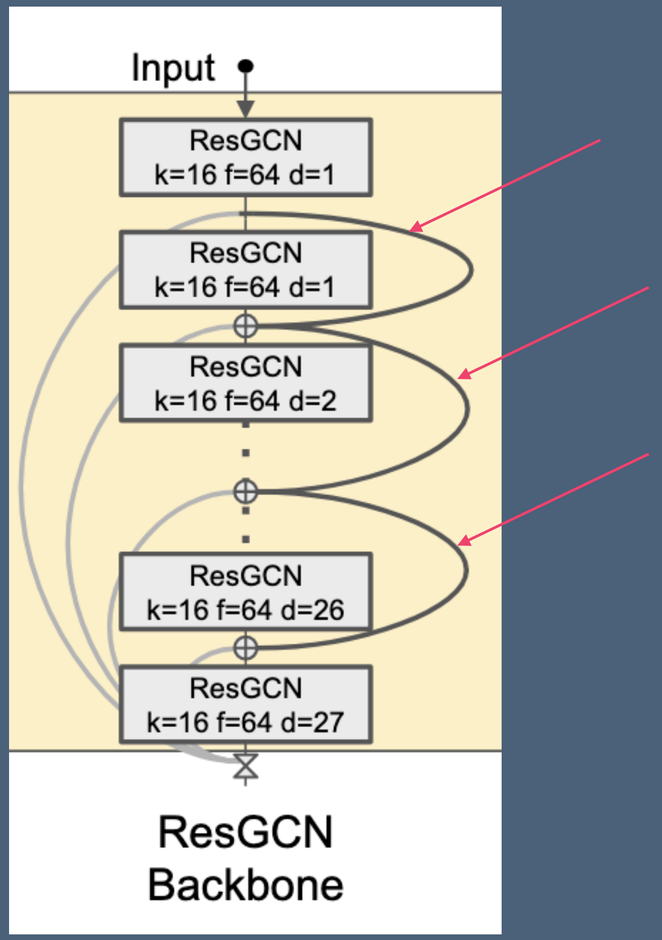
- Complete architecture.
- They vary the backbones.
- The baseline backbone is the PlainGCN backbone.
- Fusion block inspired from the DenseNet architecture.





