Hypercolumns for Object Segmentation and Fine-grained Localization

Bharath Hariharan, Pablo Arbelaez, Ross Girshick, Jitendra Malik

Göksu Erdoğan
Image Classification

horse, person, building
Object Detection

Slide credit: Bharath Hariharan
Simultaneous Detection and Segmentation

Detect and *segment* every *instance* of the category in the image


Slide credit: Bharath Hariharan
SDS ≠ Semantic Segmentation

Slide credit: Bharath Hariharan
Simultaneous Detection and Part Labeling

Detect and **segment** every **instance** of the category in the image and **label its parts**

Slide credit: Bharath Hariharan
Simultaneous Detection and Keypoint Prediction

Detect every instance of the category in the image and mark its keypoints.
Motivation

- Task: Assign category labels to images or bounding boxes
- General Approach: Output of last layer of CNN
  - This is most sensitive to category-level semantic information
  - The information is generalized over in the top layer

- Is output of last layer of CNN appropriate for finer-grained problems?
Motivation

- Not optimal representation!
- Last layer of CNN is mostly \textit{invariant} to 'nuisance' variables such as pose, illumination, articulation, precise location ...
- Pose and nuisance variables are precisely what we interested in.

- How can we get such an information?
Motivation

- It is present in intermediate layers
- Less sensitive to semantics
Motivation

- Top layers lose localization information
- Bottom layers are not semantic enough

- Combine both
Detection and Segmentation

Simultaneous detection and segmentation

- Proposal Generation
- Feature Extraction
  - Box CNN
  - Region CNN
- Region Classification
  - Person? +1.8
- Region Refinement

Combining features across multiple levels:

Combine *subsampled* intermediate layers with top layer

Difference

Upsampling

Figure 3: A multi-scale convolutional network. The top row of maps constitute a regular ConvNet [17]. The bottom row in which the 1st stage output is branched, subsampled again and merged into the classifier input provides a multi-stage component to the classifier stage. The multi-stage features coming out of the 2nd stage extracts a global structure as well as local details.

Pedestrian Detection with Unsupervised Multi-Stage Feature Learning
Sermanet et. al.
Framework

- Start from a detection (R-CNN)
- Heatmaps
- Use category-specific, instance-specific information to...
- Classify each pixel in detection window

Slide credit: Bharath Hariharan
One Framework, Many Tasks:

<table>
<thead>
<tr>
<th>Task</th>
<th>Classification Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDS</td>
<td>Does the pixel belong to the object?</td>
</tr>
<tr>
<td>Part labeling</td>
<td>Which part does the pixel belong to?</td>
</tr>
<tr>
<td>Pose estimation</td>
<td>Does it lie on/near a particular keypoint</td>
</tr>
</tbody>
</table>

Slide credit: Bharath Hariharan
Heatmaps for each task

- **Segmentation:**
  - Probability that a particular location inside the object

- **Part Labeling:**
  - Separate heatmap for each part
  - Each heatmap is the probability a location belongs to that part

- **Keypoint Prediction**
  - Separate heatmap for each keypoint
  - Each heatmap is the probability of the keypoint at a particular location
Hypercolumns

Slide credit: Bharath Hariharan
Hypercolumns

- Term derived from Hubel and Wiesel
- Re-imagines old ideas:
  - Jets (Koenderink and van Doorn)
  - Pyramids (Burt and Adelson)
  - Filter Banks (Malik and Perona)

Figure 1. The hypercolumn representation. The bottom image is the input, and above it are the feature maps of different layers in the CNN. The hypercolumn at a pixel is the vector of activations of all units that lie above that pixel.
Computing the Hypercolumn Representation

- Upsampling feature map $F$ to $f$
- feature vector for at location $i$

$\alpha_{ik}$: position of $i$ and $k$ in the box
- Concatenate features from every location to one long vector
Interpolating into grid of classifiers

- Fully connected layers contribute to global instance-specific bias
- Different classifier for each location contribute to separate instance-specific bias
- Simplest way to get location specific classifier:
  - train separate classifiers at each 50x50 locations

- What would be the problems of this approach?
Interpolating into grid of classifiers

1. Reduce amount of data for each classifier during training
2. Computationally expensive
3. Classifier vary with locations
4. Risk of overfitting

How can we escape from these problems?
Interpolate into coarse grid of classifiers

