CS 1674: Intro to Computer Vision

Object Recognition

Prof. Adriana Kovashka
University of Pittsburgh
October 29, 2020
Different Flavors of Object Recognition

- **Semantic Segmentation**:
  - Categories: GRASS, CAT, TREE, SKY
  - Description: No objects, just pixels

- **Classification + Localization**
  - Categories: CAT
  - Description: Single Object

- **Object Detection**
  - Categories: DOG, DOG, CAT
  - Description: Multiple Object

- **Instance Segmentation**
  - Categories: DOG, DOG, CAT
  - Description: Multiple Object

Adapted from Justin Johnson
Plan for the next three lectures

• Detection approaches
  – Pre-CNNs
    • Detection with whole windows: Pedestrian detection
    • Part-based detection: Deformable Part Models
  – Post-CNNs
    • Detection with region proposals: R-CNN, Fast R-CNN, Faster-R-CNN
    • Detection without region proposals: YOLO, SSD

• Segmentation approaches
  – Semantic segmentation: FCN
  – Instance segmentation: Mask R-CNN
Object Detection

No objects, just pixels

Single Object

Multiple Object

Slide by: Justin Johnson
Object detection: basic framework

- Build/train object model
- Generate candidate regions in new image
- Score the candidates

Adapted from Kristen Grauman
Window-template-based models
Building an object model

Given the representation, train a binary classifier

Kristen Grauman
Window-template-based models
Generating and scoring candidates

Car/non-car Classifier

Kristen Grauman
Window-template-based object detection: recap

Training:
1. Obtain training data
2. Define features
3. Define classifier

Given new image:
1. Slide window
2. Score by classifier
Evaluating detection methods

\[
\text{mAP} = \frac{1}{|\text{classes}|} \sum_{c \in \text{classes}} \frac{\#TP(c)}{\#TP(c) + \#FP(c)}
\]

• True Positive - TP(c): a predicted bounding box (pred_bb) was made for class c, there is a ground truth bounding box (gt_bb) of class c, and IoU(pred_bb, gt_bb) >= 0.5.
• False Positive - FP(c): a pred_bb was made for class c, and there is no gt_bb of class c. Or there is a gt_bb of class c, but IoU(pred_bb, gt_bb) < 0.5.
Dalal-Triggs pedestrian detector

1. Extract fixed-sized (64x128 pixel) window at multiple positions and scales
2. Compute HOG (histogram of gradient) features within each window
3. Score the window with a linear SVM classifier
4. Perform non-maxima suppression to remove overlapping detections with lower scores

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05
Histograms of oriented gradients (HOG)

Divide image into 8x8 regions

Orientation: 9 bins (for unsigned angles)

Votes weighted by magnitude

Histograms in 8x8 pixel cells

Adapted from Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05
Train SVM for pedestrian detection using HoG

$$0.16 = w^T x + b$$

$$\text{sign}(0.16) = 1$$

=> pedestrian

Adapted from Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05
Remove overlapping detections

Non-max suppression

Score = 0.1

Score = 0.8

Score = 0.8

Adapted from Derek Hoiem
Are window templates enough?

- Many objects are articulated, or have parts that can vary in configuration

- Many object categories look very different from different viewpoints, or from instance to instance

Images from Caltech-256, D. Ramanan
Parts-based Models

Define object by collection of parts modeled by

1. Appearance
2. Spatial configuration

Slide credit: Rob Fergus
How to model spatial relations?

• One extreme: fixed template
Fixed part-based template

- Object model = sum of scores of features at fixed positions

\[+3 +2 -2 -1 -2.5 = -0.5 > 7.5\]
Non-object

\[+4 +1 +0.5 +3 +0.5 = 10.5 > 7.5\]
Object
How to model spatial relations?

• Another extreme: bag of words
How to model spatial relations?

