Object Detection: Impact of Deep Learning

Figure copyright Ross Girshick, 2015. Reproduced with permission.
Object Detection: Single Object
(Classification + Localization)

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Fully Connected: 4096 to 1000

Vector: 4096

Fully Connected: 4096 to 4

Box Coordinates (x, y, w, h)

Treat localization as a regression problem!
Object Detection: Single Object
(Classification + Localization)

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Correct label:
Cat

Fully Connected:
4096 to 1000

Softmax Loss

Correct box:
(x', y', w', h')

L2 Loss

Box Coordinates
(x, y, w, h)

Fully Connected:
4096 to 4

Vector:
4096

Treat localization as a regression problem!

This image is CC0 public domain
Object Detection: Single Object
(Classification + Localization)

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01...

Vector:
4096

Fully Connected:
4096 to 1000

Correct label:
Cat

Softmax Loss

Multitask Loss

Correct box:
(x’, y’, w’, h’)

Box Coordinates
(x, y, w, h)

L2 Loss

Treat localization as a regression problem!
Object Detection: Single Object
(Classification + Localization)

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Vector: 4096

Fully Connected: 4096 to 1000

Softmax Loss

Correct label:
Cat

Loss

Correct box:
(x', y', w', h')

L2 Loss

Box Coordinates
(x, y, w, h)

Treat localization as a regression problem!

Often pretrained on ImageNet (Transfer learning)

This image is CC0 public domain

Fei-Fei Li & Justin Johnson & Serena Yeung

Lecture 12 -  40  May 14, 2019
Object Detection: Multiple Objects

CAT: (x, y, w, h)

DOG: (x, y, w, h)

DUCK: (x, y, w, h)

....
Object Detection: Multiple Objects

Each image needs a different number of outputs!

CAT: \((x, y, w, h)\)  \(4\) numbers

DOG: \((x, y, w, h)\)  \(16\) numbers

DUCK: \((x, y, w, h)\)  Many numbers!
Object Detection: Multiple Objects

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background.

Dog? NO
Cat? NO
Background? YES
Object Detection: Multiple Objects

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

Dog? YES
Cat? NO
Background? NO
Object Detection: Multiple Objects

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background.
Object Detection: Multiple Objects

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background.

Dog? NO
Cat? YES
Background? NO
Object Detection: Multiple Objects

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background.

Problem: Need to apply CNN to huge number of locations, scales, and aspect ratios, very computationally expensive!

Dog? NO
Cat? YES
Background? NO
Region Proposals: Selective Search

- Find “blobby” image regions that are likely to contain objects
- Relatively fast to run; e.g. Selective Search gives 2000 region proposals in a few seconds on CPU

Alexe et al, “Measuring the objectness of image windows”, TPAMI 2012
Cheng et al, “BING: Binarized normed gradients for objectness estimation at 300fps”, CVPR 2014
Zitnick and Dollar, “Edge boxes: Locating object proposals from edges”, ECCV 2014
R-CNN

Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.
R-CNN


Figure copyright Ross Girshick, 2015; source. Reproduced with permission.
R-CNN

Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.
R-CNN

Forward each region through ConvNet

Warped image regions (224x224 pixels)

Regions of Interest (RoI) from a proposal method (~2k)

R-CNN

Classify regions with SVMs

Forward each region through ConvNet

Warped image regions (224x224 pixels)

Regions of Interest (RoI) from a proposal method (~2k)

Figure copyright Ross Girshick, 2015; [source](source). Reproduced with permission.
R-CNN

Predict “corrections” to the RoI: 4 numbers: \((dx, dy, dw, dh)\)

- Classify regions with SVMs
- Forward each region through ConvNet
- Warped image regions (224x224 pixels)
- Regions of Interest (RoI) from a proposal method (\(~2k\))

Figure copyright Ross Girshick, 2015; source. Reproduced with permission.
R-CNN

Predict “corrections” to the RoI: 4 numbers: (dx, dy, dw, dh)

Classify regions with SVMs

Forward each region through ConvNet

Warped image regions (224x224 pixels)

Regions of Interest (RoI) from a proposal method (~2k)

Problem: Very slow! Need to do ~2k independent forward passes for each image!

