3D CNNs

Zhile Ren
Postdoc @ Georgia Tech
http://cs.brown.edu/people/zr1/
3D Deep Learning (on point sets)

Zhile Ren
Postdoc @ Georgia Tech
http://cs.brown.edu/people/zr1/
Scene Understanding on RGB Images

[Image courtesy: Koltun, Russakovsky, Gupta]
3D Scene Understanding

- Robotics
- Augmented Reality
- Autonomous driving
- Medical Image Processing
3D Deep Learning Tasks

3D geometry analysis

Classification

Parsing (object/scene)

Correspondence
3D Deep Learning Tasks

3D synthesis

Monocular 3D reconstruction  Shape completion  Shape modeling
The Lack of 3D Data

Status as of 2010:

- Stanford bunny
- Utah teapot
- 1800 models in 90 categories
- Princeton shape benchmark [Shilane et al. 04]
3D Repositories on the Internet

Growing market of crowd-sourcing for 3D modeling

- Yobi 3D
- Clara.io
- Google SketchUp
- Sketchfab
- TinkerCAD
Growing market of crowd-sourcing for 3D modeling

Mid Century Modern Desk
Deep Learning on RGB Images

Images: Unique representation with regular data structure

![Image of a woman with hat]

```
1  44  33  12  20  23  35  14
51  16  40  32  46  48  28  17
29  60  3  63  49  55  36  7
52  22  26  41  38  10  61  53
  2  24  19  11  34  43  5  8
57  9  37  42  25  21  27  18
30  56  50  64  4  59  6  13
58  47  45  31  39  15  62  54
```
3D Representations

Point cloud

3D Mesh

3D Voxels
3D CNNs on Voxelized Data

3DShapeNets from Princeton CVPR 2015

VoxNet from CMU Robotics IEEE/RSJ 2015

Information loss in voxelization

CAD model
Occupancy Grid 30x30x30

Rendering + 2D CNN

90.1%

MVCNN from UMass ICCV 2015
3D ShapeNet

3D CNNs applied to 3D input
Unordered Point Set

Lidar scan from autonomous vehicles
(Deep) Learning on 3D point sets

This Lecture:
- PointNet [Qi et al., CVPR 2017]
- PointNet++ [Qi et al., NeurIPS 2017]
- SPLATNET [Su et al., CVPR 2018]

Most slides in this lecture are from: Hao Su, Charles Qi, Hang Su
(Deep) Learning on 3D point sets

This Lecture:
- PointNet [Qi et al., CVPR 2017]
- PointNet++ [Qi et al., NeurIPS 2017]
- SPLATNET [Su et al., CVPR 2018]

Most slides in this lecture are from: Hao Su, Charles Qi, Hang Su
End-to-end learning for **scattered, unordered** point data
PointNet

End-to-end learning for **scattered, unordered** point data

**Unified** framework for various tasks

- Object Classification
- Object Part Segmentation
- Semantic Scene Parsing
- ...
PointNet

End-to-end learning for **scattered, unordered** point data

**Unified** framework for various tasks
Unordered point set as input

Model needs to be invariant to N! permutations.

Invariance under geometric transformations

Point cloud rotations should not alter classification results.
Unordered point set as input

Model needs to be invariant to $N!$ permutations.

Invariance under geometric transformations

Point cloud rotations should not alter classification results.
Unordered Input

Point cloud: $N$ orderless points, each represented by a $D$-dim vector
Unordered Input

Point cloud: N orderless points, each represented by a D dim vector

represents the same set as
Unordered Input

Point cloud: N orderless points, each represented by a D dim vector

N represents the same set as

Model needs to be invariant to N! permutations
Permutation Invariance: Symmetric Function

\[ f(x_1, x_2, \ldots, x_n) \equiv f(x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}), \quad x_i \in \mathbb{R}^D \]
Permutation Invariance: Symmetric Function

\[ f(x_1, x_2, \ldots, x_n) \equiv f(x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}), \quad x_i \in \mathbb{R}^D \]

Examples:

\[ f(x_1, x_2, \ldots, x_n) = \max \{x_1, x_2, \ldots, x_n\} \]
\[ f(x_1, x_2, \ldots, x_n) = x_1 + x_2 + \ldots + x_n \]

...
Permutation Invariance: Symmetric Function

\[ f(x_1, x_2, \ldots, x_n) \equiv f(x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}), \quad x_i \in \mathbb{R}^D \]

Examples:

\[ f(x_1, x_2, \ldots, x_n) = \max \{x_1, x_2, \ldots, x_n \} \]

\[ f(x_1, x_2, \ldots, x_n) = x_1 + x_2 + \ldots + x_n \]

\[ \vdots \]

How can we construct a family of symmetric functions by neural networks?
Permutation Invariance: Symmetric Function

Observe:

\[ f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n)) \] is symmetric if \( g \) is symmetric.
Observe:

\[ f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n)) \]

is symmetric if \( g \) is symmetric.

