Project Admin

Someone from every team must come and see me!

If you chose:
- Depth Prediction
- Neural Style Transfer
- Image Captioning

I will outline what project involves IMMEDIATELY after this class

For other projects, come and see me during office hours.
Object Detection

Image Classification
(what?)

Object Detection
(what + where?)
Detection with ConvNets

• So far, all about classification

• What about localizing objects within the scene?
Two General Approaches

1. Examine very position / scale
   - E.g. Overfeat: Integrated recognition, localization and detection using convolutional networks, Sermanet et al., ICLR 2014

2. Use some kind of proposal mechanism to attend to a set of possible regions
   - E.g. Region-CNN [Rich feature hierarchies for accurate object detection and semantic segmentation, Girshick et al., CVPR 2014]
Sliding Window with ConvNet
Sliding Window with ConvNet

Input Window

224

Filter size 7
Stride 2
Image size 224

Conv

110

55

Layer 1

Conv

26

224

Layer 2

13

3x3 max pool
Stride 2

96

Layer 3

Conv

13

3x3 max pool
Stride 2

96

Layer 4

Conv

13

3x3 max pool
Stride 2

256

Layer 5

Conv

256

4096 units

Layer 6

Full

4096 units

Layer 7

Full

Output

C

Class softmax

Feature Extractor

224

224

Full

Classifier

6

256

6

C

Classes
Sliding Window with ConvNet

No need to compute two separate windows --- Just one big input window
Multi-Scale Sliding Window ConvNet
Multi-Scale Sliding Window ConvNet

Feature Maps

Bounding Box Maps

Feature Extractor

Regression Network

4

256

4

4

4
OverFeat – Output before NMS
Overfeat Detection Results

[Sermanet et al. ICLR 2014]
Two General Approaches

1. Examine very position / scale
   - E.g. Overfeat: Integrated recognition, localization and detection using convolutional networks, Sermanet et al., ICLR 2014

2. Use some kind of proposal mechanism to attend to a set of possible regions
   - E.g. Region-CNN [Rich feature hierarchies for accurate object detection and semantic segmentation, Girshick et al., CVPR 2014]
Object Detection: R-CNN

Region-based CNN pipeline

Object Detection: R-CNN

• R-CNN

Object Detection: Fast R-CNN

• Fast R-CNN

pre-computed Regions-of-Interest (RoIs)

shared conv layers

CNN

feature

feature

feature

RoI pooling

End-to-End training

Girshick. Fast R-CNN. ICCV 2015
Object Detection: Faster R-CNN

- Faster R-CNN
  - Solely based on CNN
  - No external modules
  - Each step is end-to-end

Object Detection

- **ImageNet data**
  - backbone structure
  - pre-train

- **Classification network**
  - features

- **Detection network**
  - fine-tune

- **“plug-in” features**
  - AlexNet
  - VGG-16
  - GoogleNet
  - ResNet-101
  - ...

- **Independently developed detectors**
  - R-CNN
  - Fast R-CNN
  - Faster R-CNN
  - MultiBox
  - SSD
  - ...

- **Detectors**
# Object Detection

- Simply “Faster R-CNN + ResNet”

<table>
<thead>
<tr>
<th>Faster R-CNN baseline</th>
<th>mAP@.5</th>
<th>mAP@.5:.95</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG-16</td>
<td>41.5</td>
<td>21.5</td>
</tr>
<tr>
<td>ResNet-101</td>
<td>48.4</td>
<td>27.2</td>
</tr>
</tbody>
</table>

**COCO detection results**

**ResNet-101 has 28% relative gain vs VGG-16**

---

Object Detection

• RPN learns proposals by extremely deep nets
  • We use only 300 proposals (no hand-designed proposals)

• Add components:
  • Iterative localization
  • Context modeling
  • Multi-scale testing

• All components are based on CNN features; all steps are end-to-end

• All benefit more from deeper features – cumulative gains!

ResNet’s object detection result on COCO

*the original image is from the COCO dataset


*the original image is from the COCO dataset*
Results on real video. Models trained on MS COCO (80 categories). (frame-by-frame; no temporal processing)

FAIR COCO Object Detection
Sergey Zagoruyko*, Tsung-Yi Lin*, Pedro Pinheiro*, Adam Lerer, Sam Gross, Soumith Chintala, Piotr Dollár

(*equal contribution)
## Results

<table>
<thead>
<tr>
<th></th>
<th>APbbox</th>
<th>AP small</th>
<th>AP medium</th>
<th>AP large</th>
<th>AR max=100</th>
<th>AP segm</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSRA</td>
<td>0.373</td>
<td>0.183</td>
<td>0.419</td>
<td>0.524</td>
<td>0.491</td>
<td>0.282</td>
</tr>
<tr>
<td><strong>FAIRCNN</strong></td>
<td><strong>0.335</strong></td>
<td><strong>0.139</strong></td>
<td><strong>0.378</strong></td>
<td><strong>0.477</strong></td>
<td><strong>0.485</strong></td>
<td><strong>0.251</strong></td>
</tr>
<tr>
<td>ION</td>
<td>0.310</td>
<td>0.123</td>
<td>0.332</td>
<td>0.447</td>
<td>0.457</td>
<td></td>
</tr>
<tr>
<td>FastRCNN</td>
<td>0.197</td>
<td>0.035</td>
<td>0.188</td>
<td>0.346</td>
<td>0.298</td>
<td></td>
</tr>
</tbody>
</table>