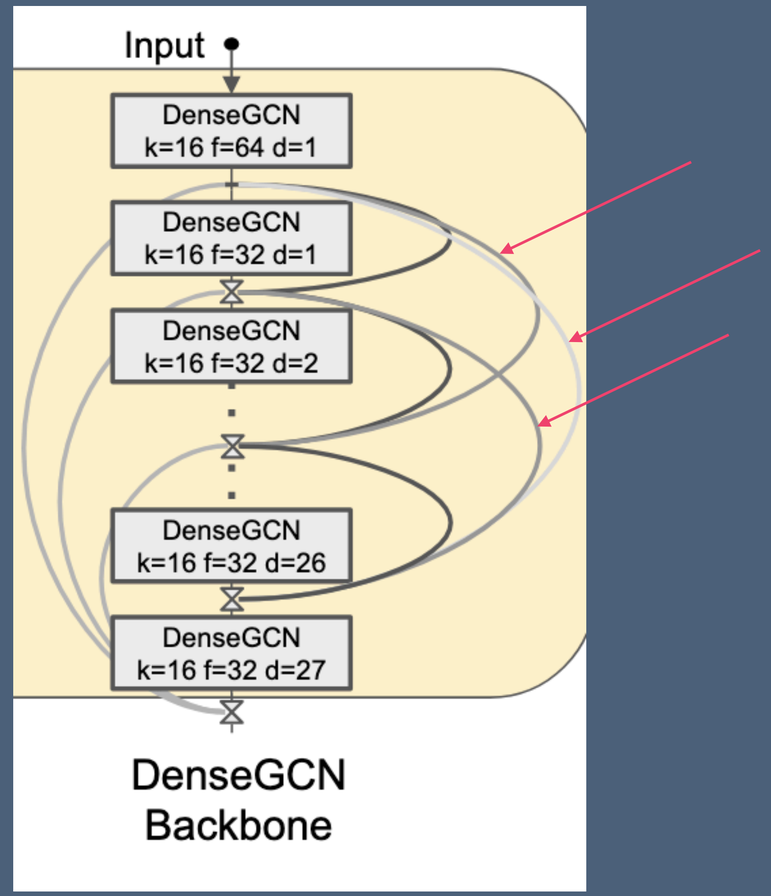
ResGCN

- Learn an underlying mapping H.
- $G_{l+1} = H(G_l, W_l)$
- $H(G_l, W_l) = F(G_l, W_l) + G_l = G_{l+1}^{res} + G_l$
- This preserves the magnitude of the gradient signal through the layers (adding G_l means that the derivative is at least 1 and therefore doesn't collapse).



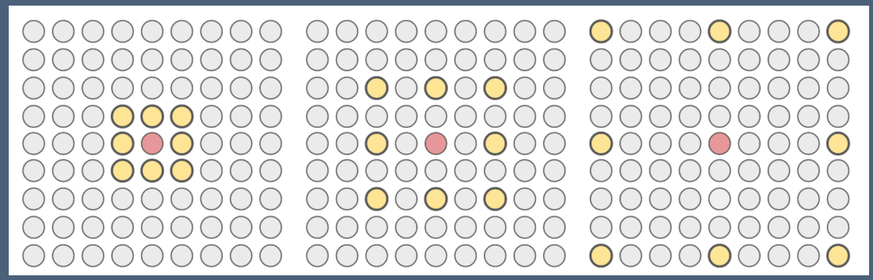
DenseGCN

- $G_{l+1} = H(G_l, W_l)$
- $H(G_l, W_l) = T(F(G_l, W_l), G_l)$
- $H(G_l, W_l) = T(F(G_l, W_l), \ldots, F(G_0, W_0), G_0)$
- Where T is a vertex-wise concatenation function that densely fuses G_0 with all the intermediate GCN layer outputs.
- Dimension of each vertex feature of G_{l+1} is $D_0 + D(l+1)$.



Dilated Convolutions

In CNNs (Yu et al.)



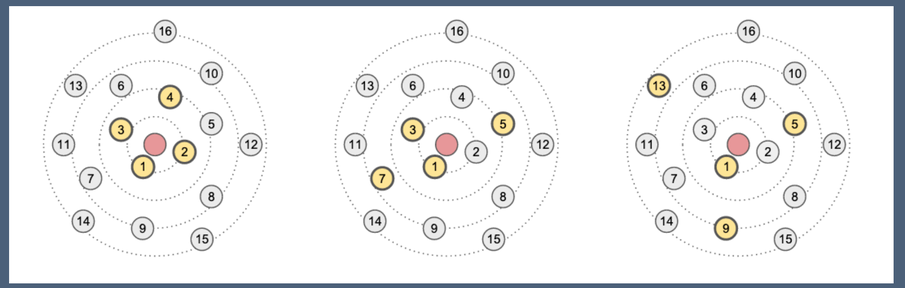
- A standard convolution computes an output feature at every spatial location by sliding a kernel over the input. In a dilated convolution, the kernel is “stretched” by inserting spaces (zeros) between its weights.

- Retains the spatial dimension. The dilation rate increases the effective size of the kernel (i.e., its receptive field) but does not change the resolution of the feature map.

- $2DDilatedConv(i,j) = \sum_{m=0}^{k-1}\sum_{n=0}^{k-1} X_{i+dm,j+dn}K_{m,n}$. Where $X \in R^m \times R^n$ is the input to the convolution operation, K is the $k \times k$ kernel, and d is the dilation factor. If $d = 1$, we get the “normal” convolution operation.