• Star-shaped model
Parts-based Models

• Articulated parts model
  – Object is configuration of parts
  – Each part is detectable and can move around
# Deformable Part Models

<table>
<thead>
<tr>
<th>Root filter</th>
<th>Part filters</th>
<th>Deformation weights</th>
</tr>
</thead>
</table>

P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, [Object Detection with Discriminatively Trained Part Based Models](https://dx.doi.org/10.1109/TPAMI.2010.31), PAMI 32(9), 2010
Scoring an object hypothesis

• The score of a hypothesis is the sum of appearance scores minus the sum of deformation costs

\[ z = (p_0, \ldots, p_n) \]

\[ p_0 : \text{location of root} \]

\[ p_1, \ldots, p_n : \text{location of parts} \]

\[ \text{part loc} \quad \text{anchor loc (where we expect to see part)} \]

\[ (dx_i, dy_i) = (x_i, y_i) - (2(x_0, y_0) + v_i) \]

\[ \text{Displacements} \]

i.e. how much the part \( p_i \) moved from its expected anchor location in the \( x, y \) directions

\[ \text{Appearance weights} \quad \text{Part features} \quad \text{Deformation weights} \]

\[ \text{i.e. how much we'll penalize the part } p_i \text{ for moving from its expected location} \]
Detection

Felzenszwalb et al.
Car model

Component 1

Component 2
Car detections

high scoring true positives

high scoring false positives
Person model
Person detections

high scoring true positives

high scoring false positives
(not enough overlap)
Cat model
Cat detections

high scoring true positives

high scoring false positives
(not enough overlap)
“Sliding window” detector
Plan for the next three lectures

• Detection approaches
  – Pre-CNNs
    • Detection with whole windows: Pedestrian detection
    • Part-based detection: Deformable Part Models
  – Post-CNNs
    • Detection with region proposals: R-CNN, Fast R-CNN, Faster-R-CNN
    • Detection without region proposals: YOLO, SSD

• Segmentation approaches
  – Semantic segmentation: FCN
  – Instance segmentation: Mask R-CNN
Complexity and the plateau

[Source: http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc20{07,08,09,10,11,12}/results/index.html]

Impact of Deep Learning

Slide by: Justin Johnson
Before: Image Classification with CNNs

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

Vector: 4096

Fully-Connected: 4096 to 1000
Classification + Localization

- **No objects, just pixels**
- **Single Object**
- **Multiple Object**

Slide by: Justin Johnson
Classification + Localization

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

Vector:
4096

Fully Connected:
4096 to 1000

Box Coordinates
(x, y, w, h)

Treat localization as a regression problem!

Slide by: Justin Johnson
Classification + Localization

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

Correct label:
- Cat

Softmax Loss

Fully Connected:
- 4096 to 1000

Box Coordinates
- $(x, y, w, h)$

L2 Loss

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

Treat localization as a regression problem!

Vector:
- 4096

Fully Connected:
- 4096 to 4

Slide by: Justin Johnson
Classification + Localization

Class Scores

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

Correct label: Cat

Softmax Loss

Multitask Loss

Fully Connected: 4096 to 1000

Vector: 4096

Fully Connected: 4096 to 4

Box Coordinates

(x, y, w, h)

L2 Loss

Correct box:

(x', y', w', h')

Treat localization as a regression problem!

Slide by: Justin Johnson
Classification + Localization

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

Correct label:
- Cat

Softmax Loss

L2 Loss

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

Treat localization as a regression problem!

Often pretrained on ImageNet (Transfer learning)

Vector:
- 4096

Fully Connected:
- 4096 to 1000

Box Coordinates
- \((x, y, w, h)\)
- \((x', y', w', h')\)

Fully Connected:
- 4096 to 4

Slide by: Justin Johnson
Object Detection as Regression?

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

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

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

DUCK: \((x, y, w, h)\)

DUCK: \((x, y, w, h)\)

....
Object Detection as Regression?

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

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

DUCK: \((x, y, w, h)\)

Each image needs a different number of outputs!
Object Detection as Classification: Sliding Window

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 as Classification: Sliding Window

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

Dog? YES
Cat? NO
Background? NO

Slide by: Justin Johnson
Object Detection as Classification: Sliding Window

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 as Classification: Sliding Window

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 as Classification: Sliding Window

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 and scales, very computationally expensive!
Region Proposals

- Find “blobby” image regions that are likely to contain objects
- Relatively fast to run; e.g. Selective Search gives 1000 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
Speeding up detection: Restrict set of windows we pass through SVM to those w/ high “objectness”

Fig. 1: Desired behavior of an objectness measure. The desired objectness measure should score the blue windows, partially covering the objects, lower than the ground truth windows (green), and score even lower the red windows containing only stuff or small parts of objects.

