# Semantic Segmentation for Point Clouds

Yiqun, Lin

Supervised by Prof. Han

Aug. 22, 2019

# 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks

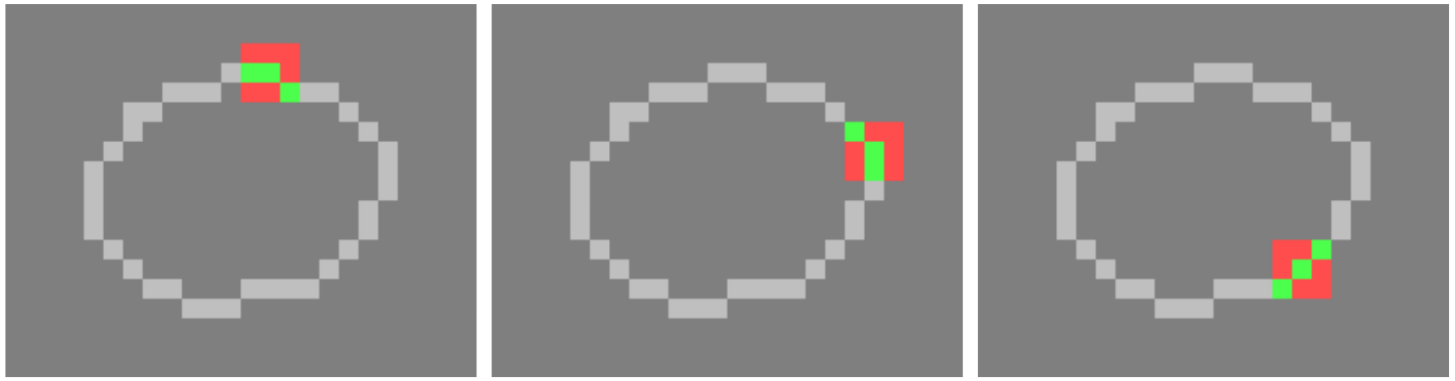
Benjamin Graham
Facebook AI Research
benjamingraham@fb.com

Martin Engelcke*
University of Oxford
martin@robots.ox.ac.uk

Laurens van der Maaten
Facebook AI Research
lvdmaaten@fb.com

## Highlights:

- 3-D voxel convolution
- Submanifold sparse convolution
- Remain sparsity



[paper] [code]

# Submanifold Dilation Problem

- The number of *active sites* grows very rapidly.
- Sparsity of the grid disappears rapidly when applying regular convolution on sparse data.
- Dense data costs much more memory and computation when convolution.

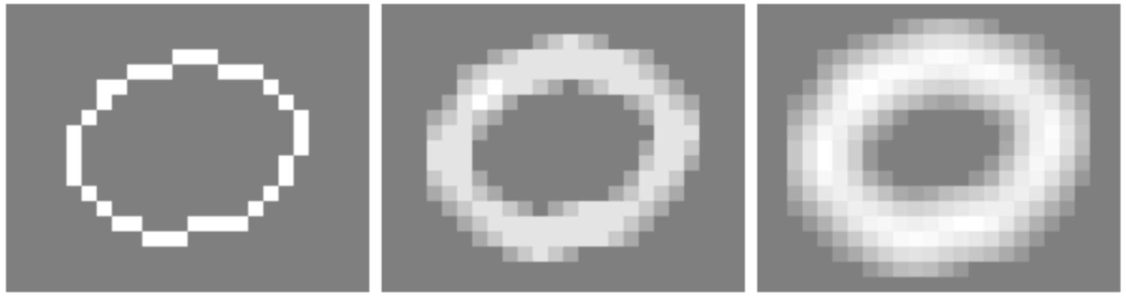
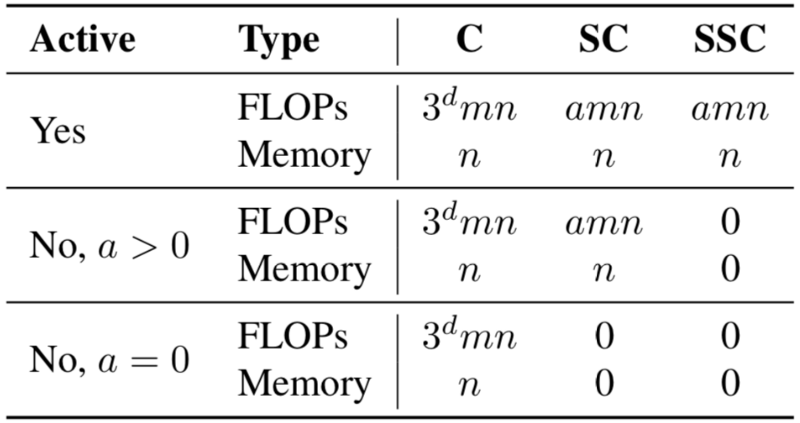


Figure 2: Example of "submanifold" dilation. **Left:** Original curve. **Middle:** Result of applying a regular $3 \times 3$ convolution with weights $1/9$. **Right:** Result of applying the same convolution again. Regular convolutions substantially reduce the feature sparsity with each convolutional layer.

