Object Detection I

CS 4391 Introduction Computer Vision
Professor Yu Xiang
The University of Texas at Dallas
Object Detection

• Localize objects in images and classify them

Why using bounding boxes?
• Easy to store
  • \((x, y, w, h)\): box center with width, height
  • \((x_1, y_1, x_2, y_2)\): top left corner and bottom right corner

• Easy for image processing
  • Crop a region
Object Detection

• Localization + Classification
Localization: Sliding Window

• Select a window with a fixed size

• Scan the input image with the window (bounding box)

• How to deal with different object scales and aspect ratios?
  • Use boxes with different aspect ratios
  • Image pyramid

https://cvexplained.wordpress.com/tag/sliding-windows/
Localization: Region Proposal

• Leverage methods that can generate regions with high likelihood of containing objects
  • E.g., bottom-up segmentation methods, using edges

Selective Search, Sande et al., ICCV’11

Edge Boxes. Zitnick & Dollar, ECCV’14
Classification: Features

• Traditional methods: Hand-crafted features
• Deep learning methods: learned features in the network
Classification: Classifiers

• Traditional methods
  • AdaBoost
  • Support vector machines (SVMs)

• Deep learning methods
  • Neural networks

Viola and Jones: AdaBoost

Felzenszwalb et al: SVM
Object detection with discriminatively trained part-based models. TPAMI, 2009.
R-CNN

1. Input image
2. Extract region proposals (~2k)
3. Compute CNN features
4. Classify regions

Selective Search

Rich feature hierarchies for accurate object detection and semantic segmentation. Girshick et al., CVPR, 2014
### R-CNN

<table>
<thead>
<tr>
<th>VOC 2007 test</th>
<th>aero</th>
<th>bike</th>
<th>bird</th>
<th>boat</th>
<th>bottle</th>
<th>bus</th>
<th>car</th>
<th>cat</th>
<th>chair</th>
<th>cow</th>
<th>table</th>
<th>dog</th>
<th>horse</th>
<th>mbike</th>
<th>person</th>
<th>plant</th>
<th>sheep</th>
<th>sofa</th>
<th>train</th>
<th>tv</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-CNN pool&lt;sub&gt;5&lt;/sub&gt;</td>
<td>51.8</td>
<td>60.2</td>
<td>36.4</td>
<td>27.8</td>
<td>23.2</td>
<td>52.8</td>
<td>60.6</td>
<td>49.2</td>
<td>18.3</td>
<td>47.8</td>
<td>44.3</td>
<td>40.8</td>
<td>56.6</td>
<td>58.7</td>
<td>42.4</td>
<td>23.4</td>
<td>46.1</td>
<td>36.7</td>
<td>51.3</td>
<td>55.7</td>
<td>44.2</td>
</tr>
<tr>
<td>R-CNN fc&lt;sub&gt;6&lt;/sub&gt;</td>
<td>59.3</td>
<td>61.8</td>
<td>43.1</td>
<td>34.0</td>
<td>25.1</td>
<td>53.1</td>
<td>60.6</td>
<td>52.8</td>
<td>21.7</td>
<td>47.8</td>
<td>42.7</td>
<td>47.8</td>
<td>52.5</td>
<td>58.5</td>
<td>44.6</td>
<td>25.6</td>
<td>48.3</td>
<td>34.0</td>
<td>53.1</td>
<td>58.0</td>
<td>46.2</td>
</tr>
<tr>
<td>R-CNN fc&lt;sub&gt;7&lt;/sub&gt;</td>
<td>57.6</td>
<td>57.9</td>
<td>38.5</td>
<td>31.8</td>
<td>23.7</td>
<td>51.2</td>
<td>58.9</td>
<td>51.4</td>
<td>20.0</td>
<td>50.5</td>
<td>40.9</td>
<td>46.0</td>
<td>51.6</td>
<td>55.9</td>
<td>43.3</td>
<td>23.3</td>
<td>48.1</td>
<td>35.3</td>
<td>51.0</td>
<td>57.4</td>
<td>44.7</td>
</tr>
<tr>
<td>R-CNN FT pool&lt;sub&gt;5&lt;/sub&gt;</td>
<td>58.2</td>
<td>63.3</td>
<td>37.9</td>
<td>27.6</td>
<td>26.1</td>
<td>54.1</td>
<td>66.9</td>
<td>51.4</td>
<td>26.7</td>
<td>55.5</td>
<td>43.4</td>
<td>43.1</td>
<td>57.7</td>
<td>59.0</td>
<td>45.8</td>
<td>28.1</td>
<td>50.8</td>
<td>40.6</td>
<td>53.1</td>
<td>56.4</td>
<td>47.3</td>
</tr>
<tr>
<td>R-CNN FT fc&lt;sub&gt;6&lt;/sub&gt;</td>
<td>63.5</td>
<td>66.0</td>
<td>47.9</td>
<td>37.7</td>
<td>29.9</td>
<td>62.5</td>
<td>70.2</td>
<td>60.2</td>
<td>32.0</td>
<td>57.9</td>
<td>47.0</td>
<td>53.5</td>
<td>60.1</td>
<td>64.2</td>
<td>52.2</td>
<td>31.3</td>
<td>55.0</td>
<td>50.0</td>
<td>57.7</td>
<td>63.0</td>
<td>53.1</td>
</tr>
<tr>
<td>R-CNN FT fc&lt;sub&gt;7&lt;/sub&gt;</td>
<td>64.2</td>
<td>69.7</td>
<td>50.0</td>
<td>41.9</td>
<td>32.0</td>
<td>62.6</td>
<td>71.0</td>
<td>60.7</td>
<td>32.7</td>
<td>58.5</td>
<td>46.5</td>
<td>56.1</td>
<td>60.6</td>
<td>66.8</td>
<td>54.2</td>
<td>31.5</td>
<td>52.8</td>
<td>48.9</td>
<td>57.9</td>
<td>64.7</td>
<td>54.2</td>
</tr>
<tr>
<td><strong>R-CNN FT fc&lt;sub&gt;7&lt;/sub&gt; BB</strong></td>
<td>68.1</td>
<td>72.8</td>
<td>56.8</td>
<td>43.0</td>
<td>36.8</td>
<td>66.3</td>
<td>74.2</td>
<td>67.6</td>
<td>34.4</td>
<td>63.5</td>
<td>54.5</td>
<td>61.2</td>
<td>69.1</td>
<td>68.6</td>
<td>58.7</td>
<td>33.4</td>
<td>62.9</td>
<td>51.1</td>
<td>62.5</td>
<td>64.8</td>
<td>58.5</td>
</tr>
</tbody>
</table>

