R-CNN for Object Detection

Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik
(UC Berkeley)

presented by
Ezgi Mercan
Outline

1. Problem Statement: Object Detection (and Segmentation)
2. Background: DPM, Selective Search, Regionlets
3. Method overview
4. Evaluation
5. Extensions to DPM and RGB-D
6. Discussion
Detection and Segmentation

input image

object detection

segmentation
Background: VOC

- PASCAL Visual Object Classes Challenge
- 20 classes, ~10K images, ~25K annotated objects
- Training, validation, test data sets.
- Evaluation:
  - Average Precision (AP) per class
  - mean Average Precision
Background: Deformable Parts Model

• Strong low-level features based on histograms of oriented gradients (HOG)
• Efficient matching algorithms for deformable part-based models (pictorial structures)
• Discriminative learning with latent variables (latent SVM)
• mean Average Precision (mAP): 33.7% - 33.4%
• mAP with “context”: 35.4%
• mAP with “sketch tokens”: 29.1%
• mAP with “histograms of sparse codes”: 34.3%

Background: Selective search

• Alternative to exhaustive search with sliding window.

• Starting with over-segmentation, merge *similar* regions and produce region proposals.

• Bag-of-Words Model with Dense SIFT, OpponentSIFT and RGB-SIFT, plus SVM.

• mAP: ? – 35.1%

Background: Regionlets

- Start with *selective search*.
- Define sub-parts of regions whose position/resolution are relative and normalized to a detection window, as the basic units to extract appearance features.
- Features: HOG, LBP, Covariance.
- mAP: 41.7% - 39.7%

Deep Learning is back!

UToronto “SuperVision” CNN

ImageNet 2012
whole-image classification with 1000 categories

<table>
<thead>
<tr>
<th>Model</th>
<th>Top-1 (val)</th>
<th>Top-5 (val)</th>
<th>Top-5 (test)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIFT + Fisher Vectors</td>
<td>-</td>
<td>-</td>
<td>26.2%</td>
</tr>
<tr>
<td>1 CNN</td>
<td>40.7%</td>
<td>18.2%</td>
<td>-</td>
</tr>
<tr>
<td>5 CNNs</td>
<td>38.1%</td>
<td>16.4%</td>
<td>16.4%</td>
</tr>
<tr>
<td>1 CNN (pre-trained)</td>
<td>39.0%</td>
<td>16.6%</td>
<td>-</td>
</tr>
<tr>
<td>7 CNNs (pre-trained)</td>
<td>36.7%</td>
<td>15.4%</td>
<td>15.3%</td>
</tr>
</tbody>
</table>

• Can it be used in object recognition?
• Problems:
  • localization: Where is the object?
  • annotation: Labeled data is scarce.

R-CNN: Region proposals + CNN

<table>
<thead>
<tr>
<th></th>
<th>localization</th>
<th>feature extraction</th>
<th>classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>this paper:</td>
<td>selective search</td>
<td>deep learning CNN</td>
<td>binary linear SVM</td>
</tr>
</tbody>
</table>
| alternatives:       | objectness, constrained          | HOG, SIFT, LBP, BoW, DPM ...    | SVM, Neural networks, Logistic regression ...
|                     | constrained parametric min-cuts,  |                                |                             |
|                     | sliding window ...               |                                |                             |

10/3/2014
R-CNN: Training

1. Pre-train CNN for image classification

   large auxiliary dataset (ImageNet)

2. Fine-tune CNN for object detection

   small target dataset (PASCAL VOC)

3. Train linear predictor for object detection

   region proposals
   small target dataset (PASCAL VOC)
   ~2000 warped windows/image
   CNN features
   training labels