- \( h(1, 2, 3) \)
- \( h(1, 1, 1) \)
- \( h(2, 3, 2) \)
- \( \vdots \)
- \( h(2, 3, 4) \)
Permutation Invariance: Symmetric Function

Observe:

\[ f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n)) \]

is symmetric if \( g \) is symmetric.
Permutation Invariance: Symmetric Function

Observe:

\[ f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n)) \] is symmetric if \( g \) is symmetric.

**PointNet (vanilla)**
What symmetric functions can be constructed by PointNet?
Universal Set Function Approximator

**Theorem:**
A Hausdorff continuous symmetric function \( f : 2^\mathcal{X} \to \mathbb{R} \) can be arbitrarily approximated by PointNet.

\[
\left| f(S) - \gamma \left( \operatorname{MAX}_{x_i \in S} \{ h(x_i) \} \right) \right| < \epsilon
\]

\( S \subseteq \mathbb{R}^d \)

**PointNet (vanilla)**
Empirically, we use multi-layer perceptron (MLP) and max pooling:

PointNet (vanilla)
Challenges

Unordered point set as input
Model needs to be invariant to N! permutations.

Invariance under geometric transformations
Point cloud rotations should not alter classification results.
Idea: Data dependent transformation for automatic alignment
The transformation is just matrix multiplication!

Input Alignment by Transformer Network

Data $\rightarrow$ T-Net $\rightarrow$ Matrix Mult. $\rightarrow$ Transformed Data
Embedding Space Alignment

T-Net

transform
params: $64 \times 64$

Input
embeddings:
$Nx64$

Matrix
Mult.

Transformed
embeddings:
$Nx64$
Embedding Space Alignment

T-Net

Transform params: 64x64

Matrix Mult.

Input embeddings: Nx64

Transformed embeddings: Nx64

Regularization:

Transform matrix $A_{64x64}$ close to orthogonal:

$$L_{reg} = \| I - AA^T \|^2_F$$
PointNet Classification Network
PointNet Classification Network
PointNet Classification Network
PointNet Classification Network
PointNet Classification Network

Input points: nx3
- Input transform: nx3
- MLP (64, 64)
  - Shared: nx64
  - Feature transform: nx64
- MLP (64, 128, 1024)
  - Shared: nx1024

T-Net: 3x3 transform
- Matrix multiply

T-Net: 64x64 transform
- Matrix multiply
PointNet Classification Network

input points→ input transform

nx3 → nx3

mlp (64,64)→ feature transform

nx64 → nx64

mlp (64,128,1024)→ max pool

nx1024 → global feature
PointNet Classification Network
Extension to PointNet Segmentation Network

input points

nx3

input transform

nx3

mlp (64,64)

feature transform

nx64

local embedding

nx64

mlp (64,128,1024)

nx1024

max pool

1024

global feature

output scores

T-Net

3x3 transform

matrix multiply

T-Net

64x64 transform

matrix multiply

mlp (512,256,k)
Extension to PointNet Segmentation Network
Results
Princeton ModelNet Dataset

A large-scale CAD dataset of more than 150,000 models in 660 categories.
## Results on Object Classification

<table>
<thead>
<tr>
<th>Model</th>
<th>Input</th>
<th>Views</th>
<th>Avg. Class Accuracy</th>
<th>Overall Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPH [12]</td>
<td>mesh</td>
<td>-</td>
<td>68.2</td>
<td></td>
</tr>
<tr>
<td>3DShapeNets [29]</td>
<td>volume</td>
<td>1</td>
<td>77.3</td>
<td>84.7</td>
</tr>
<tr>
<td>VoxNet [18]</td>
<td>volume</td>
<td>12</td>
<td>83.0</td>
<td>85.9</td>
</tr>
<tr>
<td>Subvolume [19]</td>
<td>volume</td>
<td>20</td>
<td>86.0</td>
<td><strong>89.2</strong></td>
</tr>
<tr>
<td>LFD [29]</td>
<td>image</td>
<td>10</td>
<td>75.5</td>
<td>-</td>
</tr>
<tr>
<td>MVCNN [24]</td>
<td>image</td>
<td>80</td>
<td><strong>90.1</strong></td>
<td>-</td>
</tr>
<tr>
<td>Ours baseline</td>
<td>point</td>
<td>-</td>
<td>72.6</td>
<td>77.4</td>
</tr>
<tr>
<td>Ours PointNet</td>
<td>point</td>
<td>1</td>
<td>86.2</td>
<td><strong>89.2</strong></td>
</tr>
</tbody>
</table>