---

**BB:** bounding box regression

**Features from AlexNet**

---

Rich feature hierarchies for accurate object detection and semantic segmentation. Girshick et al., CVPR, 2014
Fast R-CNN

Fast R-CNN. Girshick, ICCV, 2015
**RoI Pooling**

Divide the mapping RoI into $H \times W$ grids

![RoI mapping to feature map](image)

$$s \times (x, y, h, w)$$

$$s = \frac{1}{16}$$

Max pooling for each grid

$7 \times 7$ RoI pooling in Fast R-CNN
Bounding Box Regression

• Predict bounding box regression offset for K object classes

\[ t_k = (t_{x_k}, t_{y_k}, t_{w_k}, t_{h_k}) \]

Offset

\[ G = (G_x, G_y, G_w, G_h) \]

G: ground truth

\[ P = (P_x, P_y, P_w, P_h) \]

P: input RoI

\[
\begin{align*}
t_x &= (G_x - P_x) / P_w \\
t_y &= (G_y - P_y) / P_h \\
t_w &= \log(G_w / P_w) \\
t_h &= \log(G_h / P_h) \\
\end{align*}
\]

G: ground truth, P: input RoI

\[
\begin{align*}
\hat{G}_x &= P_w d_x(P) + P_x \\
\hat{G}_y &= P_h d_y(P) + P_y \\
\hat{G}_w &= P_w \exp(d_w(P)) \\
\hat{G}_h &= P_h \exp(d_h(P)).
\end{align*}
\]
Fast R-CNN

• Loss function

\[
L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda[u \geq 1]L_{\text{loc}}(t^u, v)
\]

Softmax classification probabilities

\[
p = (p_0, \ldots, p_K)
\]

True class label

\[
t^u = (t^u_x, t^u_y, t^u_w, t^u_h)
\]

Bounding box regress prediction

\[
L_{\text{loc}}(t^u, v) = \sum_{i \in \{x,y,w,h\}} \text{smooth}_{L_1}(t^u_i - v_i)
\]

Bounding box regress target

\[
\text{smooth}_{L_1}(x) = \begin{cases} 
0.5x^2 & \text{if } |x| < 1 \\
|x| - 0.5 & \text{otherwise}
\end{cases}
\]
## Fast R-CNN

<table>
<thead>
<tr>
<th></th>
<th>Fast R-CNN</th>
<th>R-CNN</th>
<th>SPPnet</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>S</td>
<td>M</td>
<td>L</td>
</tr>
<tr>
<td>train time (h)</td>
<td>1.2</td>
<td>2.0</td>
<td>9.5</td>
</tr>
<tr>
<td>train speedup</td>
<td>18.3×</td>
<td>14.0×</td>
<td>8.8×</td>
</tr>
<tr>
<td>test rate (s/im)</td>
<td>0.10</td>
<td>0.15</td>
<td>0.32</td>
</tr>
<tr>
<td>▶ with SVD</td>
<td>0.06</td>
<td>0.08</td>
<td>0.22</td>
</tr>
<tr>
<td>test speedup</td>
<td>98×</td>
<td>80×</td>
<td>146×</td>
</tr>
<tr>
<td>▶ with SVD</td>
<td>169×</td>
<td>150×</td>
<td>213×</td>
</tr>
<tr>
<td>VOC07 mAP</td>
<td>57.1</td>
<td>59.2</td>
<td>66.9</td>
</tr>
<tr>
<td>▶ with SVD</td>
<td>56.5</td>
<td>58.7</td>
<td>66.6</td>
</tr>
</tbody>
</table>

S: AlexNet, M: VGG, L: deep VGG
SVD for FCs layers

\[ W \approx U \Sigma_t V^T \]
Faster R-CNN

• A single network for object detection
  • Region proposal network
  • Classification network

Region Proposal Network

An anchor is centered at the sliding window in question, and is associated with a scale and aspect ratio (3 scales, 3 aspect ratios, k=9)
Two stage vs One stage

- Two stage detection methods
  - Stage 1: generate region proposals
  - Stage 2: classify region proposals and refine their locations
  - E.g., R-CNN, Fast R-CNN, Faster R-CNN

- One stage detection methods
  - An end-to-end network for object detection
  - E.g., YOLO
Summary

• Two-stage detectors
  • R-CNN, Fast R-CNN, Faster R-CNN
  • Region proposal + classification
  • Good performance, slow
Further Reading

• Viola–Jones object detection, 2001
  https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf

• Deformable part model, 2010,