Dilated Convolutions

Dilated k-NNs & Dilated Graphs



- Use a Dilated k-NN to find dilated neighbours and construct a Dilated Graph.
- Given a graph $G = (V, E)$, a Dilated k-NN and d as the dilation factor, the Dilate k-NN returns the k nearest neighbours within the $k \times d$ neighbourhood by skipping every d neighboors.
- If $(u_1, u_2, \ldots, u_{k\times d})$ be the first sorted $k \times d$ nearest neighbours. Let $\mathcal{N}^{(d)}(v)$ be the d-dilated neighborhood of vertex v.

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

Dilated Convolutions

- The GCN aggregation operation are applied based on the edges of the Dilated Graph $G^{(d)} = (V^{(d)}, E^{(d)})$ created by the Dilated k-NN rather than the original graph G.
- To improve generalization, they us *stochastic dilation* for training (but not inference):
  - With probability $(1 - \epsilon)$ they perform a dilated aggregation.
  - With small probability ϵ, they uniformly at random sample k neighbours from $(u_1, u_2, \ldots, u_{k \times d})$.
  - Improves generalization.

Experiments

Setup

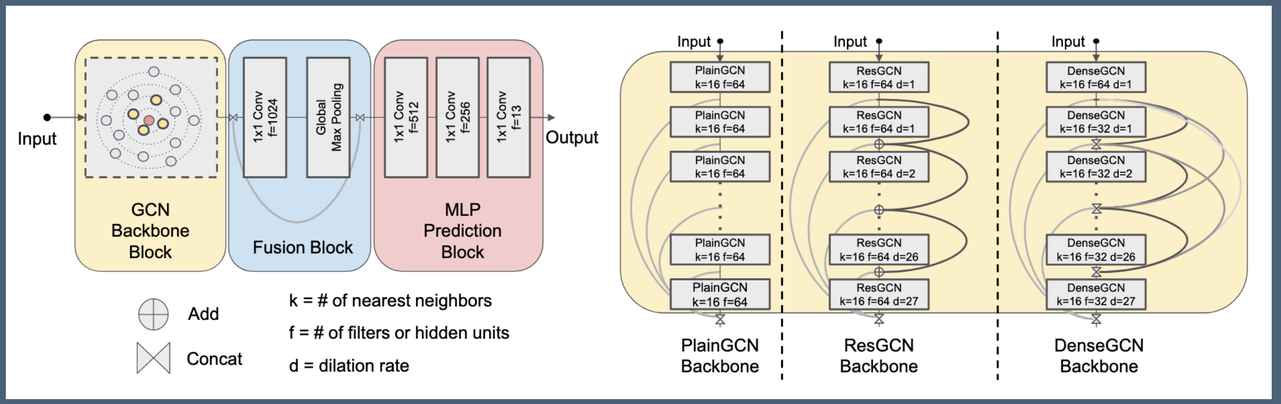
- Data: 3D point clouds. (S3DIS dataset)
  - Each point in the cloud is a vertex. Features are spatial coordinates and auxiliary features like color and surface normal.
  - They use a k-NN to construct the edges at every GCN layer.
- Metrics:
  - Overall accuracy (OA).
  - Mean intersection over union (mIoU): $\frac{TP}{TP + FP + FN}$. (TP is true positive, FP is false positive and FN is false negative).

Minor Issue

- In the paper, they define IoU as $\frac{TP}{TP + T - P}$ where T is the number of ground truth point of that class and P is the number of predicted positive points.
- But $\frac{TP}{TP + FP + FN} \neq \frac{TP}{TP + T - P}$, since $T = TP + FN$, and $P = TP + FP$.
- $\frac{TP}{TP + T - P} = \frac{TP}{TP - FP + FN} \neq \frac{TP}{TP + FP + FN}$