Alexe et al., CVPR 2010
Objectness cue #1: Where people look

Fig. 2: MS success and failure.
Fig. 3: CC success and failure. Success: the windows containing the objects (cyan) have high color contrast with their surrounding ring (yellow) in images (a) and (b). Failure: the color contrast for windows in cyan in image (c) is much lower.
Objectness cue #3: no segments “straddling” the object box

Fig. 5: The SS cue. Given the segmentation (b) of image (a), for a window $w$ we compute $\text{SS}(w, \theta_{SS})$ (eq. 4). In (c), most of the surface of $w_1$ is covered by superpixels contained almost entirely inside it. Instead, all superpixels passing by $w_2$ continue largely outside it. Therefore, $w_1$ has a higher SS score than $w_2$. The window $w_3$ has an even higher score as it fits the object tightly.
R-CNN

R-CNN

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

Input image

R-CNN

R-CNN

R-CNN

ConvNet

ConvNet

ConvNet

SVMs

SVMs

SVMs

Input image

Classify regions with SVMs

Forward each region through ConvNet

Warped image regions

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

R-CNN
Linear Regression for bounding box offsets

Classify regions with SVMs

Forward each region through ConvNet

Warped image regions

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

Input image

R-CNN: Regions with CNN features

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

R-CNN at test time: Step 1

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

Proposal-method agnostic, many choices
- Selective Search [van de Sande, Uijlings et al.] (Used in this work)
- Objectness [Alexe et al.]
- Category independent object proposals [Endres & Hoiem]
- CPMC [Carreira & Sminchisescu]

R-CNN at test time: Step 2

R-CNN at test time: Step 2

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

Dilate proposal

R-CNN at test time: Step 2

Input image

Extract region proposals (~2k / image)

Compute CNN features

```
aeroplane? no.
: person? yes.
: tvmonitor? no.
```

R-CNN at test time: Step 2

Input image

Extract region proposals (~2k / image)

Compute CNN features

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

R-CNN at test time: Step 2

R-CNN at test time: Step 3

Input image

Extract region proposals (~2k / image)

Compute CNN features

Classify regions

proposals

4096-dimensional fc7 feature vector

linear classifiers (SVM or softmax)

Step 4: Object proposal refinement

Original proposal

Linear regression on CNN features

Predicted object bounding box

Bounding-box regression

R-CNN on ImageNet detection

R-CNN

Linear Regression for bounding box offsets

Classify regions with SVMs

Forward each region through ConvNet

Warped image regions

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

Input image

Post hoc component

What’s wrong with slow R-CNN?

• Ad-hoc training objectives
  • Train post-hoc linear SVMs (hinge loss)
  • Train post-hoc bounding-box regressions (L2 loss)

• Training is slow (84h), takes a lot of disk space
  • Need to store all region crops

• Inference (detection) is slow
  • 47s / image with VGG16 [Simonyan & Zisserman, ICLR15]

Adapted from Girshick, “Fast R-CNN”, ICCV 2015
Fast R-CNN

- One network, applied one time, not 2000 times
- Trained end-to-end (in one stage)
- Fast test time
- Higher mean average precision

Adapted from Girshick, “Fast R-CNN”, ICCV 2015
Fast R-CNN

Fast R-CNN

“conv5” feature map of image

Forward whole image through ConvNet

ConvNet

Input image

Fast R-CNN

Regions of Interest (RoIs) from a proposal method

“conv5” feature map of image

Forward whole image through ConvNet

ConvNet

Input image

Fast R-CNN

“RoI Pooling” layer
“conv5” feature map of image
Forward whole image through ConvNet

Regions of Interest (RoIs) from a proposal method

ConvNet

Input image

Fast R-CNN

Fast R-CNN

Fast R-CNN (Training)

Log loss + Smooth L1 loss

Linear + softmax

Linear

FCs

ConvNet

Input image

Multi-task loss

Fast R-CNN (Training)