## Submanifold Sparse Convolution

- **Regular Convolution**: convolve in each kernel location
- **Sparse Convolution**: convolve in only *active* location
- **Submanifold Sparse Convolution**: apply sparse convolution **iff** central location of the kernel is *active*.
- Other operators: pooling, batch normalization, …

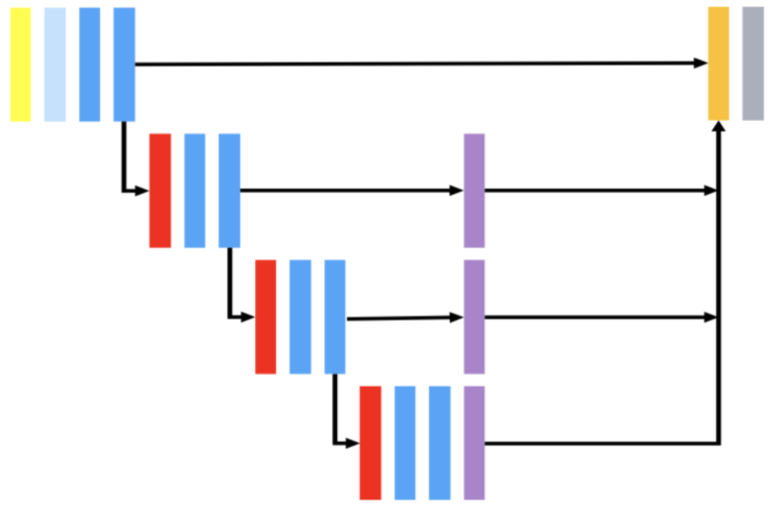


| **Active** | **Type** | **C** | **SC** | **SSC** |
|---|---|---|---|---|
| Yes | FLOPs | $3^d mn$ | $amn$ | $amn$ |
| | Memory | $n$ | $n$ | $n$ |
| No, $a > 0$ | FLOPs | $3^d mn$ | $amn$ | $0$ |
| | Memory | $n$ | $n$ | $0$ |
| No, $a = 0$ | FLOPs | $3^d mn$ | $0$ | $0$ |
| | Memory | $n$ | $0$ | $0$ |

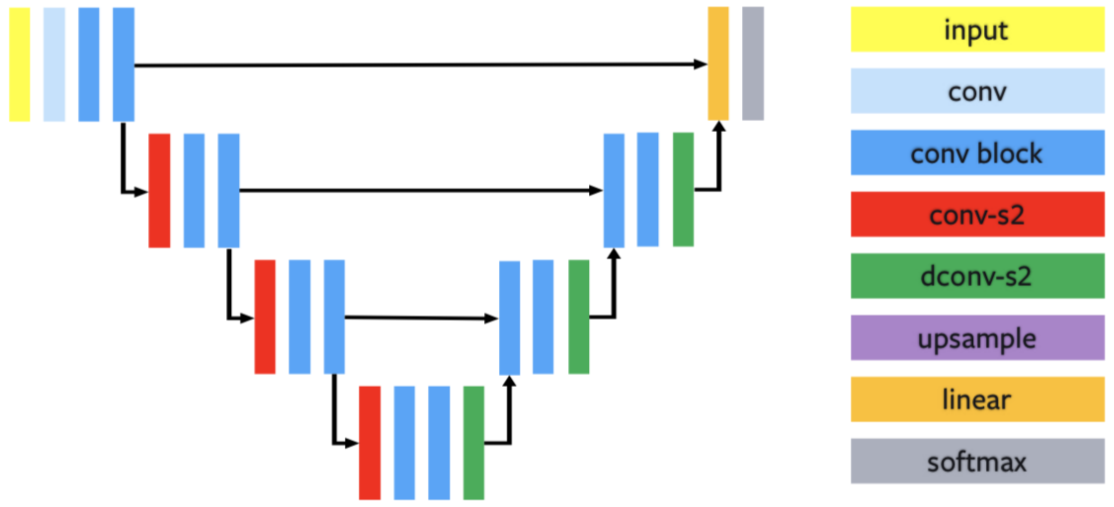
Table 2: Computational and memory requirements of three convolutions: regular convolution (C), sparse convolution (SC), and submanifold sparse convolution (SSC). We consider convolutions with size $f = 3$ and padding $s = 1$ at a single location in $d$ dimensions. Herein, $a$ is the number of active inputs to the spatial location, $m$ the number of input feature planes, and $n$ the number of output feature planes.