Problem: Very slow! Need to do ~2k independent forward passes for each image!

Idea: Process image before cropping! Swap convolution and cropping!

---

Fast R-CNN

“Slow” R-CNN

Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.
Fast R-CNN

“Backbone” network: AlexNet, VGG, ResNet, etc

“conv5” features

Run whole image through ConvNet

ConvNet

Input image

“Slow” R-CNN

SVMs

Conv Net

Conv Net

Conv Net

SVMs

Input image

Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.
Fast R-CNN

Regions of Interest (RoIs) from a proposal method

“Backbone” network: AlexNet, VGG, ResNet, etc

“conv5” features

Run whole image through ConvNet

ConvNet

Input image

“Slow” R-CNN

SVMs

ConvNet

SVMs

ConvNet

Input image

Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.
Fast R-CNN

Regions of Interest (RoIs) from a proposal method

“Backbone” network: AlexNet, VGG, ResNet, etc

Crop + Resize features
“conv5” features
Run whole image through ConvNet

“Slow” R-CNN

Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.
Fast R-CNN

Regions of Interest (RoIs) from a proposal method

“Backbone” network: AlexNet, VGG, ResNet, etc

Input image

CNN

Per-Region Network

Crop + Resize features

“conv5” features

Run whole image through ConvNet

“Slow” R-CNN

SVMs

Conv Net

Conv Net

Conv Net

Input image

Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.
Fast R-CNN

Object category

Regions of Interest (RoIs) from a proposal method

“Backbone” network: AlexNet, VGG, ResNet, etc

ConvNet

Input image

CNN

Per-Region Network

Crop + Resize features

“conv5” features

Run whole image through ConvNet

Linear + softmax

Linear

Box offset

“Slow” R-CNN

SVMs

SVMs

Conv Net

Conv Net

Conv Net

Input image

Fei-Fei Li & Justin Johnson & Serena Yeung

Lecture 12 - 62 May 14, 2019
Fast R-CNN

Regions of Interest (RoIs) from a proposal method

"Backbone" network: AlexNet, VGG, ResNet, etc

Crop + Resize features

“conv5” features

Run whole image through ConvNet

Per-Region Network

Box offset

Linear

Linear + softmax

CNN

Input image

“Slow” R-CNN

SVMs

Conv Net

SVMs

Conv Net

SVMs

Conv Net

Input image

Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.
Cropping Features: RoI Pool

Input Image (e.g. 3 x 640 x 480)

CNN

Image features (e.g. 512 x 20 x 15)

Cropping Features: RoI Pool

Project proposal onto features

Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 20 x 15)

CNN
Cropping Features: RoI Pool

Project proposal onto features

“Snap” to grid cells

Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 20 x 15)

Cropping Features: RoI Pool

Input Image
(e.g. 3 x 640 x 480)

CNN

Image features
(e.g. 512 x 20 x 15)

Project proposal onto features

“Snap” to grid cells

Divide into 2x2 grid of (roughly) equal subregions

Cropping Features: RoI Pool

Project proposal onto features

“Snap” to grid cells

Divide into 2x2 grid of (roughly) equal subregions

Max-pool within each subregion

Region features (here 512 x 2 x 2; in practice e.g. 512 x 7 x 7)

Region features always the same size even if input regions have different sizes!


Input Image (e.g. 3 x 640 x 480)

CNN

Image features (e.g. 512 x 20 x 15)
Cropping Features: RoI Pool

Project proposal onto features

“Snap” to grid cells

Divide into 2x2 grid of (roughly) equal subregions

Max-pool within each subregion

Region features always the same size even if input regions have different sizes!

Problem: Region features slightly misaligned

Input Image (e.g. 3 x 640 x 480)

CNN

Image features (e.g. 512 x 20 x 15)

Region features (here 512 x 2 x 2; In practice e.g. 512 x 7 x 7)
Cropping Features: RoI Align

He et al, "Mask R-CNN", ICCV 2017
Cropping Features: RoI Align

Input Image (e.g. 3 x 640 x 480)

CNN

Image features (e.g. 512 x 20 x 15)

Sample at regular points in each subregion using bilinear interpolation

No “snapping”!

