Sparse 3D Convolutional Neural Networks for Large-Scale Shape Retrieval

Alexandr Notchenko, Ermek Kapushev, Evgeny Burnaev
{avnotchenko,kapushev,burnaevevgeny}@gmail.com
3D Shape representations

- Meshes
- Point clouds
- Implicit surfaces / potentials
- Voxels
- Set of 2D projections
3D Shape representations

- Meshes
- Point clouds
- Implicit surfaces / potentials
- Voxels
- Set of 2D projections

Irregular size, not clear how to use in NN
Regular size, good to go in CNN
3D Shape representations

- Meshes
- Point clouds
- Implicit surfaces / potentials
- Voxels
- Set of 2D projections

Irregular size, not clear how to use in NN

Regular size, good to go in CNN

Not really 3D, 2D CNNs are powerful enough already
Sparsity of voxel representation

Mean sparsity for all classes of ModelNet40 train dataset at voxel resolution 40 equal to 5.5%.

Figure: Examples of some objects voxelizations at different resolutions 30, 50, 70, 100 (from left to right), left-most objects are depicted using original meshes
SparseConvNet

Dr. Benjamin Graham
formerly: Associate Professor at Warwick University
now at Facebook AI Research, Paris Lab

Convolutional filter shapes for different lattices: (i) A $4 \times 4$ square grid with a $2 \times 2$ convolutional filter. (ii) A triangular grid with size 4, and a triangular filter with size 2. (iii) A $3 \times 3 \times 3$ cubic grid, and a $2 \times 2 \times 2$ filter. (iv) A tetrahedral grid with size 3, and a filter of size 2.

To show how sparsity operates in 3D, consider a trefoil knot has been drawn in the cubic lattice (left). Applying a $2 \times 2 \times 2$ convolution, the number of active/non-zero sites increases (middle). Applying a $2 \times 2 \times 2$ pooling operation reduces the scale, which tends to decrease the number of active sites (right).

http://www2.warwick.ac.uk/fac/sci/statistics/staff/academic-research/graham/bmvc.pdf
PySparseConvNet

- Python wrapper for SparseConvNet, with extended functionality.
- Fixed several Memory issues that prevented large scale learning.
- Made possible to use different loss functions.
- Made layer activations accessible to debugging.
- Interactivity for exploration of models — a way to perform operations step by step, to explore properties of models.
Shape Retrieval

Problem statement

Given a query object find several the most “similar” to the query objects from the given database.

The objects are considered to be similar if they belong to the same category of objects and have similar shapes.
Shape Retrieval

Precomputed feature vector of dataset. ($V_{\text{car}}, V_{\text{person}}, \ldots$)

Cosine distance

Retrieved items
The representation can be efficiently learned by minimizing triplet loss.

Triplet is a set \((a, p, n)\), where

- \(a\) is an anchor object
- \(p\) is a positive object - an object that is similar to anchor object
- \(n\) is a negative object - an object that is not similar to anchor object

\[
\lambda(\delta_+, \delta_-) = \max(\mu + \delta_+ - \delta_-)
\]

where \(\mu\) is a margin parameter, \(\delta_+\) and \(\delta_-\) are distances between \(p\) and \(a\) and \(n\) and \(a\)
Our approach

- Use very large resolutions, and sparse representations.
- Used triplet learning for 3D shapes.
- Used Large Scale Shape Datasets ModelNet.
## Network description