## Architectures

- SSC can be easily applied on different network architectures.
- Submanifold sparse FCN
- Submanifold sparse U-Net



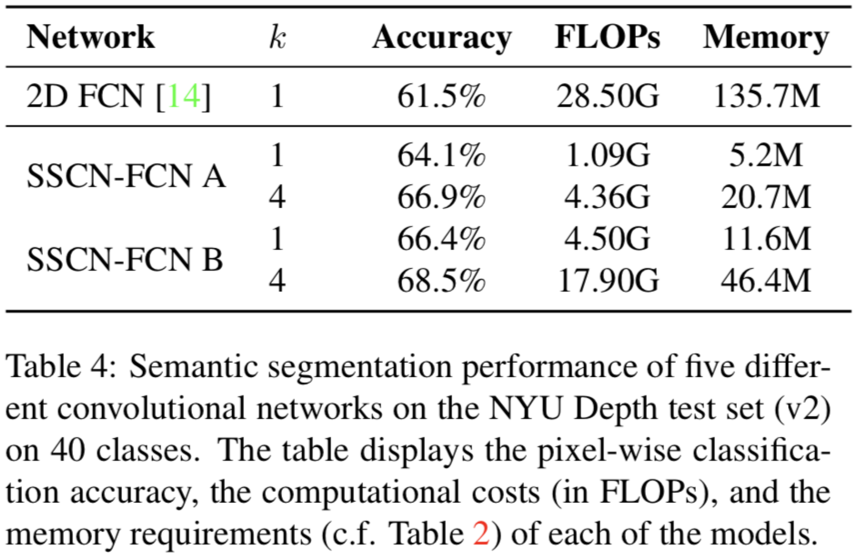
(a) Submanifold sparse FCN.



(b) Submanifold sparse U-Net.

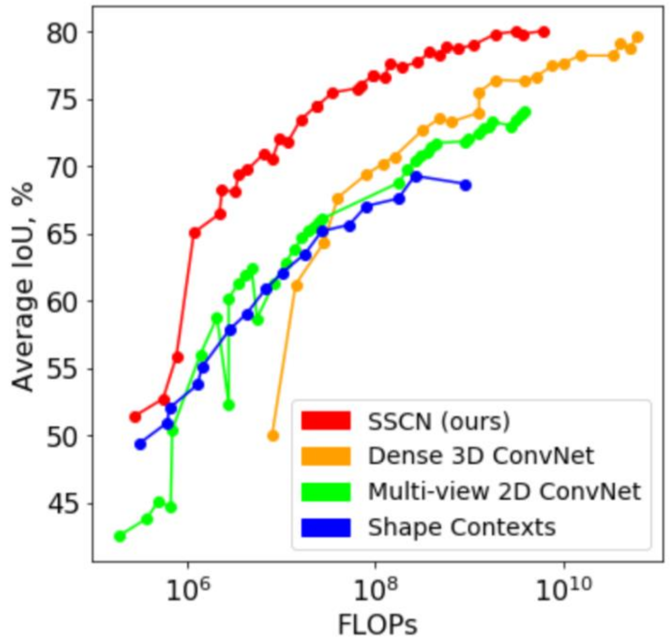
## Conclusions

- High-dimensional sparse convolution or feature extractors (not only for 3-D data)
- Computationally efficient and help the networks go deeper



| Network | $k$ | Accuracy | FLOPs | Memory |
|---|---|---|---|---|
| 2D FCN [14] | 1 | 61.5% | 28.50G | 135.7M |
| SSCN-FCN A | 1 | 64.1% | 1.09G | 5.2M |
| | 4 | 66.9% | 4.36G | 20.7M |
| SSCN-FCN B | 1 | 66.4% | 4.50G | 11.6M |
| | 4 | 68.5% | 17.90G | 46.4M |

Table 4: Semantic segmentation performance of five different convolutional networks on the NYU Depth test set (v2) on 40 classes. The table displays the pixel-wise classification accuracy, the computational costs (in FLOPs), and the memory requirements (c.f. Table 2) of each of the models.



(a) Comparison with baseline methods.

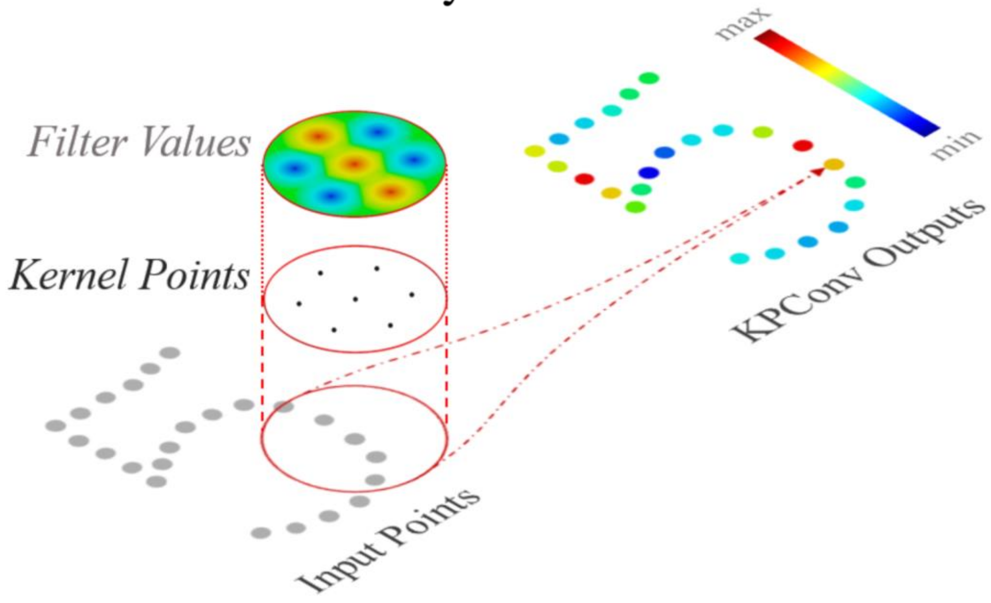
# KPConv: Flexible and Deformable Convolution for Point Clouds

