Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation

ROSS GIRSHICK, JEFF DONAHUE, TREVOR DARRELL, JITENDRA MALIK

PRESENTED BY: COLLIN MCCARTHY
Goal

To produce a scalable, state-of-the-art detection algorithm
- CNN’s using bottom-up regions
- Pre-training (general), fine-tuning (specific)
Background
Neurons

Single Neuron

Neural Network
Artificial Neural Network

Sigmoid (logistic) activation function

- How much a neuron “fires”
Artificial Neural Network
Previous Work
PASCAL VOC Challenge Results
Post-Challenge Results (2013)

- Regionlets (2013)
- SegDPM (2013)
- DPM++
- Selective Search
- DPM++, MKL

Post-competition results (2013 - present)

- Top competition results (2007 - 2012)

PASCAL VOC challenge dataset
Regions with CNN Features (2014)

30% Relative Improvement!
Previous Work on CNNs

Fukushima 1980
Neocognitron

Rumelhart, Hinton, Williams 1986
“T” versus “C” problem

LeCun et al. 1989-1998
Handwritten digit reading / OCR

Krizhevsky, Sutskever, Hinton 2012
ImageNet classification breakthrough
“SuperVision” CNN
Recent Work on CNNs for Object Detection

- Vaillant, Monrocq, LeCun 1994
  - Multi-scale face detection

- LeCun, Huang, Bottou 2004
  - NORB dataset

- Cireşan et al. 2013
  - Mitosis detection

- Sermanet et al. 2013
  - Pedestrian detection

- Szegedy, Toshev, Erhan 2013
  - PASCAL detection (VOC’07 mAP 30.5%)
Methods
Regions with CNN Features

Input image

Extract region proposals (~2k / image)

Compute CNN features

Classify regions (linear SVM)

- aeroplane? no.
- person? yes.
- tvmonitor? no.
R-CNN: Step 1

Selective Search [van de Sande, Uijlings et al.]
Selective Search

Approximate segmentation at multiple scales
- Capture more background
- Less expensive than exhaustive

R-CNN: Step 1

Selective Search [van de Sande, Uijlings et al.]
R-CNN: Step 2
R-CNN: Step 2

Input image

Extract region proposals (~2k / image)

Compute CNN features

Dilate proposal
R-CNN: Step 2

Input image → Extract region proposals (~2k/image) → Compute CNN features

- aeroplane? no.
- person? yes.
- tvmonitor? no.

a. Crop
R-CNN: Step 2

- Extract region proposals (~2k/image)
- Compute CNN features
- a. Crop
- b. Scale (anisotropic)
R-CNN: Step 2

Input image → Extract region proposals (~2k/image) → Compute CNN features

1. Crop
2. Scale (anisotropic)
3. Forward propagate
Output: “fc7” features
R-CNN: Step 2

Forward Propagation

11x11 kernel size w/ sliding window
R-CNN: Step 2

Kernel Convolution

R-CNN: Step 2

Forward Propagation

At each stage
- Higher level features, from convolution alone!
- Max pooling keeps best features
- Convolution kernels learned from training
R-CNN: Step 3

Input image  
Extract region proposals (~2k / image)  
Compute CNN features  
Classify regions

person? 1.6  
horse? -0.3  

4096-dimensional fc7 feature vector  
linear classifiers (SVM or softmax)

acroplane? no.  
person? yes.  
tvmonitor? no.
R-CNN: Step 4

**Step 4: Object proposal refinement**

- **Original proposal**
- **Linear regression on CNN features**
- **Predicted object bounding box**
- **Bounding-box regression**
R-CNN: Step 4

Predicting Object Bounding Box

Ground Truth Bounding Box
R-CNN: Step 4

Predicting Object Bounding Box

Features:

$F_{\text{human-upper}}$

Ground Truth Bounding Box
R-CNN: Step 4

Predicting Object Bounding Box

Ground Truth Bounding Box

Features: $F_{\text{human-upper}}$

Transformation: $T_{\text{human-upper}}$
R-CNN: Step 4

Predicting Object Bounding Box

Features: $F_{\text{human-upper}}$

Ground Truth Bounding Box

Transformation: $T_{\text{human-upper}}$

Features: $F_{\text{human-lower}}$
R-CNN: Step 4

Predicting Object Bounding Box

Transformation: $T_{\text{human-upper}}$

Transformation: $T_{\text{human-lower}}$

Features: $F_{\text{human-upper}}$

Features: $F_{\text{human-lower}}$

Ground Truth Bounding Box
Training
Train What?