*dataset: ModelNet40; metric: 40-class classification accuracy (%)*
Results on Object Part Segmentation
Results on Object Part Segmentation

<table>
<thead>
<tr>
<th></th>
<th>mean</th>
<th>aero</th>
<th>bag</th>
<th>cap</th>
<th>car</th>
<th>chair</th>
<th>ear</th>
<th>phone</th>
<th>guitar</th>
<th>knife</th>
<th>lamp</th>
<th>laptop</th>
<th>motor</th>
<th>mug</th>
<th>pistol</th>
<th>rocket</th>
<th>skate</th>
<th>board</th>
<th>table</th>
</tr>
</thead>
<tbody>
<tr>
<td># shapes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wu [28]</td>
<td>-</td>
<td>63.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>73.5</td>
<td>-</td>
<td>-</td>
<td>74.4</td>
<td>-</td>
<td>-</td>
<td>70.6</td>
<td>91.9</td>
<td>85.9</td>
<td>53.1</td>
<td>69.8</td>
<td>75.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yi [30]</td>
<td>81.4</td>
<td>81.0</td>
<td>78.4</td>
<td>77.7</td>
<td>75.7</td>
<td>87.6</td>
<td>61.9</td>
<td>92.0</td>
<td>85.4</td>
<td>82.5</td>
<td>95.7</td>
<td>70.6</td>
<td>91.9</td>
<td>85.9</td>
<td>53.1</td>
<td>69.8</td>
<td>75.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3DCNN</td>
<td>79.4</td>
<td>75.1</td>
<td>72.8</td>
<td>73.3</td>
<td>70.0</td>
<td>87.2</td>
<td>63.5</td>
<td>88.4</td>
<td>79.6</td>
<td>74.4</td>
<td>93.9</td>
<td>58.7</td>
<td>91.8</td>
<td>76.4</td>
<td>51.2</td>
<td>65.3</td>
<td>77.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours</td>
<td>83.7</td>
<td>83.4</td>
<td>78.7</td>
<td>82.5</td>
<td>74.9</td>
<td>89.6</td>
<td>73.0</td>
<td>91.5</td>
<td>85.9</td>
<td>80.8</td>
<td>95.3</td>
<td>65.2</td>
<td>93.0</td>
<td>81.2</td>
<td>57.9</td>
<td>72.8</td>
<td>80.6</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*dataset: ShapeNetPart; metric: mean IoU (%)*
Results on Semantic Scene Parsing

dataset: Stanford 2D-3D-S (Matterport scans)
Robustness to Data Corruption

dataset: ModelNet40; metric: 40-class classification accuracy (%)
Robustness to Data Corruption

Less than 2% accuracy drop with 50% missing data

dataset: ModelNet40; metric: 40-class classification accuracy (%)
Robustness to Data Corruption

dataset: ModelNet40; metric: 40-class classification accuracy (%)
Robustness to Data Corruption

Why is PointNet so robust to missing data?
Visualizing Global Point Cloud Features

Which input points are contributing to the global feature? (critical points)
Visualizing Global Point Cloud Features

Original Shape:

Critical Point Sets:
Conclusion

• PointNet is a novel deep neural network that directly consumes point cloud.
• A unified approach to various 3D recognition tasks.
• Rich theoretical analysis and experimental results.

Code & Data Available!  
http://stanford.edu/~rqi/pointnet  
See you at Poster 9!
(Deep) Learning on 3D point sets

This Lecture:
- PointNet [Qi et al., CVPR 2017]
- PointNet++ [Qi et al., NeurIPS 2017]
- SPLATNET [Su et al., CVPR 2018]

Most slides in this lecture are from: Hao Su, Charles Qi, Hang Su
Limitations of PointNet

Hierarchical feature learning
Multiple levels of abstraction

Global feature learning
Either one point or all points

3D CNN (Wu et al.)
PointNet (vanilla) (Qi et al.)
Limitations of PointNet

Hierarchical feature learning
Multiple levels of abstraction

Global feature learning
Either one point or all points

No local context for each point!