Hugues Thomas[1] Charles R. Qi[2] Jean-Emmanuel Deschaud[1] Beatriz Marcotegui[1]
François Goulette[1] Leonidas J. Guibas[2,3]

[1]Mines ParisTech [2]Facebook AI Research [3]Stanford University

**Highlights**:

- Use kernel points for convolution
- Interpolate convolution weights
- Deformable kernel: add a learnable shift



[paper] [code]

# Kernel Point Convolution

- Use points to carry weights
- Interpolate convolution weights for each neighbor points

$$(\mathcal{F} * g)(x) = \sum_{x_i \in \mathcal{N}_x} g(x_i - x) f_i \qquad g(y_i) = \sum_{k<K} h(y_i, \widetilde{x}_k) W_k \qquad h(y_i, \widetilde{x}_k) = \max\left(0, 1 - \frac{\|y_i - \widetilde{x}_k\|}{\sigma}\right)$$

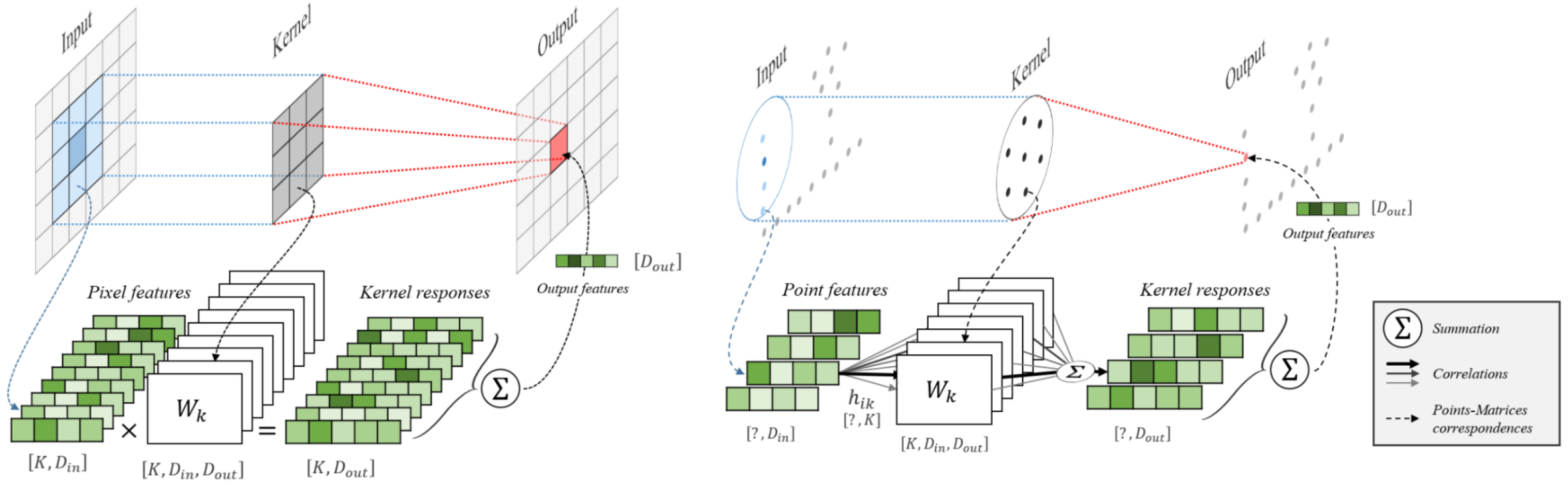
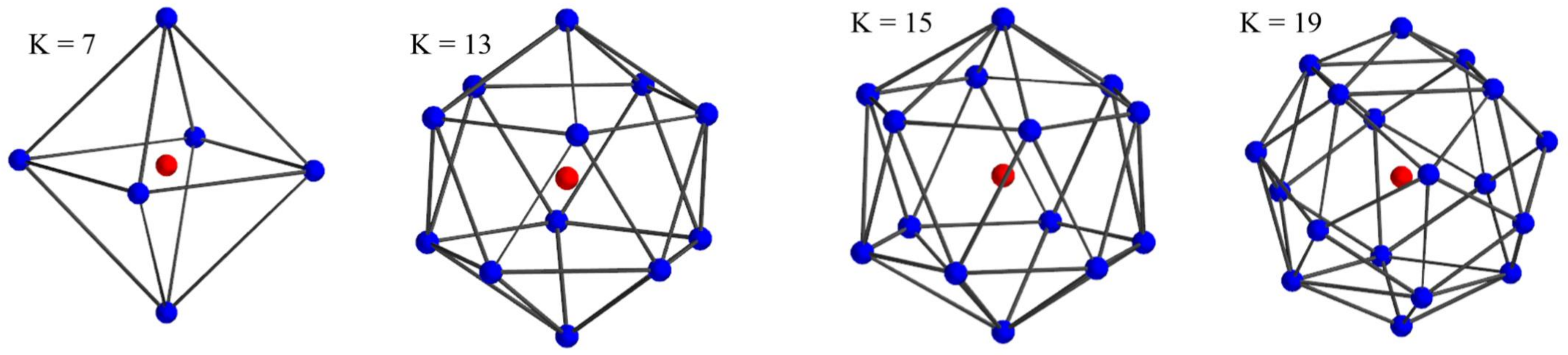


