Convolutional-Recursive Deep Learning for 3D Object Classification

Richard Socher, Brody Huval, Bharath Bhat, Christopher D. Manning, Andrew Y. Ng
NIPS 2012

Iro Armeni, Manik Dhar
Motivation

- Hand-designed features
- Useful extra information in the depth modality
- Affordable sensing for RGB-D (Kinect)
- Many applications (robotics)
Previous Work: Designed Features

- SURF, SIFT
- Kernel Descriptors
- Pyramid Matching Kernel

Grauman et al., *The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features*, ICCV’05
Bo et al., *Depth kernel descriptors for object recognition*, IROS’11
Bay et al., *Speeded-Up Robust Features (SURF)*, Computer Vision and Image Understanding, 110(3), ’08
Previous Work: Learned Features

Features of the second layer
Sparse coding + Spatial pooling + Normalization
Features of the first layer
Spatial pooling + Normalization

Raw pixel values

[Left] Bo et al., Unsupervised Feature Learning for RGB-D Based Object Recognition, ISER’12
[Right] Blum et al., A Learned Feature Descriptor for Object Recognition in RGB-D Data, ICRA’12
Limitations of previous work

- Difficult to extend to new modalities
- Capture limited features
- May not use the full image
- Need additional input
Convolutional - Recursive Neural Networks

1. Unsupervised filter learning

2. Single-layer CNN

3. Multi-random RNNs

4. Softmax Classification
Contributions

- New RNN-based architecture to combine low level CNN features (*fixed tree structures, multiple RNNs*)
- RNNs with random weights produce high quality features
- No additional input channels (*surface normals*)
- Fast approach
- State of the art results on detecting household objects.
Convolutional - Recursive Neural Networks
Model architecture details
Learning the CNN filters

Contrast Normalization + ZCA whitening

K-Means
CNN filters
A single CNN layer

Recursive Neural Networks (RNN)

Output

Layer 1

Input

- Structured Prediction: *learn compositional features and part interactions*
- Hierarchical Representation

\[ p = f \left( W \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right) \]

non-linearity e.g. \( \text{tanh} \)
Recursive Neural Networks (RNN)

Example of RNN architecture

```
p = f \left( W \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} \right)
p_i = f \left( W \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \right)
W \in \mathbb{R}^{K \times nK}
x_i \in \mathbb{R}^K
X \in \mathbb{R}^{K \times r \times r}
```

2nd layer
\((r/b)^2\) parents
\((r/b)^2 = 1\)

1st layer
\((r/b)^2\) parents
\((r/b)^2 = 4\)

output of CNN
\(K \times r \times r\) 3D matrix
\(r=4, b=2\)
Recursive Neural Networks (RNN)

- Learn hierarchical feature representations by applying the same NN recursively in a tree structure
- \textit{Object classification} + CNN allow to use fixed tree structure
- Generalized RNN architecture to merge blocks of adjacent vectors, not only pairs
Back-prop vs Multi-Random RNNs

Back propagation is costly (very slow).

Alternatively use multiple RNNs, with randomly initialized weights $W$.

Concatenate all RNN outputs to a $N \times K$-dimensional vector (input to softmax).
Experiments

Object Category Recognition
Large-scale hierarchical multi-view RGB-D object dataset*

- 51 object categories (*house/office*)
- Total of 300 instances
- Multi-view RGB-D images
- 3 angles, ~600 images per instance
- Total of 207,920 images

* Lai et al., A Large-Scale Hierarchical Multi-View RGB-D Object Dataset, ICRA’11
Experimental Setup: General

- Subsample images per instance to $\frac{1}{6}$ (120 images)
- Same setup as in Lai et al.* - 10 random splits
- Test: 1 instance per category (120 images)
- Training: remaining images (~34,000)
- Resize: 148x148

*Lai et al., A Large-Scale Hierarchical Multi-View RGB-D Object Dataset, ICRA’11*
Experimental Setup: Filters

- K-means on 500,000 image patches (random sampling from each split’s training set)
- # of clusters/filters: 128
- Size of patches: 9x9x3 (RGB), 9x9 (Depth)
- Pre-processing: Individual normalization (-mean, /std) & ZCA whitening
Experimental Setup: CNN layer

- Filter bank size: 128
- Filter WxH: 9x9
- Average Pooling: size 10, stride 5
- Output: 128x27x27
Experimental Setup: Multi RNNs