<table>
<thead>
<tr>
<th>layer #</th>
<th>layer type</th>
<th>size</th>
<th>stride</th>
<th>channels</th>
<th>spatial size</th>
<th>mean sparsity (%)$^1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Data input</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>126</td>
<td>0.18</td>
</tr>
<tr>
<td>1</td>
<td>Sparse Convolutional Layer</td>
<td>2</td>
<td>1</td>
<td>8</td>
<td>125</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>Leaky ReLU ($\alpha = 0.33$)</td>
<td>-</td>
<td>-</td>
<td>32</td>
<td>125</td>
<td>0.35</td>
</tr>
<tr>
<td>3</td>
<td>Sparse MaxPooling Layer</td>
<td>3</td>
<td>2</td>
<td>32</td>
<td>62</td>
<td>0.69</td>
</tr>
<tr>
<td>4</td>
<td>Sparse Convolutional Layer</td>
<td>2</td>
<td>1</td>
<td>256</td>
<td>61</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>Leaky ReLU ($\alpha = 0.33$)</td>
<td>-</td>
<td>-</td>
<td>64</td>
<td>61</td>
<td>1.07</td>
</tr>
<tr>
<td>6</td>
<td>Sparse MaxPooling Layer</td>
<td>3</td>
<td>2</td>
<td>64</td>
<td>30</td>
<td>1.93</td>
</tr>
<tr>
<td>7</td>
<td>Sparse Convolutional Layer</td>
<td>2</td>
<td>1</td>
<td>512</td>
<td>29</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td>Leaky ReLU ($\alpha = 0.33$)</td>
<td>-</td>
<td>-</td>
<td>96</td>
<td>29</td>
<td>3.26</td>
</tr>
<tr>
<td>9</td>
<td>Sparse MaxPooling Layer</td>
<td>3</td>
<td>2</td>
<td>96</td>
<td>14</td>
<td>7.32</td>
</tr>
<tr>
<td>10</td>
<td>Sparse Convolutional Layer</td>
<td>2</td>
<td>1</td>
<td>768</td>
<td>13</td>
<td>-</td>
</tr>
<tr>
<td>11</td>
<td>Leaky ReLU ($\alpha = 0.33$)</td>
<td>-</td>
<td>-</td>
<td>128</td>
<td>13</td>
<td>15.14</td>
</tr>
<tr>
<td>12</td>
<td>Sparse MaxPooling Layer</td>
<td>3</td>
<td>2</td>
<td>128</td>
<td>6</td>
<td>46.30</td>
</tr>
<tr>
<td>13</td>
<td>Sparse Convolutional Layer</td>
<td>2</td>
<td>1</td>
<td>1024</td>
<td>5</td>
<td>-</td>
</tr>
<tr>
<td>14</td>
<td>Leaky ReLU ($\alpha = 0.33$)</td>
<td>-</td>
<td>-</td>
<td>160</td>
<td>5</td>
<td>97.54</td>
</tr>
<tr>
<td>15</td>
<td>Sparse MaxPooling Layer</td>
<td>3</td>
<td>2</td>
<td>160</td>
<td>2</td>
<td>100.00</td>
</tr>
<tr>
<td>16</td>
<td>Sparse Convolutional Layer</td>
<td>2</td>
<td>1</td>
<td>1280</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>17</td>
<td>Leaky ReLU ($\alpha = 0.33$)</td>
<td>-</td>
<td>-</td>
<td>192</td>
<td>1</td>
<td>100.00</td>
</tr>
</tbody>
</table>

$^1$Last column “sparsity” is computed for render size = 40
Forward Pass Activations

Input

Layer 2
Layer 3
Layer 11
Layer 15
Layer 17
Training Dynamics

Can finish learning when all samples outside of margin.

Optimisation algorithm:
Nesterov Accelerated Gradient with momentum = 0.99
Constant Learning Rate = 0.002
## Experimental results

<table>
<thead>
<tr>
<th>method</th>
<th>Classification</th>
<th>Retrieval AUC</th>
<th>Retrieval mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>3DShapeNet</td>
<td>77.32%</td>
<td>49.94%</td>
<td>49.23%</td>
</tr>
<tr>
<td>MVCNN</td>
<td>90.10%</td>
<td>---</td>
<td>80.20%</td>
</tr>
<tr>
<td>3DSCNN</td>
<td>90.3%</td>
<td>47.30%</td>
<td>45.16%</td>
</tr>
<tr>
<td>S3DCNN + triplet</td>
<td>---</td>
<td>48.81%</td>
<td>46.71%</td>
</tr>
</tbody>
</table>

### Precision-Recall Curve

- **S3DCNN+triplet**
- **MVCNN**
- **MVCNN+metric**
- **3D ShapeNet**

### mAP vs. Render Size

- **mAP** values for different render sizes.
## State-of-the-art

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>ModelNet40 Classification</th>
<th>ModelNet40 Retrieval (mAP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geometry Image [13]</td>
<td>83.9%</td>
<td>51.3%</td>
</tr>
<tr>
<td>Set-convolution [11]</td>
<td>90%</td>
<td></td>
</tr>
<tr>
<td>3D-GAN [10]</td>
<td>83.3%</td>
<td></td>
</tr>
<tr>
<td>VRN Ensemble [9]</td>
<td>95.54%</td>
<td></td>
</tr>
<tr>
<td>FusionNet [7]</td>
<td>90.8%</td>
<td></td>
</tr>
<tr>
<td>Pairwise [6]</td>
<td>90.7%</td>
<td></td>
</tr>
<tr>
<td>MVCNN [3]</td>
<td>90.1%</td>
<td>79.5%</td>
</tr>
<tr>
<td>GIFT [5]</td>
<td>83.10%</td>
<td>81.94%</td>
</tr>
<tr>
<td>VoxNet [2]</td>
<td>83%</td>
<td>77%</td>
</tr>
<tr>
<td>DeepPano [4]</td>
<td>77.63%</td>
<td>76.81%</td>
</tr>
<tr>
<td>3DShapeNets [1]</td>
<td>77%</td>
<td>49.2%</td>
</tr>
</tbody>
</table>

[11] Siyamak Ravanbakhsh, Jeff Schneider, Barnabas Poczos. Deep Learning with sets and point clouds
Conclusions

- For Modelnet in voxel form - resolution beyond $30^3$ doesn’t improves much
- More voxels - change scale of features, probably needs more layers
- Quality of representation depends on RS non smoothly but is maxed around render size of 55
Thank you.

Alexandr Notchenko, Ermek Kapushev, Evgeny Burnaev