Figure 2. Comparison between an image convolution (left) and a KPConv (right) on 2D points for a simpler illustration. In the image, each pixel feature vector is multiplied by a weight matrix $(W_k)_{k<K}$ assigned by the alignment of the kernel with the image. In KPConv, input points are not aligned with kernel points, and their number can vary. Therefore, each point feature $f_i$ is multiplied by all the kernel weight matrices, with a correlation coefficient $h_{ik}$ depending on its relative position to kernel points.

# Rigid Kernel

Given a ***K***, place the kernel points by solving an optimization problem:

- each point applies a repulsive force on the others
- points are constrained to stay in the sphere with an attractive force
- one of them is constrained to be at the center

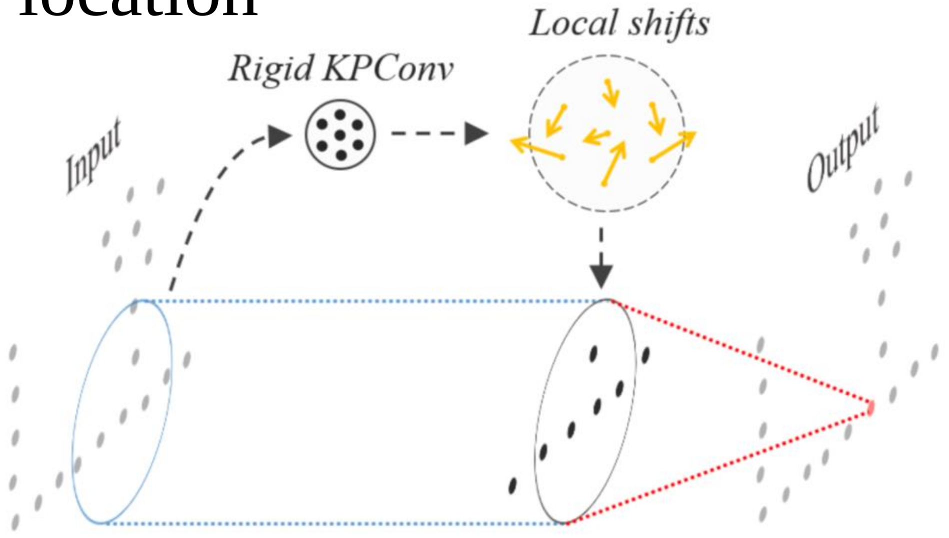


# Deformable Kernel

- Initialize the kernel with Rigid Kernel
- Add a **learnable** shift for each kernel point location

$$(\mathcal{F} * g)(x) = \sum_{x_i \in \mathcal{N}_x} g_{deform}(x - x_i, \Delta(x)) f_i$$

$$g_{deform}(y_i, \Delta(x)) = \sum_{k<K} h\left(y_i, \widetilde{x}_k + \Delta_k(x)\right) W_k$$



# Conclusions

- Convolution kernel uses points to carry weights
- Add linear correlation to interpolate convolution weights
- Deformable kernel to learn local shifts to fit the point cloud geometry distribution