Experiments

Model details: Backbones

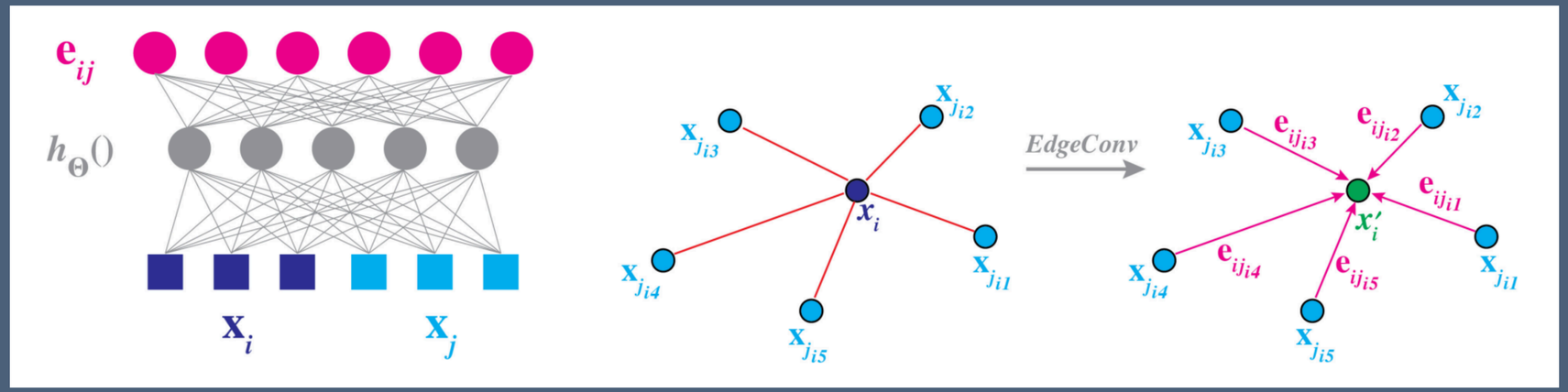


- **PlainGCN:** 28 EdgeConv layers with dynamic k-NN.
- **ResGCN:** Adds dynamic dilated k-NN and residual graph connections to PlainGCN.
- **DenseGCN:** Adds dynamic dilated k-NN and dense graph connections to PlainGCN.

- EdgeConv (Wang et al.): dynamic edge convolution algorithm for point clouds.

EdgeConv

(Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein,
and J. M. Solomon. Dynamic graph cnn for learning on point clouds.
arXiv preprint arXiv:1801.07829, 2018. 1, 2, 3, 5, 6, 7, 8, 11, 12)



- Given a learnable function h and two vertices x_i, x_j connected by an edge, we compute the edge feature $e_{ij} = h(x_i, x_j)$.

- EdgeConv aggregates the edge features for a given vertex to compute the new features of the vertex: $x'_i = Agg_{j:(i,j)\in E}(h(x_i, x_j))$

Ablation	Model	mIoU	ΔmIoU	dynamic	connection	dilation	stochastic	# NNs	# filters	# layers
Reference	*ResGCN-28*	**52.49**	0.00	✓	⊕	✓	✓	16	64	28
Dilation		51.98	-0.51	✓	⊕	✓		16	64	28
		49.64	-2.85	✓	⊕			16	64	28
	PlainGCN-28	**40.31**	-12.18	✓				16	64	28
Fixed k-NN		48.38	-4.11		⊕			16	64	28
		43.43	-9.06					16	64	28
Connections	*DenseGCN-28*	**51.27**	-1.22	✓	⋈	✓	✓	8	32	28
		40.47	-12.02	✓		✓	✓	16	64	28
		38.79	-13.70	✓		✓	✓	8	64	56
		49.23	-3.26	✓		✓	✓	16	64	14
		47.92	-4.57	✓		✓	✓	16	64	7
Neighbors		49.98	-2.51	✓	⊕	✓	✓	8	64	28
		49.22	-3.27	✓	⊕	✓	✓	4	64	28
Depth	*ResGCN-56*	**53.64**	1.15	✓	⊕	✓	✓	8	64	56
	ResGCN-14	49.90	-2.59	✓	⊕	✓	✓	16	64	14
	ResGCN-7	48.95	-3.53	✓	⊕	✓	✓	16	64	7
Width	*ResGCN-28W*	**53.78**	1.29	✓	⊕	✓	✓	8	128	28
		49.18	-3.31	✓	⊕	✓	✓	32	32	28
		48.80	-3.69	✓	⊕	✓	✓	16	32	28
		45.62	-6.87	✓	⊕	✓	✓	16	16	28