- Train a coarse KxK grid of classifiers and interpolate between them
- Interpolate grid of functions instead of values
- Each classifier in the grid is a function \( g_k(.) \)
- \( g_k(\text{feature vector}) = \text{probability} \)
- Score of \( i \)'th pixel

\[
p_i = \sum_k \alpha_{ik} g_k(f_i) = \sum_k \alpha_{ik} p_{ik}
\]
Training classifiers

- Interpolation is not used in train time
- Divide each box to KxK grid
- Training data for k’th classifier only consists of pixels from the k’th grid cell across all training instances.
- Train with logistic regression
Hypercolumns
Efficient pixel classification

- Upsampling large feature maps is expensive!
- If classification and upsampling are linear
  - Classification $\circ$ upsampling = Upsampling $\circ$ classification
- Linear classification = $1 \times 1$ convolution
  - Extension: use $n \times n$ convolution
- Classification = convolve, upsample, sum, sigmoid
Efficient pixel classification
Efficient pixel classification

Slide credit: Bharath Hariharan
Efficient pixel classification

Slide credit: Bharath Hariharan
Representation as a neural network

Figure 2. Representing our hypercolumn classifiers as a neural network. Layers of the original classification CNN are shown in red, and layers that we add are in blue.
Training classifiers

- MCG candidates overlaps with ground truth by \( \%70 \) or more
- For each candidate find most overlapped ground truth instance
- Crop ground truth to the expanded bounding box of the candidate
- Label locations positive or negative according to problem
Experiments
Evaluation Metric

- Similar to bounding box detection metric

- Box overlap = \( \frac{\cap}{\cup} \)

- If box overlap > threshold, **correct**
Evaluation Metric

- Similar to bounding box detection metric
- But with segments instead of bounding boxes
- Each detection/GT comes with a segment

\[
\text{segment overlap} = \frac{\cap}{\cup}
\]

- If segment overlap > threshold, correct

Slide credit: Bharath Hariharan
Task 1: SDS

- **System 1:**
  - Refinement step with hypercolumns representation
  - Features
    - Top-level fc7 features
    - Conv4 features
    - Pool2 features
    - 1/0 according to location was inside original region candidate or not
    - Coarse 10x10 discretization of original candidate into 100-dimensional vector
  - 10x10 grid of classifiers
  - Project predictions over superpixels and average
Task 1: SDS

System 1

<table>
<thead>
<tr>
<th>Metric</th>
<th>[22] refined</th>
<th>[22]</th>
<th>Hyp +bbox-reg</th>
<th>Hyp +bbox-reg</th>
<th>Hyp+FT +bbox-reg</th>
<th>Only fc7</th>
<th>Only fc7+ pool2+ conv4</th>
<th>Only fc7+ pool2+ conv4</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean AP$\tau$ at 0.5</td>
<td>47.7</td>
<td>49.7</td>
<td>51.2</td>
<td>51.9</td>
<td><strong>52.8</strong></td>
<td>49.7</td>
<td>50.5</td>
<td>51.0</td>
</tr>
<tr>
<td>mean AP$\tau$ at 0.7</td>
<td>22.8</td>
<td>25.3</td>
<td>31.6</td>
<td>32.4</td>
<td><strong>33.7</strong></td>
<td>25.8</td>
<td>30.6</td>
<td>31.2</td>
</tr>
</tbody>
</table>

Table 1. Results on SDS on VOC2012 val using System 1. Our system (Hyp+FT+bbox-reg) is significantly better than [22] (Section 4.1).
Task 1: SDS

- System 2:
  - MCG instead of selective search
  - Expand set of boxes by adding nearby high-scoring boxes after NMS

Figure 3. An alternative pipeline for SDS starting from bounding box detections (Section 4)
**Task 1: SDS**

<table>
<thead>
<tr>
<th>Metric</th>
<th>T-Net Only $fc7$</th>
<th>T-Net Hyp $fc7$</th>
<th>O-Net Only $fc7$</th>
<th>O-Net Hyp</th>
<th>O-Net Hyp+ Rescore</th>
</tr>
</thead>
<tbody>
<tr>
<td>mAP$^r$ at 0.5</td>
<td>44.0</td>
<td>49.1</td>
<td>52.6</td>
<td>56.5</td>
<td><strong>60.0</strong></td>
</tr>
<tr>
<td>mAP$^r$ at 0.7</td>
<td>16.3</td>
<td>29.1</td>
<td>22.4</td>
<td>37.0</td>
<td><strong>40.4</strong></td>
</tr>
</tbody>
</table>