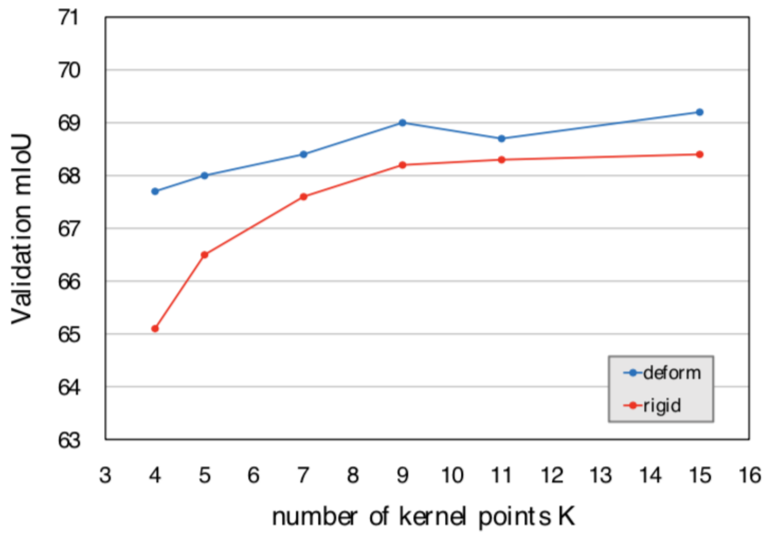
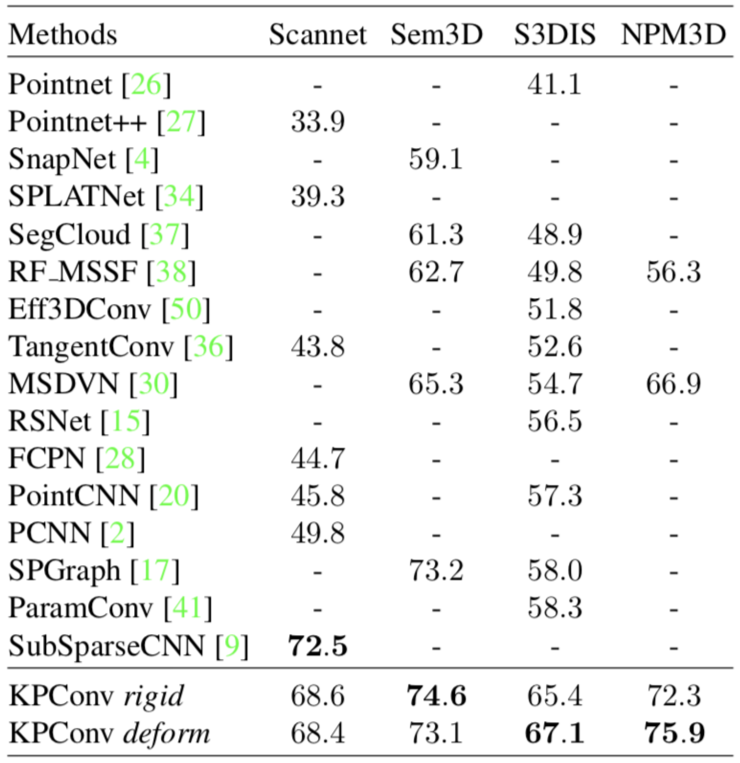


Figure 5. Ablation study on Scannet validation set. Evolution of the mIoU when reducing the number of kernel points.



| Methods | Scannet | Sem3D | S3DIS | NPM3D |
|---|---|---|---|---|
| Pointnet [26] | - | - | 41.1 | - |
| Pointnet++ [27] | 33.9 | - | - | - |
| SnapNet [4] | - | 59.1 | - | - |
| SPLATNet [34] | 39.3 | - | - | - |
| SegCloud [37] | - | 61.3 | 48.9 | - |
| RF_MSSF [38] | - | 62.7 | 49.8 | 56.3 |
| Eff3DConv [50] | - | - | 51.8 | - |
| TangentConv [36] | 43.8 | - | 52.6 | - |
| MSDVN [30] | - | 65.3 | 54.7 | 66.9 |
| RSNet [15] | - | - | 56.5 | - |
| FCPN [28] | 44.7 | - | - | - |
| PointCNN [20] | 45.8 | - | 57.3 | - |
| PCNN [2] | 49.8 | - | - | - |
| SPGraph [17] | - | 73.2 | 58.0 | - |
| ParamConv [41] | - | - | 58.3 | - |
| SubSparseCNN [9] | **72.5** | - | - | - |
| KPConv *rigid* | 68.6 | **74.6** | 65.4 | 72.3 |
| KPConv *deform* | 68.4 | 73.1 | **67.1** | **75.9** |