Table 1. **Ablation study on area 5 of S3DIS**. We compare our reference network (*ResGCN-28*) with 28 layers, residual graph connections, and dilated graph convolutions to several ablated variants. All models were trained with the same hyper-parameters for 100 epochs on all areas except for area 5, which is used for evaluation. We denote residual and dense connections with the ⊕ and ⋈ symbols respectively. We highlight the most important results in bold. ΔmIoU denotes the difference in mIoU with respect to the reference model *ResGCN-28*.

What affects performance?

Ablation studies

- Residual graph connections
- Dilation
- Dynamic k-NN
- Dense graph connections
- Nearest neighbors
- Network depth
- Network width

Residual connections

Ablation	Model	mIoU	ΔmIoU	dynamic	connection	dilation	stochastic	# NNs	# filters	# layers
Reference	*ResGCN-28*	**52.49**	0.00	✓	⊕	✓	✓	16	64	28
Fixed k-NN		48.38	-4.11		⊕			16	64	28
		43.43	-9.06					16	64	28

- Adding residual connections increases mIoU by 11.4%.
- It seems to account for about 45% of the performance benefit compared to the reference that has dynamic k-NN, stochastic dilation.

Dilation

Ablation	Model	mIoU	ΔmIoU	dynamic	connection	dilation	stochastic	# NNs	# filters	# layers
Reference	*ResGCN-28*	**52.49**	0.00	✓	⊕	✓	✓	16	64	28
Dilation		51.98	-0.51	✓	⊕	✓		16	64	28
		49.64	-2.85	✓	⊕			16	64	28
	PlainGCN-28	**40.31**	-12.18	✓				16	64	28
Fixed k-NN		48.38	-4.11		⊕			16	64	28
		43.43	-9.06					16	64	28

- Dilation improves performance 4.7% (row 2 vs row 3).
- Stochastic dilation further improves performance by 1%.
- However these improvements are only seen for deep networks with residual connections. (Hard to say how they draw that conclusion as they don't compare shallow networks with and without dilation).

Dynamic k-NN

Ablation	Model	mIoU	ΔmIoU	dynamic	connection	dilation	stochastic	# NNs	# filters	# layers
Reference	*ResGCN-28*	**52.49**	0.00	✓	⊕	✓	✓	16	64	28
Dilation		51.98	-0.51	✓	⊕	✓		16	64	28
		49.64	-2.85	✓	⊕			16	64	28
	PlainGCN-28	**40.31**	-12.18	✓				16	64	28
Fixed k-NN		48.38	-4.11		⊕			16	64	28
		43.43	-9.06					16	64	28

- With residual connections, adding dynamic k-NN improves performance by 2.6%.
- However, without them, it degrades performance by 7.2%.
- Comes at a relatively high computational cost: have to recompute the neighbours for every layer.

Dense connections

Ablation	Model	mIoU	ΔmIoU	dynamic	connection	dilation	stochastic	# NNs	# filters	# layers
Reference	*ResGCN-28*	**52.49**	0.00	✓	⊕	✓	✓	16	64	28
Connections	*DenseGCN-28*	**51.27**	-1.22	✓	⋈	✓	✓	8	32	28
		40.47	-12.02	✓		✓	✓	16	64	28
		38.79	-13.70	✓		✓	✓	8	64	56
		49.23	-3.26	✓		✓	✓	16	64	14
		47.92	-4.57	✓		✓	✓	16	64	7

- Prohibitive memory cost means they can only use a smaller network size.
- Achieves slightly worse performance (2.4%) than using residual connections, at a higher memory cost.
- As such, they conclude that residual connections are more practical.