He et al, “Mask R-CNN”, ICCV 2017
Cropping Features: RoI Align

Project proposal onto features

No “snapping”!

Sample at regular points in each subregion using bilinear interpolation

Feature $f_{xy}$ for point $(x, y)$ is a linear combination of features at its four neighboring grid cells:

He et al, “Mask R-CNN”, ICCV 2017
Cropping Features: RoI Align

Project proposal onto features

No “snapping”!

Sample at regular points in each subregion using bilinear interpolation

Feature $f_{xy}$ for point $(x, y)$ is a linear combination of features at its four neighboring grid cells:

$$f_{xy} = \sum_{i,j=1}^{2} f_{i,j} \max(0, 1 - |x - x_i|) \max(0, 1 - |y - y_j|)$$
Cropping Features: RoI Align

Project proposal onto features

Sample at regular points in each subregion using bilinear interpolation

No “snapping”!

Max-pool within each subregion

Region features (here 512 x 2 x 2; In practice e.g 512 x 7 x 7)

He et al, "Mask R-CNN", ICCV 2017
R-CNN vs Fast R-CNN

Training time (Hours)

<table>
<thead>
<tr>
<th>Model</th>
<th>Time (Hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-CNN</td>
<td>84</td>
</tr>
<tr>
<td>SPP-Net</td>
<td>25.5</td>
</tr>
<tr>
<td>Fast R-CNN</td>
<td>8.75</td>
</tr>
</tbody>
</table>

Test time (seconds)

<table>
<thead>
<tr>
<th>Model</th>
<th>Test time (including Region proposals)</th>
<th>Test time (excluding Region proposals)</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-CNN</td>
<td>49</td>
<td>47</td>
</tr>
<tr>
<td>SPP-Net</td>
<td>2.3</td>
<td>2.3</td>
</tr>
<tr>
<td>Fast R-CNN</td>
<td>2.3</td>
<td>0.32</td>
</tr>
</tbody>
</table>

Girshick, "Fast R-CNN", ICCV 2015
R-CNN vs Fast R-CNN

**Training time (Hours)**
- R-CNN: 84 hours
- SPP-Net: 25.5 hours
- Fast R-CNN: 8.75 hours

**Test time (seconds)**
- Including Region proposals:
  - R-CNN: 49 seconds
  - SPP-Net: 4.3 seconds
  - Fast R-CNN: 2.3 seconds
- Excluding Region proposals:
  - R-CNN: 47 seconds
  - SPP-Net: 2.3 seconds
  - Fast R-CNN: 0.32 seconds

**Problem:**
Runtime dominated by region proposals!

References:
- Girshick, "Fast R-CNN", ICCV 2015
**Faster R-CNN:**
Make CNN do proposals!

Insert **Region Proposal Network (RPN)** to predict proposals from features.

Otherwise same as Fast R-CNN:
Crop features for each proposal, classify each one.

---

Figure copyright 2015, Ross Girshick; reproduced with permission
Region Proposal Network

Input Image
(e.g. 3 x 640 x 480)

CNN

Image features
(e.g. 512 x 20 x 15)
Region Proposal Network

Imagine an anchor box of fixed size at each point in the feature map.

Input Image (e.g. 3 x 640 x 480)

Image features (e.g. 512 x 20 x 15)
Region Proposal Network

Imagine an anchor box of fixed size at each point in the feature map.

At each point, predict whether the corresponding anchor contains an object (per-pixel logistic regression).

Input Image (e.g. 3 x 640 x 480)

Image features (e.g. 512 x 20 x 15)

Anchor is an object? 1 x 20 x 15
Imagine an anchor box of fixed size at each point in the feature map.

For positive boxes, also predict a transformation from the anchor to the ground-truth box (regress 4 numbers per pixel).
Region Proposal Network

In practice use $K$ different anchor boxes of different size / scale at each point.

Anchor is an object? $K \times 20 \times 15$

Box transforms $4K \times 20 \times 15$

Input Image (e.g. $3 \times 640 \times 480$)

Image features (e.g. $512 \times 20 \times 15$)
Region Proposal Network

In practice use K different anchor boxes of different size / scale at each point.