Log loss + Smooth L1 loss

Multi-task loss

Linear + softmax

Linear

FCs

ConvNet

Input image

# Fast R-CNN vs R-CNN

<table>
<thead>
<tr>
<th></th>
<th>Fast R-CNN</th>
<th>R-CNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train time (h)</td>
<td>9.5</td>
<td>84</td>
</tr>
<tr>
<td>Speedup</td>
<td>8.8x</td>
<td>1x</td>
</tr>
<tr>
<td>Test time / image</td>
<td>0.32s</td>
<td>47.0s</td>
</tr>
<tr>
<td>Test speedup</td>
<td>146x</td>
<td>1x</td>
</tr>
<tr>
<td>mAP</td>
<td>66.9%</td>
<td>66.0%</td>
</tr>
</tbody>
</table>

Timings exclude object proposal time, which is equal for all methods. All methods use VGG16 from Simonyan and Zisserman.

*Girshick, “Fast R-CNN”, ICCV 2015*
Faster R-CNN

Make CNN do proposals!

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

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

Accurate object detection is slow!

<table>
<thead>
<tr>
<th></th>
<th>Pascal 2007 mAP</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM v5</td>
<td>33.7</td>
<td>.07 FPS</td>
</tr>
<tr>
<td>R-CNN</td>
<td>66.0</td>
<td>.05 FPS</td>
</tr>
</tbody>
</table>

⅓ Mile, 1760 feet

Accurate object detection is slow!

<table>
<thead>
<tr>
<th></th>
<th>Pascal 2007 mAP</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM v5</td>
<td>33.7</td>
<td>.07 FPS</td>
</tr>
<tr>
<td>R-CNN</td>
<td>66.0</td>
<td>.05 FPS</td>
</tr>
<tr>
<td>Fast R-CNN</td>
<td>70.0</td>
<td>.5 FPS</td>
</tr>
<tr>
<td>Faster R-CNN</td>
<td>73.2</td>
<td>7 FPS</td>
</tr>
<tr>
<td>YOLO</td>
<td>69.0</td>
<td>45 FPS</td>
</tr>
</tbody>
</table>

Detection without Proposals: YOLO

Each cell predicts boxes and confidences: $P(\text{Object})$

Each cell also predicts a probability
\[ P(\text{Class} \mid \text{Object}) \]

<table>
<thead>
<tr>
<th>Bicycle</th>
<th>Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dog</td>
<td></td>
</tr>
</tbody>
</table>

Combine the box and class predictions

Finally do NMS and threshold detections

This parameterization fixes the output size

Each cell predicts:

- For each bounding box:
  - 4 coordinates (x, y, w, h)
  - 1 confidence value
- Some number of class probabilities

For Pascal VOC:

- 7x7 grid
- 2 bounding boxes / cell
- 20 classes

\[ 7 \times 7 \times (2 \times 5 + 20) = 7 \times 7 \times 30 \text{ tensor} = 1470 \text{ outputs} \]

YOLO works across many natural images

It also generalizes well to new domains

Redmon and Farhadi, “YOLO9000: Better, Faster, Stronger”, CVPR 2017
Redmon and Farhadi, “YOLO9000: Better, Faster, Stronger”, CVPR 2017
Redmon and Farhadi, “YOLO9000: Better, Faster, Stronger”, CVPR 2017
Redmon and Farhadi, “YOLO9000: Better, Faster, Stronger”, CVPR 2017
Plan for the next two lectures

• Detection approaches
  – Pre-CNNs
    • Detection with whole windows: Pedestrian detection
    • Part-based detection: Deformable Part Models
  – Post-CNNs
    • Detection with region proposals: R-CNN, Fast R-CNN, Faster-R-CNN
    • Detection without region proposals: YOLO, SSD

• Segmentation approaches
  – Semantic segmentation: FCN
  – Instance segmentation: Mask R-CNN
Semantic Segmentation

- **GRASS, CAT, TREE, SKY**
  - No objects, just pixels

- **CAT**
  - Single Object

- **DOG, DOG, CAT**
  - Multiple Object

Slide by: Justin Johnson
Semantic Segmentation

Label each pixel in the image with a category label

Don’t differentiate instances, only care about pixels
Semantic Segmentation Idea: Sliding Window