3D CNN (Wu et al.)

PointNet (vanilla) (Qi et al.)
Limitation of PointNet

Lack of local context => artifacts in segmentation tasks

Semantic segmentation of randomly translated table-cup scene

Instance mask prediction in table-chair-cup scene
Limitation of PointNet

Lack of local context => artifacts in segmentation tasks

Global feature depends on absolute coordinate. Hard to generalize to unseen scene configurations!
Use pointnet in local regions, aggregate local features by pointnet again

=> Hierarchical feature learning
PointNet++

Use pointnet in local regions, aggregate local features by pointnet again

=> Hierarchical feature learning

Common issue: inconsistent sampling density

=> Robust to non-uniform sampling density
Hierarchical Point Set Feature Learning

Sampling: Farthest Point Sampling (FPS)
Grouping: radius based ball query
Hierarchical Point Set Feature Learning

\[ (N, d+C) \rightarrow (N_1, K, d+C) \rightarrow (N_1, d+C_f) \]

Shared pointnet applied in each local region using local coord.
Hierarchical Point Set Feature Learning

Recursively apply pointnet:

\[(N, d+1, C) \rightarrow (N_1, K, d+1, C) \rightarrow (N_1, d, C_1) \rightarrow (N_2, K, d, C_1) \rightarrow (N_2, d, C_2)\]
PointNet layer v.s. Convolution layer

Input:
- Point set

Operation:
- PointNet (order invariant)

Neighborhood:
- Radius ball query (varying #points)

PointNet layer

Convolution layer

- Dense array
- Convolution (index-ordered)
  - Array index (fixed #pixel/voxel)
PointNet++ for Classification and Segmentation

*Hierarchical point set feature learning*

- Sampling & grouping
- Pointnet
- Set abstraction
- Sampling & grouping
- Pointnet
- Set abstraction

**Classification**

- Pointnet
- Fully connected layers
- Class scores

**Hierarchical point set feature learning**

\[
(N, d+C) \rightarrow (N, K, d+C) \rightarrow (N, d+C) \rightarrow (N, d+C) \rightarrow (N_2, d+C) \rightarrow (N_2, d+C) \rightarrow (N_2, d+C) 
\]

**Classification**

\[
(1, C_4) \rightarrow \text{fully connected layers} \rightarrow (k) \text{ class scores} 
\]
PointNet++ for Classification and Segmentation

**Hierarchical point set feature learning**

- Sampling & grouping
- Pointnet
- Sampling & grouping
- Pointnet
- Set abstraction
- Set abstraction

**Segmentation**

- Interpolation: inverted distance weighting.
- Unit pointnet: MLP on each point's feature

**Equations**

- Classification: $(N, C)$
- Segmentation: $(N, d+C)$, $(N, d+C)$, $(N, d+C)$, $(N, d+C)$, $(N, d+C)$, $(N, d+C)$, $(N, d+C)$, $(N, d+C)$, $(N, d+C)$, $(N, d+C)$, $(N, d+C)$, $(N, d+C)$, $(N, d+C)$
By hierarchical feature learning, we can achieve even better results than the original PointNet. By using normals, we can even beat best view-CNN based method.

<table>
<thead>
<tr>
<th>Method</th>
<th>Input</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subvolume [20]</td>
<td>vox</td>
<td>89.2</td>
</tr>
<tr>
<td>MVCNN [25]</td>
<td>img</td>
<td>90.1</td>
</tr>
<tr>
<td>PointNet (vanilla) [19]</td>
<td>pc</td>
<td>87.2</td>
</tr>
<tr>
<td>PointNet [19]</td>
<td>pc</td>
<td>89.2</td>
</tr>
<tr>
<td>Ours</td>
<td>pc</td>
<td>90.7</td>
</tr>
<tr>
<td>Ours (with normal)</td>
<td>pc</td>
<td><strong>91.9</strong></td>
</tr>
</tbody>
</table>

Table 2: ModelNet40 shape classification.
For organic shape recognition, PointNet++ can generalize to non-Euclidean space: intrinsic point features (HKS, WKS, Gaussian curvature) intrinsic distance metric (geodesic)