- Block size (children): 3x3
- (Single) Tree layers: 128x27x27, 128x9x9, 128x3x3, 128x1x1
- # of RNNs: 128

Final concatenated output (RGB-D): \(2 \times 128^2 = 32768\)
## Results: Comparison to other methods

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Extra Features for 3D;RGB</th>
<th>3D</th>
<th>RGB</th>
<th>Both</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear SVM [2]</td>
<td>Spin Images, efficient match kernel (EMK), random Fourier sets, width, depth, height; SIFT, EMK, texton histogram, color histogram</td>
<td>53.1±1.7</td>
<td>74.3±3.3</td>
<td>81.9±2.8</td>
</tr>
<tr>
<td>Kernel SVM [2]</td>
<td>same as above</td>
<td>64.7±2.2</td>
<td>74.5±3.1</td>
<td>83.9±3.5</td>
</tr>
<tr>
<td>Random Forest [2]</td>
<td>same as above</td>
<td>66.8±2.5</td>
<td>74.7±3.6</td>
<td>79.6±4.0</td>
</tr>
<tr>
<td>SVM [5]</td>
<td>3D shape, physical size of the object, depth edges, gradients, kernel PCA, local binary patterns, multiple depth kernels</td>
<td>78.8±2.7</td>
<td>77.7±1.9</td>
<td>86.2±2.1</td>
</tr>
<tr>
<td>CKM [6]</td>
<td>SURF interest points</td>
<td>–</td>
<td>–</td>
<td>86.4±2.3</td>
</tr>
<tr>
<td>SP+HMP [7]</td>
<td>surfacing normals</td>
<td>81.2±2.3</td>
<td>82.4±3.1</td>
<td>87.5±2.9</td>
</tr>
<tr>
<td>CNN-RNN</td>
<td>–</td>
<td>78.9±3.8</td>
<td>80.8±4.2</td>
<td>86.8±3.3</td>
</tr>
</tbody>
</table>

[2] Lai et al., A Large-Scale Hierarchical Multi-View RGB-D Object Dataset, ICRA’11

[5] Bo et al., Depth Kernel descriptors for object recognition, IROS’11

[6] Blum et al., A Learned Feature Descriptor for Object Recognition in RGB-D Data, ICRA’12

[7] Bo et al., Unsupervised Feature Learning for RGB-D Based Object Recognition, ISER’12
Model Analysis: **CNN-RNN** vs **2-layer CNN**

<table>
<thead>
<tr>
<th>Filters</th>
<th>2nd Layer</th>
<th>Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>See [17]</td>
<td>CNN</td>
<td>77.66</td>
</tr>
<tr>
<td>See [17]</td>
<td>RNN</td>
<td>77.04</td>
</tr>
<tr>
<td>$k$-means</td>
<td>tRNN</td>
<td>78.10</td>
</tr>
<tr>
<td>$k$-means</td>
<td>TNN</td>
<td>79.67</td>
</tr>
<tr>
<td>$k$-means</td>
<td>CNN</td>
<td>78.65</td>
</tr>
<tr>
<td>$k$-means</td>
<td>RNN*</td>
<td>80.15</td>
</tr>
</tbody>
</table>

RNN* outperforms and is 4x faster

---

[17] Jarrett et al., *What is the Best Multi-Stage Architecture for Object Recognition?*, ICCV'09
TNN: Tree structured NN with untied weights (same weights within a layer, different among layers)

It has more parameters but worst performance.

Tying the weights is beneficial.

<table>
<thead>
<tr>
<th>Filters</th>
<th>2nd Layer</th>
<th>Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>See [17]</td>
<td>CNN</td>
<td>77.66</td>
</tr>
<tr>
<td>See [17]</td>
<td>RNN</td>
<td>77.04</td>
</tr>
<tr>
<td>$k$-means</td>
<td>tRNN</td>
<td>78.10</td>
</tr>
<tr>
<td>$k$-means</td>
<td>TNN</td>
<td>79.67</td>
</tr>
<tr>
<td>$k$-means</td>
<td>CNN</td>
<td>78.65</td>
</tr>
<tr>
<td>$k$-means</td>
<td>RNN*</td>
<td>80.15</td>
</tr>
</tbody>
</table>

*only RGB
**Model Analysis:** Multi-random RNNs vs single tRNN

*tRNN: Trained RNN with careful parameter tuning*

128 random RNNs perform 2% better and faster.