- Extract patch
- Classify center pixel with CNN

Full image

Cow

Cow

Grass

Farabet et al, "Learning Hierarchical Features for Scene Labeling," TPAMI 2013
Pinheiro and Collobert, "Recurrent Convolutional Neural Networks for Scene Labeling", ICML 2014
Semantic Segmentation Idea: Sliding Window

Problem: Very inefficient! Not reusing shared features between overlapping patches

Farabet et al, "Learning Hierarchical Features for Scene Labeling," TPAMI 2013
Pinheiro and Collobert, "Recurrent Convolutional Neural Networks for Scene Labeling", ICML 2014
Semantic Segmentation Idea: Fully Convolutional

Design a network as a bunch of convolutional layers to make predictions for pixels all at once!

Input: $3 \times H \times W$

Convolutions: $D \times H \times W$

Scores: $C \times H \times W$

Predictions: $H \times W$

Slide by: Justin Johnson
Lecture 11

Input: $3 \times H \times W$

Convolutions: $D \times H \times W$

Scores: $C \times H \times W$

Predictions: $H \times W$

Problem: convolutions at original image resolution will be very expensive...

Design a network as a bunch of convolutional layers to make predictions for pixels all at once!

Semantic Segmentation Idea: Fully Convolutional

Slide by: Justin Johnson
Semantic Segmentation Idea: Fully Convolutional

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!

- **Input**: \(3 \times H \times W\)
- **High-res**: \(D_1 \times H/2 \times W/2\)
- **Med-res**: \(D_2 \times H/4 \times W/4\)
- **Low-res**: \(D_3 \times H/4 \times W/4\)
- **High-res**: \(D_1 \times H/2 \times W/2\)
- **Predictions**: \(H \times W\)

Semantic Segmentation Idea: Fully Convolutional

**Downsampling:**
Pooling, strided convolution

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!

**Input:**
3 x H x W

**High-res:**
D₁ x H/2 x W/2

**Med-res:**
D₂ x H/4 x W/4

**Low-res:**
D₃ x H/4 x W/4

**High-res:**
D₁ x H/2 x W/2

**Predictions:**
H x W

---


---

Slide by: Justin Johnson
In-Network upsampling: “Unpooling”

Nearest Neighbor

Input: 2 x 2

Output: 4 x 4

“Bed of Nails”

Input: 2 x 2

Output: 4 x 4
In-Network upsampling: “Max Unpooling”

Max Pooling
Remember which element was max!

Max Unpooling
Use positions from pooling layer

Input: 4 x 4

Output: 2 x 2

Rest of the network

Input: 2 x 2

Output: 4 x 4

Corresponding pairs of downsampling and upsampling layers

Slide by: Justin Johnson
Learnable Upsampling: Transpose Convolution

3 x 3 transpose convolution, stride 2 pad 1

Input: 2 x 2
Output: 4 x 4
Learnable Upsampling: Transpose Convolution

3 x 3 transpose convolution, stride 2 pad 1

Input: 2 x 2
Output: 4 x 4
Learnable Upsampling: Transpose Convolution

3 x 3 transpose convolution, stride 2 pad 1

Input: 2 x 2

Output: 4 x 4

Input gives weight for filter

Sum where output overlaps

Filter moves 2 pixels in the output for every one pixel in the input

Stride gives ratio between movement in output and input

Slide by: Justin Johnson
Transpose Convolution: 1D Example

Output contains copies of the filter weighted by the input, summing at where at overlaps in the output.
Instance Segmentation

- **GRASS, CAT, TREE, SKY** (No objects, just pixels)
- **CAT** (Single Object)
- **DOG, DOG, CAT** (Multiple Object)

Slide by: Justin Johnson
What is Mask R-CNN: Parallel Heads

• Easy, fast to implement and use

(slow) R-CNN

Fast/er R-CNN

Mask R-CNN

He et al, “Mask R-CNN”, ICCV 2017
Mask R-CNN

Classification Scores: C
Box coordinates (per class): 4 * C

Predict a mask for each of C classes

Adapted from Justin Johnson