Nearest neighbours

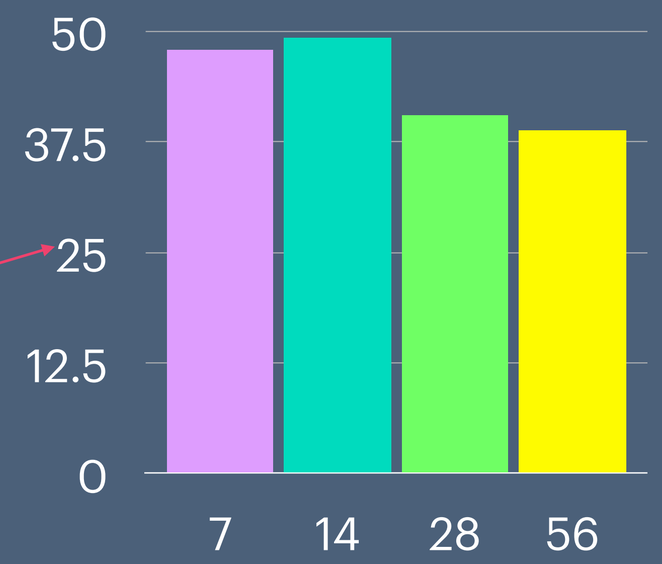
Ablation	Model	mIoU	ΔmIoU	dynamic	connection	dilation	stochastic	# NNs	# filters	# layers
Reference	*ResGCN-28*	**52.49**	0.00	✓	⊕	✓	✓	16	64	28
Neighbors		49.98	-2.51	✓	⊕	✓	✓	8	64	28
		49.22	-3.27	✓	⊕	✓	✓	4	64	28
		48.80	-3.69	✓	⊕	✓	✓	16	32	28
		45.62	-6.87	✓	⊕	✓	✓	16	16	28

- More neighbours improve performance for larger models.
- However, this is only true if the network capacity is high enough.

Network depth

Ablation	Model	mIoU	ΔmIoU	dynamic	connection	dilation	stochastic	# NNs	# filters	# layers
Reference	*ResGCN-28*	**52.49**	0.00	✓	⊕	✓	✓	16	64	28
Depth	*ResGCN-56*	**53.64**	1.15	✓	⊕	✓	✓	8	64	56
	ResGCN-14	49.90	-2.59	✓	⊕	✓	✓	16	64	14
	ResGCN-7	48.95	-3.53	✓	⊕	✓	✓	16	64	7
Connections	*DenseGCN-28*	**51.27**	-1.22	✓	⋈	✓	✓	8	32	28
		40.47	-12.02	✓		✓	✓	16	64	28
		38.79	-13.70	✓		✓	✓	8	64	56
		49.23	-3.26	✓		✓	✓	16	64	14
		47.92	-4.57	✓		✓	✓	16	64	7

- Depth improves performance when using residual connections.
- However, this is not necessarily true for models without residual connections.