Convolution kernels

◦ To minimize a cost function
◦ Update kernels after every training image
Cost Function

**Cost function**

Logistic regression:

\[
J(\theta) = -\frac{1}{m} \left[ \sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right] + \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2
\]

Neural network:

\[
h_\Theta(x) \in \mathbb{R}^K \quad (h_\Theta(x))_i = i^{th} \text{ output}
\]

\[
J(\Theta) = -\frac{1}{m} \left[ \sum_{i=1}^{m} \sum_{k=1}^{K} y_k^{(i)} \log(h_\Theta(x^{(i)}))_k + (1 - y_k^{(i)}) \log(1 - (h_\Theta(x^{(i)}))_k) \right] + \frac{\lambda}{2m} \sum_{l=1}^{L-1} \sum_{i=1}^{s_l} \sum_{j=1}^{s_{i+1}} (\Theta_{ji}^{(l)})^2
\]
R-CNN Training: Step 1

Supervised pre-training
Train a SuperVision CNN* for the 1000-way ILSVRC image classification task

Auxiliary task: ILSVRC 2012 classification (1.2 million images)
R-CNN Training: Step 2

Fine-tune the CNN for detection
Transfer the representation learned for ILSVRC classification to PASCAL (or ImageNet detection)

Target task:
PASCAL VOC detection
(~25k object labels)
R-CNN Training: Step 3

Train detection SVMs
(With the softmax classifier from fine-tuning mAP decreases from 54% to 51%)
Tuning: Worth it?
Tuning: Worth it?

<table>
<thead>
<tr>
<th></th>
<th>VOC 2007</th>
<th>VOC 2010</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regionlets (Wang et al. 2013)</td>
<td>41.7%</td>
<td>39.7%</td>
</tr>
<tr>
<td>SegDPM (Fidler et al. 2013)</td>
<td>40.4%</td>
<td></td>
</tr>
<tr>
<td>R-CNN pool₅</td>
<td>44.2%</td>
<td></td>
</tr>
<tr>
<td>R-CNN fc₆</td>
<td>46.2%</td>
<td></td>
</tr>
<tr>
<td>R-CNN fc₇</td>
<td>44.7%</td>
<td></td>
</tr>
<tr>
<td>R-CNN FT pool₅</td>
<td>47.3%</td>
<td></td>
</tr>
<tr>
<td>R-CNN FT fc₆</td>
<td>53.1%</td>
<td></td>
</tr>
<tr>
<td>R-CNN FT fc₇</td>
<td>54.2%</td>
<td>50.2%</td>
</tr>
</tbody>
</table>
Tuning: Worth it?

No fine-tuning
Tuning: Worth it?
Tuning: Worth it?

After bounding-box regression
Performance
## R-CNN, PASCAL Results

<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. 2013)</td>
<td>41.7%</td>
<td>35.1%</td>
</tr>
<tr>
<td>Regionlets (Wang et al. 2013)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SegDPM (Fidler et al. 2013)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>R-CNN</strong></td>
<td>54.2%</td>
<td>50.2%</td>
</tr>
<tr>
<td><strong>R-CNN + bbox regression</strong></td>
<td>58.5%</td>
<td>53.7%</td>
</tr>
</tbody>
</table>
ImageNet Detection

200 object categories instead of 20

- Can this approach work?
ImageNet Detection

200 object categories instead of 20

- Can this approach work? Yes!
ImageNet Detection

200 object categories instead of 20
  ◦ Can this approach work? Yes!
Results
What did the CNN learn?

Visualize images that activate pool5 a feature
What did the CNN learn?
What did the CNN learn?
What did the CNN learn?
What did the CNN learn?
What did the CNN learn?
False-Positives
False-Positive Distribution

Loc = localization
Sim = similar classes
Oth = other / dissimilar classes
BG = background

Analysis software: D. Hoiem, Y. Chodpathumwan, and Q. Dai.
False-Positive?
False-Positive?

1949 French comedy by Jacques Tati
Conclusion
R-CNN Conclusion

- Dramatically better PASCAL mAP
- Outperforms other CNN-based methods
- Detection speed manageable (~11s/image on GPU)
- Scales very well (30ms for 20 → 200 classes!)
- Relatively simple and open source
Questions