(a) Horse  (b) Cat  (c) Horse

<table>
<thead>
<tr>
<th>Metric space</th>
<th>Input feature</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepGM [13]</td>
<td>-</td>
<td>93.03</td>
</tr>
<tr>
<td>Euclidean</td>
<td>XYZ</td>
<td>60.18</td>
</tr>
<tr>
<td>Euclidean</td>
<td>Intrinsic features</td>
<td>94.49</td>
</tr>
<tr>
<td>Non-Euclidean</td>
<td>Intrinsic features</td>
<td><strong>96.09</strong></td>
</tr>
</tbody>
</table>

Table 3: SHREC15 Non-rigid shape classification.
PointNet++ Output

PointNet

Ours

Ground Truth

Wall
Floor
Chair
Desk
Bed
Door
Table
This Lecture:

- **PointNet** [Qi et al., CVPR 2017]
- **PointNet++** [Qi et al., NIPS 2017]
- **SPLATNET** [Su et al., CVPR 2018]

Most slides in this lecture are from: Hao Su, Charles Qi, Hang Su
Goal

Operate on point clouds directly

Two key capabilities:
1. Flexible specification of receptive fields
2. Joint 2D-3D processing
Bilateral Convolution Layer (BCL)

[Jampani et al. CVPR ’16] [Kiefel et al. ICLR ‘15 workshops]

Convolution on sparse unordered points

3 steps in BCL
Efficient computation

3 steps in BCL

1 Adams et al. Fast high-dimensional filtering using the permutohedral lattice. Computer Graphics Forum ‘10
### Point feature and lattice feature

<table>
<thead>
<tr>
<th></th>
<th>&quot;what&quot;</th>
<th>&quot;where&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Image</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>point feature</td>
<td>((r, g, b))</td>
<td>(f(\ldots))</td>
</tr>
<tr>
<td>lattice feature</td>
<td>((x, y))</td>
<td>((x, y))</td>
</tr>
<tr>
<td><strong>Point cloud</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>point feature</td>
<td>1</td>
<td>((x, y, z))</td>
</tr>
<tr>
<td>lattice feature</td>
<td>((x, y, z))</td>
<td>((x, y, z))</td>
</tr>
</tbody>
</table>
Controlling receptive fields

lattice scale
$\lambda_0$
$2 \cdot \lambda_0$
$3 \cdot \lambda_0$
lattice feature: $(x, y, z)$
lattice scale: $8 \cdot \lambda_0$

lattice feature: $(x, y, z)$
lattice scale: $\lambda_0$
Input and output can be at different points
Smooth projection between 2D and 3D

- Each 2D pixel $p$: $f(p), (x, y, z)$

2D features
Smooth projection between 2D and 3D

- Each 2D pixel $p$: $f(p)$, $(x, y, z)$
  - Point feature: $f(p)$
  - Lattice feature: $(x, y, z)$
SPLATNet architecture
**SPLATNet** architecture

**Input 3D point cloud**

![Input 3D point cloud]

**SPLATNet\textsubscript{3D}**

**Input images**

![Input images]

**SPLATNet\textsubscript{2D-3D}**

**3D prediction**

![3D prediction]

**2D predictions**

![2D predictions]
SPLATNet$_{3D}$

Input 3D point cloud

SPLATNet$_{3D}$

3D prediction
SPLATNet$_{3D}$

Input 3D point cloud

SPLATNet$_{3D}$

1×1 Conv

BCL $\lambda_0$

BCL $2 \cdot \lambda_0$

BCL $16 \cdot \lambda_0$

1×1 Conv

1×1 Conv

concatenation

3D prediction
SPLATNet$_{2D-3D}$

Input 3D point cloud

SPLATNet$_{3D}$

1X1 Conv

BCL $\lambda_0$

BCL $2 \cdot \lambda_0$

BCL $16 \cdot \lambda_0$

1X1 Conv

Input images

3D prediction

BCL $2D \rightarrow 3D$

BCL $3D \rightarrow 2D$

2D predictions
SPLATNet_{2D-3D}
$SPLATNet_{2D-3D}$
Experiments
## Facade segmentation (Ruemonge2014 [1])

### with only 3D data

<table>
<thead>
<tr>
<th>Method</th>
<th>IoU</th>
<th>runtime (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OctNet [2]</td>
<td>59.2</td>
<td>-</td>
</tr>
<tr>
<td>Autocontext_{3D} [3]</td>
<td>54.4</td>
<td>16</td>
</tr>
<tr>
<td>SPLATNet_{3D}</td>
<td>65.4</td>
<td>0.06</td>
</tr>
</tbody>
</table>