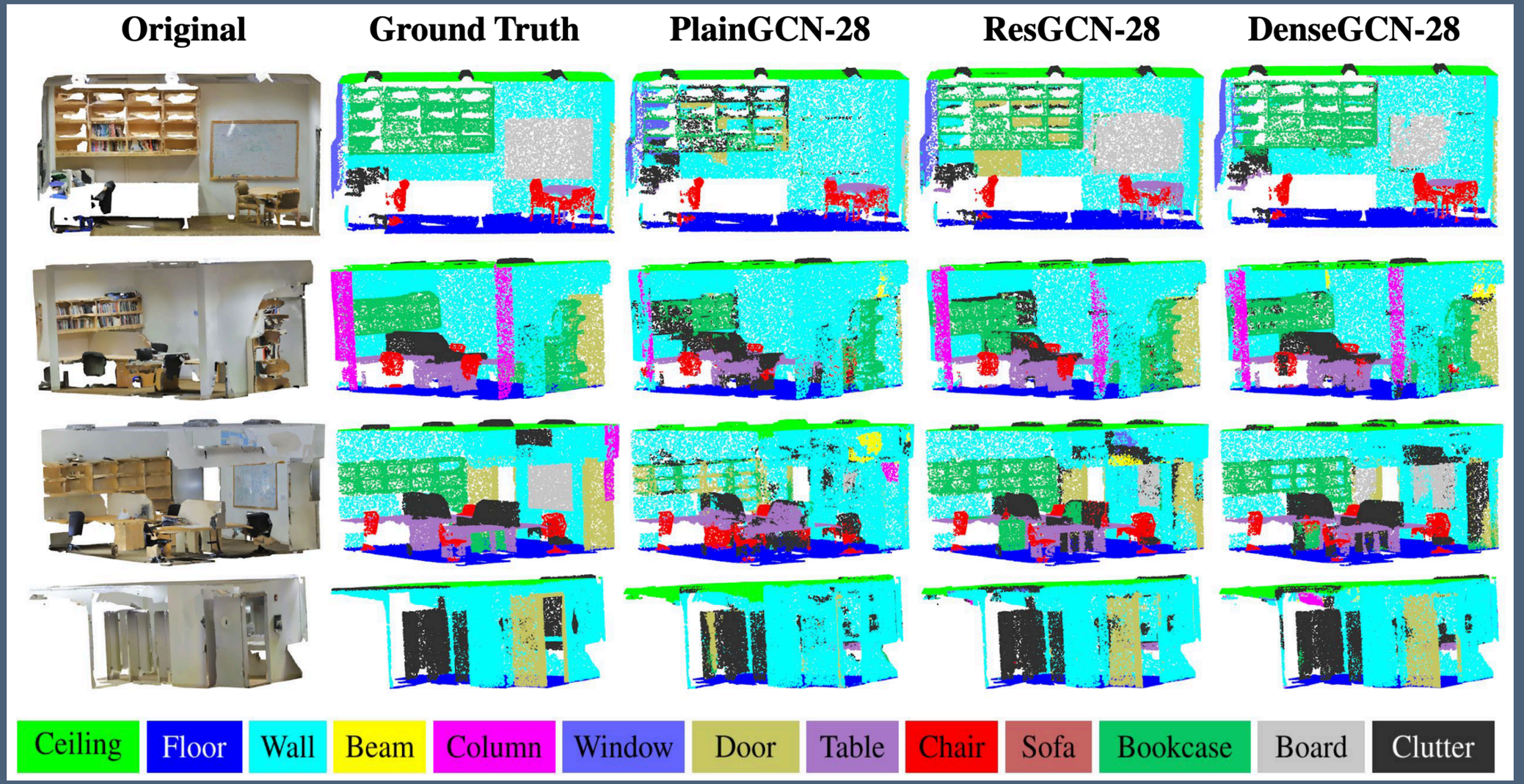
Network width

Ablation	Model	mIoU	ΔmIoU	dynamic	connection	dilation	stochastic	# NNs	# filters	# layers
Reference	*ResGCN-28*	**52.49**	0.00	✓	⊕	✓	✓	16	64	28
Width	*ResGCN-28W*	**53.78**	1.29	✓	⊕	✓	✓	8	128	28
		49.18	-3.31	✓	⊕	✓	✓	32	32	28
		48.80	-3.69	✓	⊕	✓	✓	16	32	28
		45.62	-6.87	✓	⊕	✓	✓	16	16	28

- Similar behaviour to increasing network depth: higher network capacity improves performance when using residual connections.

Semantic Segmentation

Qualitative analysis



Comparison to SOTA

Semantic Segmentation

Method	OA	mIOU	ceiling	floor	wall	beam	column	window	door	table	chair	sofa	bookcase	board	clutter
PointNet [27]	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++ [29]	-	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 [49]	**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 [42]	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**

Thank you !
Any questions ?