- Anchor is an object?
  - \( K \times 20 \times 15 \)

- Box transforms
  - \( 4K \times 20 \times 15 \)

Sort the \( K \times 20 \times 15 \) boxes by their “object” score, take top ~300 as our proposals.
Faster R-CNN: Make CNN do proposals!

Jointly train with 4 losses:
1. RPN classify object / not object
2. RPN regress box coordinates
3. Final classification score (object classes)
4. Final box coordinates

Figure copyright 2015, Ross Girshick; reproduced with permission
**Faster R-CNN:**
Make CNN do proposals!

**R-CNN Test-Time Speed**

- **R-CNN:** 49
- **SPP-Net:** 4.3
- **Fast R-CNN:** 2.3
- **Faster R-CNN:** 0.2
Faster R-CNN:
Make CNN do proposals!

Glossing over many details:
- Ignore overlapping proposals with **non-max suppression**
- How to determine whether a proposal is positive or negative?
- How many positives / negatives to send to second stage?
- How to parameterize bounding box regression?

Figure copyright 2015, Ross Girshick; reproduced with permission
Faster R-CNN:
Make CNN do proposals!

Faster R-CNN is a Two-stage object detector

First stage: Run once per image
- Backbone network
- Region proposal network

Second stage: Run once per region
- Crop features: RoI pool / align
- Predict object class
- Prediction bbox offset
Faster R-CNN: Make CNN do proposals!

Faster R-CNN is a Two-stage object detector

First stage: Run once per image
- Backbone network
- Region proposal network

Second stage: Run once per region
- Crop features: RoI pool / align
- Predict object class
- Prediction bbox offset

Do we really need the second stage?

Classification loss
Bounding-box regression loss

Proposal
Region Proposal Network
Feature map
CNN
Rol pooling
Single-Stage Object Detectors: YOLO / SSD / RetinaNet

Input image 3 x H x W

Divide image into grid 7 x 7

Image a set of base boxes centered at each grid cell

Here B = 3

Within each grid cell:
- Regress from each of the B base boxes to a final box with 5 numbers: (dx, dy, dh, dw, confidence)
- Predict scores for each of C classes (including background as a class)
- Looks a lot like RPN, but category-specific!

Output:
7 x 7 x (5 * B + C)

Lin et al, “Focal Loss for Dense Object Detection”, ICCV 2017
Object Detection: Lots of variables ...

<table>
<thead>
<tr>
<th>Backbone Network</th>
<th>“Meta-Architecture”</th>
<th>Takeaways</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG16</td>
<td>Two-stage: Faster R-CNN</td>
<td>Faster R-CNN is slower but more accurate</td>
</tr>
<tr>
<td>ResNet-101</td>
<td>Single-stage: YOLO / SSD</td>
<td>SSD is much faster but not as accurate</td>
</tr>
<tr>
<td>Inception V2</td>
<td>Hybrid: R-FCN</td>
<td>Bigger / Deeper backbones work better</td>
</tr>
<tr>
<td>Inception V3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inception</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ResNet</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MobileNet</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Image Size

# Region Proposals

...
Object Detection: Lots of variables ...

<table>
<thead>
<tr>
<th>Backbone Network</th>
<th>“Meta-Architecture”</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG16</td>
<td>Two-stage: Faster R-CNN</td>
</tr>
<tr>
<td>ResNet-101</td>
<td>Single-stage: YOLO / SSD</td>
</tr>
<tr>
<td>Inception V2</td>
<td>Hybrid: R-FCN</td>
</tr>
<tr>
<td>Inception V3</td>
<td>Image Size</td>
</tr>
<tr>
<td>Inception</td>
<td># Region Proposals</td>
</tr>
<tr>
<td>ResNet</td>
<td>...</td>
</tr>
<tr>
<td>MobileNet</td>
<td></td>
</tr>
</tbody>
</table>

**Takeaways**
- Faster R-CNN is slower but more accurate
- SSD is much faster but not as accurate
- Bigger / Deeper backbones work better

Huang et al, “Speed/accuracy trade-offs for modern convolutional object detectors”, CVPR 2017