<table>
<thead>
<tr>
<th>Filters</th>
<th>2nd Layer</th>
<th>Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>See [17]</td>
<td>CNN</td>
<td>77.66</td>
</tr>
<tr>
<td>See [17]</td>
<td>RNN</td>
<td>77.04</td>
</tr>
<tr>
<td><em>k</em>-means</td>
<td>tRNN</td>
<td>78.10</td>
</tr>
<tr>
<td><em>k</em>-means</td>
<td>TNN</td>
<td>79.67</td>
</tr>
<tr>
<td><em>k</em>-means</td>
<td>CNN</td>
<td>78.65</td>
</tr>
<tr>
<td><em>k</em>-means</td>
<td>RNN*</td>
<td>80.15</td>
</tr>
</tbody>
</table>

*only RGB
Model Analysis: how many random RNNs?

![Graph showing accuracy vs. number of RNNs]
Model Analysis: RGB, Depth or RGB+D?

Classifier benefits from RGB and Depth combination
Error Analysis

Confusion Matrix

GT: y-axis
Predictions: x-axis
Error Analysis

shampoo bottle  mushroom  cap  white cap

water bottle  garlic  pitcher  kleenex box
Conclusion

- New model combining CNN + RNN
- Learning a lower-dimensional feature vector with high representational power
- RNNs: hierarchical representation of the structure (spatial interactions)
- No back-propagation
- Applicability on Depth image domain
- Combination of modalities into one feature vector (complementing features)
- Use only of raw data
- Parallelization & High speed
Advances in RGB-D object recognition
Convolutional Fisher Kernels for RGB-D Object Recognition

Cheng et al., Convolutional Fisher Kernels for RGB-D Object Recognition, 3DV’15
Convolutional Fisher Kernels for RGB-D Object Recognition

- 85.8%
- 86.8%
- 89.23%
- 91.15%

~5% increase

Classification Accuracy (%)

- depth
- RGB
- combine

- RGB
- Depth
- RGB+Depth

80.8%
86.8%

Multimodal Deep Learning for Robust RGB-D Object Recognition
## Multimodal Deep Learning for Robust RGB-D Object Recognition

<table>
<thead>
<tr>
<th>Method</th>
<th>RGB</th>
<th>Depth</th>
<th>RGB-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nonlinear SVM [15]</td>
<td>74.5 ± 3.1</td>
<td>64.7 ± 2.2</td>
<td>83.9 ± 3.5</td>
</tr>
<tr>
<td>HKDES [4]</td>
<td>76.1 ± 2.2</td>
<td>75.7 ± 2.6</td>
<td>84.1 ± 2.2</td>
</tr>
<tr>
<td>Kernel Desc. [14]</td>
<td>77.7 ± 1.9</td>
<td>78.8 ± 2.7</td>
<td>86.2 ± 2.1</td>
</tr>
<tr>
<td>CKM Desc. [3]</td>
<td>N/A</td>
<td>N/A</td>
<td>86.4 ± 2.3</td>
</tr>
<tr>
<td><strong>CNN-RNN [22]</strong></td>
<td><strong>78.9 ± 4.2</strong></td>
<td><strong>80.8 ± 3.8</strong></td>
<td><strong>86.8 ± 3.3</strong></td>
</tr>
<tr>
<td>Upgraded HMP [5]</td>
<td>82.4 ± 3.1</td>
<td>81.2 ± 2.3</td>
<td>87.5 ± 2.9</td>
</tr>
<tr>
<td>CaRFs [1]</td>
<td>N/A</td>
<td>N/A</td>
<td>88.1 ± 2.4</td>
</tr>
<tr>
<td>CNN Features [20]</td>
<td>83.1 ± 2.0</td>
<td>N/A</td>
<td>89.4 ± 1.3</td>
</tr>
<tr>
<td><strong>Ours, Fus-CNN (HHA)</strong></td>
<td><strong>84.1 ± 2.7</strong></td>
<td><strong>83.0 ± 2.7</strong></td>
<td><strong>91.0 ± 1.9</strong></td>
</tr>
<tr>
<td><strong>Ours, Fus-CNN (jet)</strong></td>
<td><strong>84.1 ± 2.7</strong></td>
<td><strong>83.8 ± 2.7</strong></td>
<td><strong>91.3 ± 1.4</strong></td>
</tr>
</tbody>
</table>

~5% increase
Link to presentation:

https://docs.google.com/presentation/d/1HDInPn9WYhYG1j4sOhYEcpwmL7-0edfd4leIP1wLs7s/edit#slide=id.g183cef1ae4_0_42