## 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks

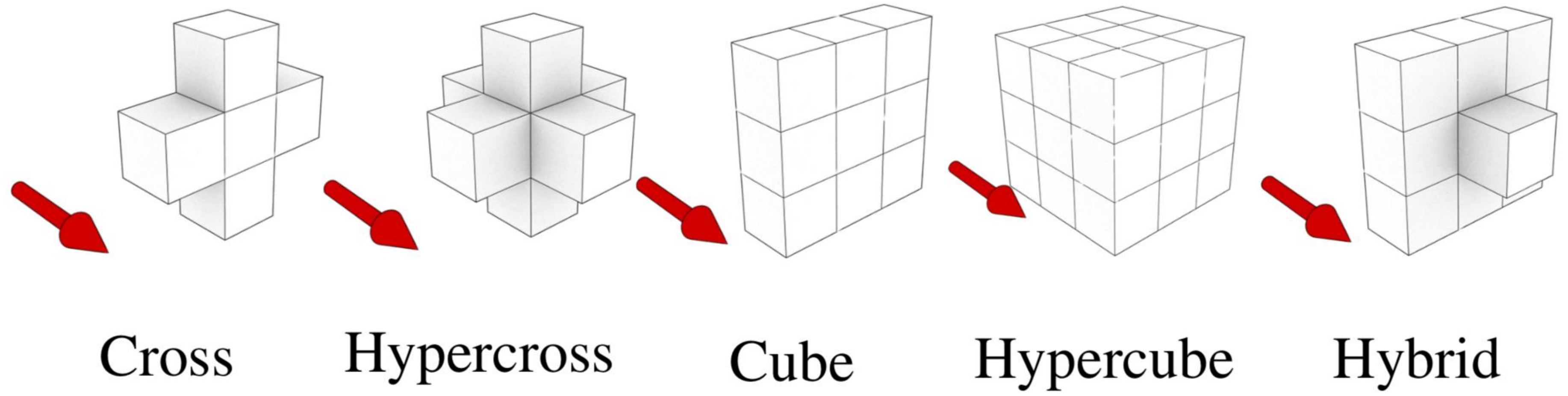
Christopher Choy
chrischoy@stanford.edu

JunYoung Gwak
jgwak@stanford.edu

Silvio Savarese
ssilvio@stanford.edu

**Highlights**:

- Generalized sparse convolution
- Hybrid kernel: combine cube kernel with cross kernel
- 7-D CRF: space(3)-time(1)-color(3)



[paper] [code]

## Generalized Sparse Convolution

$$\mathbf{x}_{\mathbf{u}}^{\text{out}} = \sum_{\mathbf{i} \in \mathcal{N}^D(\mathbf{u}, \mathcal{C}^{\text{in}})} W_{\mathbf{i}} \mathbf{x}_{\mathbf{u}+\mathbf{i}}^{\text{in}} \text{ for } \mathbf{u} \in \mathcal{C}^{\text{out}}$$

- Dense convolution
- Submanifold sparse convolution
- Strided convolution
- Customized kernel: hypercube, hypercross, hybrid kernel

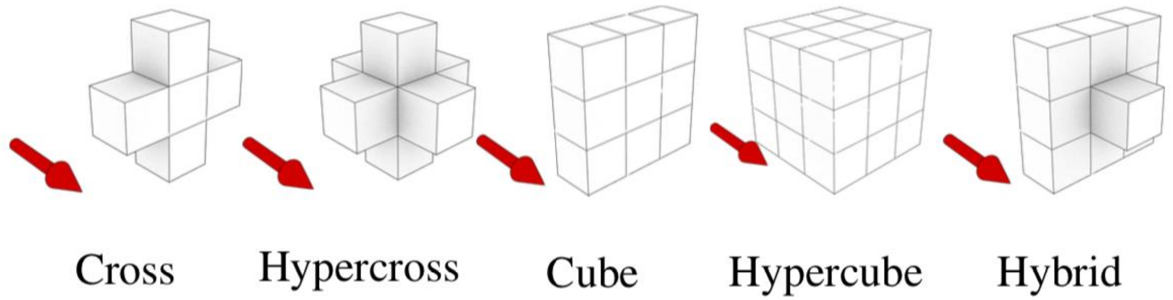


Figure 3: Various kernels in space-time. The red arrow indicates the temporal dimension and the other two axes are for spatial dimensions. The third spatial dimension is hidden for better visualization.

## Other Features

Spatial-Temporal Convolution:

- 3-D video dataset: Synthia 4D
- Use cube kernel on spatial axis and cross kernel on temporal axis

Trilateral Stationary CRF

- 7-D space: spatial(3)-time(1)-color(3)
- Learn CRF with 7-D sparse convolution

# Conclusions

- High-dimensional sparse convolution
- Customized kernel: hybridize cube and cross kernel
- Use spatial-temporal kernel instead of RNN for 4-D analysis
- Deeper network architecture