66% improvement over FastRCNN baseline
Overview

I. DeepMask segmentation proposals [Pinheiro NIPS 15]
   + iterative localization
   + top-down refinement

II. Fast R-CNN object detector [Girshick ICCV 15]
   + foveal context regions
   + modified loss function
   + skip connections
   + ensembling
I. Deep MASK Object Proposals
Model:

VGG

x: 3x224x224

DeepMask Framework
DeepMask Framework

Model:

\[ x: 3x224x224 \]

- VGG
- \( 512x14x14 \)
- \( 512x1x1 \)
- \( 1024x1x1 \)

Segmentation Mask:

- \( f_{\text{segm}}(x): 224x224 \)
- \( f_{\text{score}}(x): 1x1 \)
DeepMask Framework

Model:

VGG

x: 3x224x224

1x1 conv

512x14x14

2x2 pool

512x7x7

512x1x1

56x56

f_{segm}(x): 224x224

512x1x1

f_{score}(x): 1x1

1024x1x1

‘Objectness’ score
DeepMask Framework

Model:

Input: $x: 3x224x224$

VGG

1x1 conv

2x2 pool

512x14x14

512x7x7

512x1x1

56x56

512x14x14

512x1x1

1024x1x1

$f_{segm}(x): 224x224$

$f_{score}(x): 1x1$
DeepMask Framework

Model:

\[ x: 3 \times 224 \times 224 \]
DeepMask Framework

Model:

x: 3x224x224

VGG

1x1 conv

2x2 pool

512x14x14

512x14x14

512x1x1

512x7x7

512x14x14

512x1x1

56x56

f_{segm}(x): 224x224

f_{score}(x): 1x1

1024x1x1
DeepMask Framework

Model:

$x: 3x224x224$
Single Scale Inference

image

scores

masks
Single Scale Inference
New: Iterative Localization (+1.0 AP)
New: Top-Down Refinement (+0.7 AP)
Proposal Quality (boxes)

![Graph showing proposal quality in boxes as a function of the number of proposals](image)

- **DeepMask++**
- **DeepMask**
- **MCG**
- **SelectiveSearch**
- **Rigor**
- **Geodesic**
- **EdgeBoxes**

**Average Recall**

- 0
- 0.1
- 0.2
- 0.3
- 0.4
- 0.5
- 0.6

**# Proposals**

- $10^0$
- $10^1$
- $10^2$
- $10^3$
DeepMask Object Proposals
II. Classification framework
• Fast R-CNN setup [Girshick, ICCV15]
• Fast R-CNN setup [Girshick, ICCV15]
• Foveal structure [inspired by Gidaris & Komodakis, ICCV15] (+2 AP)
• Fast R-CNN setup [Girshick, ICCV15]
• Foveal structure [inspired by Gidaris & Komodakis, ICCV15] (+2 AP)
• Skip connections (+1 AP)
• Fast R-CNN setup [Girshick, ICCV15]
• Foveal structure [inspired by Gidaris & Komodakis, ICCV15] (+2 AP)
• Skip connections (+1 AP)
Multi-threshold Loss (+1.5 AP)

- Multi-head classifier
- Improves localization sensitivity
Inference

- Base Model 30.1 AP
- + horizontal flip 31.1 AP
- + ROI Pooling ‘2 crop’ 32.1 AP
- + 7-model Ensemble 33.5 AP
AP

<table>
<thead>
<tr>
<th>Size</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small</td>
<td>0.139</td>
</tr>
<tr>
<td>Medium</td>
<td>0.378</td>
</tr>
<tr>
<td>Large</td>
<td>0.477</td>
</tr>
</tbody>
</table>

Overall performance across different sizes:

- **Overall-all-small**:
  - C75: 0.117
  - C50: 0.240
  - Loc: 0.340
  - Sim: 0.345
  - Oth: 0.342
  - BG: 0.576
  - FN: 1.00
  - AP: 0.139

- **Overall-all-medium**:
  - C75: 0.427
  - C50: 0.600
  - Loc: 0.681
  - Sim: 0.692
  - Oth: 0.706
  - BG: 0.930
  - FN: 1.00
  - AP: 0.378

- **Overall-all-large**:
  - C75: 0.544
  - C50: 0.706
  - Loc: 0.784
  - Sim: 0.816
  - Oth: 0.944
  - BG: 1.00
  - FN: 1.00
  - AP: 0.477
Segmentation Examples

DeepMask → Proposal BBoxes → FastRCNN → Scored BBoxes → DeepMask → Scored Segments
Future Directions

• most room for improvement:
  • background confusion (FP/FN)
  • small objects
• more effective use of context
• fast / proposal-free detection
Questions?