### with both 2D & 3D data

<table>
<thead>
<tr>
<th>Method</th>
<th>IoU</th>
<th>runtime (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autocontext_{2D-3D} [3]</td>
<td>62.9</td>
<td>87</td>
</tr>
<tr>
<td>SPLATNet_{2D-3D}</td>
<td>69.8</td>
<td>1.2</td>
</tr>
</tbody>
</table>


[1] Riemenschneider et al. ECCV ’14
2D predictions

<table>
<thead>
<tr>
<th>Method</th>
<th>IoU</th>
<th>runtime (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autocontext$_{2D}$ [1]</td>
<td>60.5</td>
<td>117</td>
</tr>
<tr>
<td>Autocontext$_{2D-3D}$ [1]</td>
<td>62.7</td>
<td>146</td>
</tr>
<tr>
<td>2D CNN [2] only</td>
<td>69.3</td>
<td>0.84</td>
</tr>
<tr>
<td>SPLATNet$_{2D-3D}$</td>
<td>70.6</td>
<td>4.34</td>
</tr>
</tbody>
</table>

[1] Gadde et al. PAMI ’17
[2] Chen et al. ICLR ’15
### 3D object part labeling (ShapeNet [1])

<table>
<thead>
<tr>
<th>Model</th>
<th>class avg. IoU</th>
<th>instance avg. IoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yi et al. [1]</td>
<td>79.0</td>
<td>81.4</td>
</tr>
<tr>
<td>3DCNN [2]</td>
<td>74.9</td>
<td>79.4</td>
</tr>
<tr>
<td>Kd-network [3]</td>
<td>77.4</td>
<td>82.3</td>
</tr>
<tr>
<td>PointNet [2]</td>
<td>80.4</td>
<td>83.7</td>
</tr>
<tr>
<td>PointNet++ [4]</td>
<td>81.9</td>
<td>85.1</td>
</tr>
<tr>
<td>SyncSpecCNN [5]</td>
<td>82.0</td>
<td>84.7</td>
</tr>
<tr>
<td>SPLATNet\textsubscript{3D}</td>
<td>82.0</td>
<td>84.6</td>
</tr>
<tr>
<td>SPLATNet\textsubscript{2D-3D}</td>
<td>83.7</td>
<td>85.4</td>
</tr>
</tbody>
</table>

[1] Yi et al. SIGGRAPH Asia ’16
[2] Qi et al. CVPR ’17
[4] Qi et al. NIPS ’17
[5] Yi et al. CVPR ’17
Filtering in other lattice spaces

• Adding additional \((x, y, z, n_x, n_y, n_z)\) filtering \(\rightarrow +0.2\) IoU
**SPLATNet: Summary**

- efficient computation directly on point clouds
- flexible specifications of receptive fields
- seamless joint 2D-3D processing

Poster location: C12

https://github.com/nvlabs/SPLATNet
Bilateral Convolution Layer (BCL)

[Jampani et al. CVPR ’16] [Kiefel et al. ICLR ‘15 workshops]

Convolution on sparse unordered points

3 steps in BCL
Other Methods for Point Set Analysis

**SPLATNet: Sparse Lattice Networks for Point Cloud Processing**
A fast and end-to-end trainable neural network that directly works on point clouds and can also do joint 2D-3D processing.

Computer Vision and Pattern Recognition, CVPR’18 (oral, best paper honorable mention)

**PointCNN**

Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, Baoquan Chen
Code
NIPS 2018.

**SGPN: Similarity Group Proposal Network for 3D Point Cloud Instance Segmentation**

Weiyue Wang, Ronald Yu, Qiangui Huang and Ulrich Neumann
IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, Spotlight
pdf | bibtex | code
3D Deep Learning: What’s Next?

- Leveraging synthetic environments
  - Reliable ground truth, benchmarking algorithms
  - Training 3D deep networks in large scale
3D Deep Learning: What’s Next?

- Leveraging simulation environments
EmbodiedQA

Agents that can See, Talk, Act, and Reason
Q: What color is the car?
Q: What color is the car?
Q: What color is the car?
Turn Left
A: Orange!
Q: What color is the bathtub?
A: Light grey

Q: What color is the fishtank in the living room?
A: Light blue
3D Deep Learning: What’s Next?

- Dealing with objects in multiple views
- Understanding **why** off-the-shelf 3D CNNs cannot work
- Connecting 3D & 2D deep learning
- ...
Thanks!