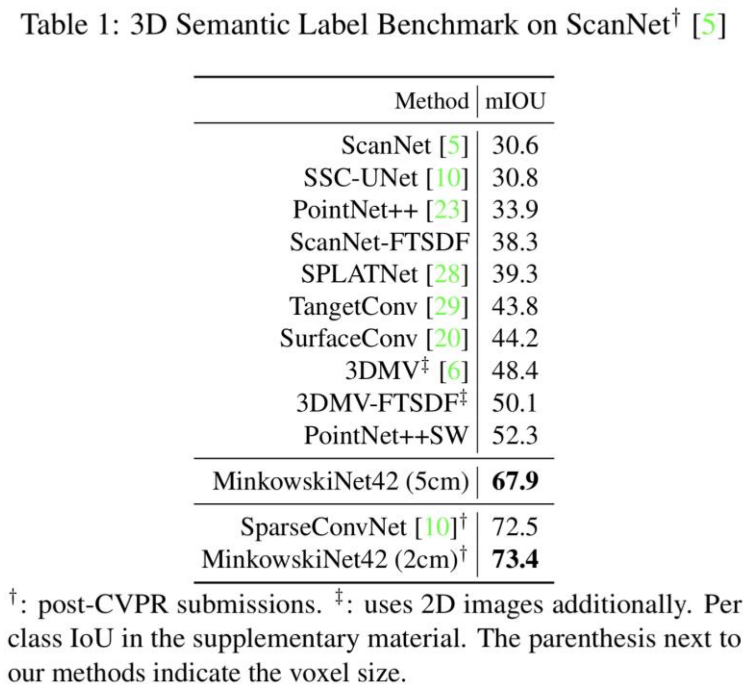
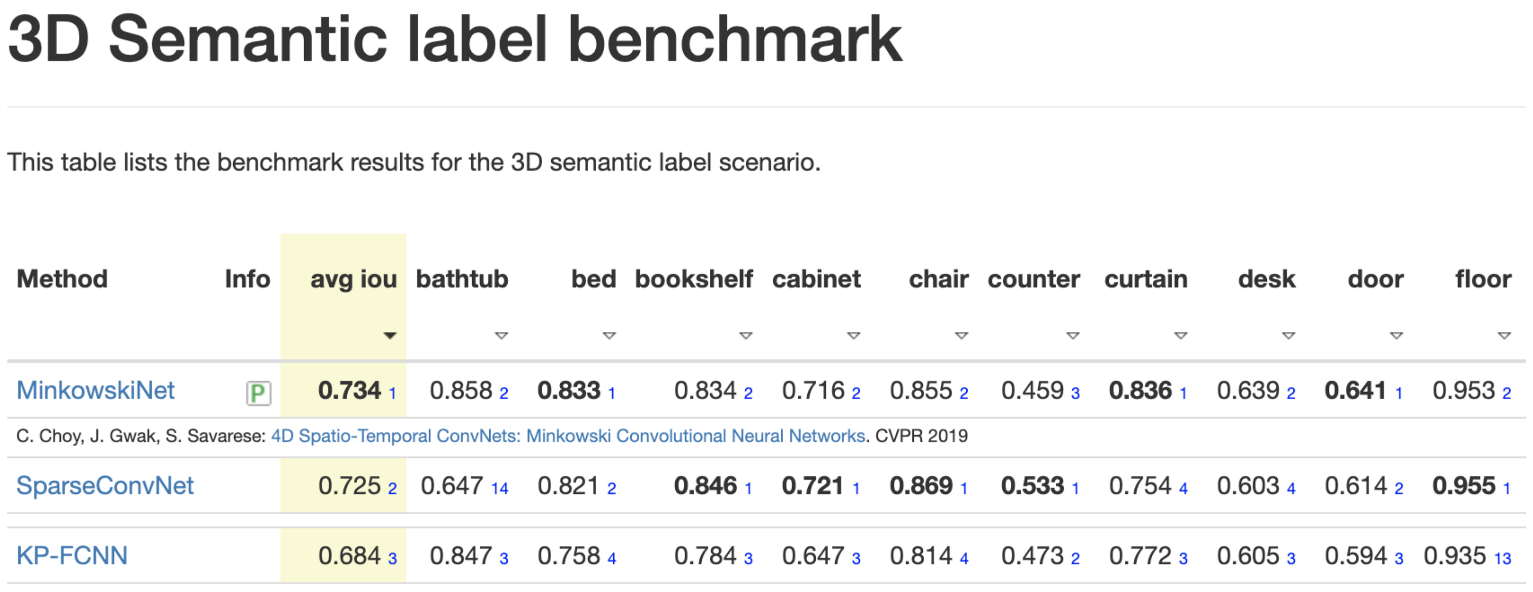


Table 1: 3D Semantic Label Benchmark on ScanNet† [5]

| Method | mIOU |
|---|---|
| ScanNet [5] | 30.6 |
| SSC-UNet [10] | 30.8 |
| PointNet++ [23] | 33.9 |
| ScanNet-FTSDF | 38.3 |
| SPLATNet [28] | 39.3 |
| TangetConv [29] | 43.8 |
| SurfaceConv [20] | 44.2 |
| 3DMV‡ [6] | 48.4 |
| 3DMV-FTSDF‡ | 50.1 |
| PointNet++SW | 52.3 |
| MinkowskiNet42 (5cm) | **67.9** |
| SparseConvNet [10]† | 72.5 |
| MinkowskiNet42 (2cm)† | **73.4** |

†: post-CVPR submissions. ‡: uses 2D images additionally. Per class IoU in the supplementary material. The parenthesis next to our methods indicate the voxel size.



## 3D Semantic label benchmark

This table lists the benchmark results for the 3D semantic label scenario.

| Method | Info | avg iou | bathtub | bed | bookshelf | cabinet | chair | counter | curtain | desk | door | floor |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MinkowskiNet | P | **0.734** 1 | 0.858 2 | **0.833** 1 | 0.834 2 | 0.716 2 | 0.855 2 | 0.459 3 | **0.836** 1 | 0.639 2 | **0.641** 1 | 0.953 2 |
| C. Choy, J. Gwak, S. Savarese: 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. CVPR 2019 | | | | | | | | | | | | |
| SparseConvNet | | 0.725 2 | 0.647 14 | 0.821 2 | **0.846** 1 | **0.721** 1 | **0.869** 1 | **0.533** 1 | 0.754 4 | 0.603 4 | 0.614 2 | **0.955** 1 |
| KP-FCNN | | 0.684 3 | 0.847 3 | 0.758 4 | 0.784 3 | 0.647 3 | 0.814 4 | 0.473 2 | 0.772 3 | 0.605 3 | 0.594 3 | 0.935 13 |