   per class SVM
UToronto “SuperVision” CNN

# Evaluation: mAP

<table>
<thead>
<tr>
<th>Method</th>
<th>VOC 2007</th>
<th>VOC 2010</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM v5 (Girshick et al. 2011)</td>
<td>33.7%</td>
<td>29.6%</td>
</tr>
<tr>
<td>UVA sel. search (Uijlings et al. 2012)</td>
<td>35.1%</td>
<td></td>
</tr>
<tr>
<td>Regionlets (Wang et al. 2013)</td>
<td>41.7%</td>
<td>39.7%</td>
</tr>
<tr>
<td><strong>pre-trained only</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R-CNN pool$_5$</td>
<td>44.2%</td>
<td></td>
</tr>
<tr>
<td>R-CNN fc$_6$</td>
<td>46.2%</td>
<td></td>
</tr>
<tr>
<td>R-CNN fc$_7$</td>
<td>44.7%</td>
<td></td>
</tr>
<tr>
<td><strong>fine-tuned</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R-CNN pool$_5$</td>
<td>47.3%</td>
<td></td>
</tr>
<tr>
<td>R-CNN fc$_6$</td>
<td>53.1%</td>
<td></td>
</tr>
<tr>
<td>R-CNN fc$_7$</td>
<td>54.2%</td>
<td>50.2%</td>
</tr>
<tr>
<td>R-CNN fc$_7$ (Bounding Box regression)</td>
<td>58.5%</td>
<td>53.7%</td>
</tr>
</tbody>
</table>
Evaluation: Top False Positives
Bicycle (AP 62.5%)
Evaluation: Top False Positives
Bird (AP 41.4%)
Evaluation: False positive types
Cat (AP 56.3%)

UToronto “SuperVision” CNN

pool5
6x6x256 = 9216 dimensional
pool5 feature: (3,3,42) (top 1 – 96)
pool5 feature: (3,4,80) (top 1 – 96)
pool5 feature: (4,5,110) (top 1 − 96)
pool5 feature: (3,5,129) (top 1 − 96)
pool5 feature: (4,2,26) (top 1 – 96)
pool5 feature: (3,3,39) (top 1 – 96)
pool5 feature: (5,6,53) (top 1 – 96)
pool5 feature: (3,3,139) (top 1 – 96)
pool5 feature: (1,4,138) (top 1 − 96)
pool5 feature: (2,3,210) (top 1 – 96)
Discussion

• Days of HOG, SIFT, LBP, and feature engineering are over?
• Machines can *design* better features than man?
Part-based R-CNNs for Fine-grained Category Detection

• Caltech-UCSD bird dataset (CUB200-2011) with ~12,000 images of 200 bird species.

• Strongly supervised setting in which ground truth bounding boxes of full objects (birds) and parts (head and body) are given.

• Each part + full object are treated as independent object categories to train SVMs in original R-CNN pipeline.

• Then geometric constraints (box + knn) are applied.

Part-based R-CNNs for Fine-grained Category Detection
Part-based R-CNNs for Fine-grained Category Detection
R-CNNs on RGB-D for Object Detection and Segmentation

Pre-trained on Image-Net using RGB images.
Fine-tuned on NYUD2 (400 images) and synthetic data.
SVM training on pool5, \textbf{fc6} and \textbf{fc7}.

R-CNNs on RGB-D
for Object Detection and Segmentation

<table>
<thead>
<tr>
<th>Model</th>
<th>DPM</th>
<th>DPM</th>
<th>CNN</th>
<th>CNN</th>
<th>CNN</th>
<th>CNN</th>
<th>CNN</th>
<th>CNN</th>
<th>CNN</th>
<th>CNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fine-tuned</td>
<td></td>
<td></td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Input channels</td>
<td>RGB</td>
<td>RGBD</td>
<td>RGB</td>
<td>RGB</td>
<td>disp</td>
<td>disp</td>
<td>HHA</td>
<td>HHA</td>
<td>HHA</td>
<td>HHA</td>
</tr>
<tr>
<td>synth data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2x</td>
<td>15x</td>
<td>2x</td>
<td>2x</td>
</tr>
<tr>
<td>CNN layer</td>
<td>fc6</td>
<td>fc6</td>
<td>fc6</td>
<td>fc6</td>
<td>fc6</td>
<td>fc6</td>
<td>fc6</td>
<td>pool5</td>
<td>fc7</td>
<td>fc6</td>
</tr>
<tr>
<td>mAP</td>
<td>8.4</td>
<td>21.7</td>
<td>16.4</td>
<td><strong>19.7</strong></td>
<td>11.3</td>
<td>20.1</td>
<td>25.2</td>
<td><strong>26.1</strong></td>
<td>25.6</td>
<td>21.9</td>
</tr>
</tbody>
</table>

**HHA:**
Horizontal disparity,
Height above ground,
Angle the pixel's local surface normal makes with the inferred gravity direction.
R-CNNs on RGB-D