Table 2. Results on SDS on VOC 2012 val using System 2. Our final pipeline is state-of-the-art on SDS. (Section 4.2)
Hypercolumns vs Top Layer

Figure 4. Figure ground segmentations starting from bounding box detections. Top row: baseline using fc7, bottom row: Ours.
Hypercolumns vs Top Layer

Slide credit: Bharath Hariharan
Task 2: Part Labeling
Task 2: Part Labeling

<table>
<thead>
<tr>
<th>$AP_{part}^r$ at 0.5</th>
<th>Person</th>
<th>Horse</th>
<th>Cow</th>
<th>Sheep</th>
<th>Cat</th>
<th>Dog</th>
<th>Bird</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only fc7</td>
<td>21.9</td>
<td>16.6</td>
<td>14.5</td>
<td>38.9</td>
<td>19.2</td>
<td>8.5</td>
<td>15.4</td>
</tr>
<tr>
<td>Hyp</td>
<td>28.5</td>
<td>27.8</td>
<td>21.5</td>
<td>44.9</td>
<td>30.3</td>
<td>14.2</td>
<td>14.2</td>
</tr>
</tbody>
</table>

Table 4. Results on part labeling. Our approach (Hyp) is almost uniformly better than using top level features (Section 5).
Task 2: Part Labeling

Figure 6. Part labeling. Top: baseline using fc7, bottom: ours (hypercolumns). Both rows use the same figure-ground segmentation. Red: head, green: torso, blue: legs, magenta: arms.
Task 3: Keypoint Prediction
Task 3: Keypoint Prediction

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[21]</td>
<td>27.3</td>
<td>11.8</td>
<td>2.8</td>
<td>27.1</td>
<td>12.2</td>
<td>3.4</td>
<td>11.4</td>
<td>3.2</td>
<td>10.6</td>
<td>4.4</td>
<td>3.8</td>
<td>42.9</td>
<td>12.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[20]</td>
<td>32.1</td>
<td>14.6</td>
<td>5.6</td>
<td>32.5</td>
<td>16.6</td>
<td>5.9</td>
<td>10.8</td>
<td>4.8</td>
<td>4.8</td>
<td>9.7</td>
<td>4.0</td>
<td>4.6</td>
<td>52.0</td>
<td>15.2</td>
<td></td>
</tr>
<tr>
<td>Only fc7</td>
<td>22.7</td>
<td>9.9</td>
<td>2.8</td>
<td>25.5</td>
<td>10.0</td>
<td>2.6</td>
<td>6.6</td>
<td>5.2</td>
<td>7.7</td>
<td>3.4</td>
<td>4.2</td>
<td>34.0</td>
<td>10.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hyp</td>
<td>32.2</td>
<td>16.5</td>
<td>11.5</td>
<td>31.2</td>
<td>16.6</td>
<td>9.3</td>
<td>9.6</td>
<td>7.1</td>
<td>9.1</td>
<td>8.0</td>
<td>4.2</td>
<td>8.2</td>
<td>57.5</td>
<td>17.0</td>
<td></td>
</tr>
<tr>
<td>Hyp+FT</td>
<td>33.7</td>
<td>21.9</td>
<td>12.3</td>
<td>35.2</td>
<td>20.9</td>
<td>15.3</td>
<td>6.7</td>
<td>7.7</td>
<td>8.1</td>
<td>9.1</td>
<td>5.6</td>
<td>6.1</td>
<td>58.4</td>
<td>18.5</td>
<td></td>
</tr>
</tbody>
</table>

Table 3. Results on keypoint prediction (APK on the Person subset of VOC2009 val). Our system is 3.3 points better than [20] (Section 5).
Task 3: Keypoint Prediction

Figure 5. Keypoint prediction (left wrist). Top row: baseline using fc7, bottom row: ours (hypercolumns without finetuning). In black is the bounding box and the predicted heatmap is in red. We normalize each heatmap so that the maximum value is 1.
Conclusion

- A general framework for fine-grained localization that:
  - Leverages information from multiple CNN layers
  - Achieves state-of-the-art performance on SDS and part labeling and accurate results on keypoint prediction
Future Work

- applying hypercolumn representation to fine-grained tasks
  - Attribute classification
  - Action classification
  - ...

Questions???
THANK YOU😊