From image classification to object detection

Image classification

Object detection

Slides from L. Lazebnik

Image source
What are the challenges of object detection?

- Images may contain more than one class, multiple instances from the same class
- Bounding box localization
- Evaluation
Outline

• Task definition and evaluation

• Generic object detection before deep learning
  • Sliding windows
  • HoG, DPMs (Components, Parts)
  • Region Classification Methods

• Deep detection approaches
  • R-CNN
  • Fast R-CNN
  • Faster R-CNN
  • SSD
Object detection evaluation

- At test time, predict bounding boxes, class labels, and confidence scores
- For each detection, determine whether it is a true or false positive
  - PASCAL criterion: \( \frac{\text{Area}(\text{GT} \cap \text{Det})}{\text{Area}(\text{GT} \cup \text{Det})} > 0.5 \)
  - For multiple detections of the same ground truth box, only one considered a true positive
Object detection evaluation

- At test time, predict bounding boxes, class labels, and confidence scores
- For each detection, determine whether it is a true or false positive
- For each class, plot Recall-Precision curve and compute Average Precision (area under the curve)
- Take mean of AP over classes to get mAP

**Precision:**
true positive detections / total detections

**Recall:**
true positive detections / total positive test instances
PASCAL VOC Challenge (2005-2012)

- 20 challenge classes:
  - Person
  - Animals: bird, cat, cow, dog, horse, sheep
  - Vehicles: aeroplane, bicycle, boat, bus, car, motorbike, train
  - Indoor: bottle, chair, dining table, potted plant, sofa, tv/monitor

- Dataset size (by 2012): 11.5K training/validation images, 27K bounding boxes, 7K segmentations

http://host.robots.ox.ac.uk/pascal/VOC/
Progress on PASCAL detection

PASCAL VOC

mean Average Precision (mAP) vs year

Before CNNs
Newer benchmark: COCO

What is COCO?

COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

-Object segmentation
-Recognition in context
-Superpixel stuff segmentation
-330K images (>200K labeled)
-1.5 million object instances
-80 object categories
-91 stuff categories
-5 captions per image
-250,000 people with keypoints

http://cocodataset.org/#home
COCO detection metrics

<table>
<thead>
<tr>
<th>Metric Description</th>
<th>Equation</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Precision (AP):</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AP</td>
<td>% AP at IoU=.50:.05:.95 (primary challenge metric)</td>
<td></td>
</tr>
<tr>
<td>AP\text{IoU=.50}</td>
<td>% AP at IoU=.50 (PASCAL VOC metric)</td>
<td></td>
</tr>
<tr>
<td>AP\text{IoU=.75}</td>
<td>% AP at IoU=.75 (strict metric)</td>
<td></td>
</tr>
<tr>
<td>AP Across Scales:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AP\text{small}</td>
<td>% AP for small objects: area &lt; 32²</td>
<td></td>
</tr>
<tr>
<td>AP\text{medium}</td>
<td>% AP for medium objects: 32² &lt; area &lt; 96²</td>
<td></td>
</tr>
<tr>
<td>AP\text{large}</td>
<td>% AP for large objects: area &gt; 96²</td>
<td></td>
</tr>
<tr>
<td>Average Recall (AR):</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AR\text{max=1}</td>
<td>% AR given 1 detection per image</td>
<td></td>
</tr>
<tr>
<td>AR\text{max=10}</td>
<td>% AR given 10 detections per image</td>
<td></td>
</tr>
<tr>
<td>AR\text{max=100}</td>
<td>% AR given 100 detections per image</td>
<td></td>
</tr>
<tr>
<td>AR Across Scales:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AR\text{small}</td>
<td>% AR for small objects: area &lt; 32²</td>
<td></td>
</tr>
<tr>
<td>AR\text{medium}</td>
<td>% AR for medium objects: 32² &lt; area &lt; 96²</td>
<td></td>
</tr>
<tr>
<td>AR\text{large}</td>
<td>% AR for large objects: area &gt; 96²</td>
<td></td>
</tr>
</tbody>
</table>

- Leaderboard: [http://cocodataset.org/#detection-leaderboard](http://cocodataset.org/#detection-leaderboard)
- Current best mAP: ~52%
- Official COCO challenges no longer include detection
- More emphasis on instance segmentation and dense segmentation
Detection before deep learning
Conceptual approach: Sliding window detection

- Slide a window across the image and evaluate a detection model at each location
  - Thousands of windows to evaluate: efficiency and low false positive rates are essential
  - Difficult to extend to a large range of scales, aspect ratios
Histograms of oriented gradients (HOG)

- Partition image into blocks and compute histogram of gradient orientations in each block

Image credit: N. Snavely

N. Dalal and B. Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005
Pedestrian detection with HOG

• Train a pedestrian template using a linear support vector machine

positive training examples

negative training examples

N. Dalal and B. Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005
Pedestrian detection with HOG

- Train a pedestrian template using a linear support vector machine
- At test time, convolve feature map with template
- Find local maxima of response
- For multi-scale detection, repeat over multiple levels of a HOG pyramid

Discriminative part-based models

- Single rigid template usually not enough to represent a category
  - Many objects (e.g. humans) are articulated, or have parts that can vary in configuration

- Many object categories look very different from different viewpoints, or from instance to instance
Discriminative part-based models

Root filter

Part filters

Deformation weights

P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, Object Detection with Discriminatively Trained Part Based Models, PAMI 32(9), 2010
Discriminative part-based models

Multiple components

P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, Object Detection with Discriminatively Trained Part Based Models, PAMI 32(9), 2010
Discriminative part-based models

P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, Object Detection with Discriminatively Trained Part Based Models, PAMI 32(9), 2010
Progress on PASCAL detection

PASCAL VOC

Before CNNs

mean Average Precision (mAP) vs. year

2006 2007 2008 2009 2010 2011 2012
Conceptual approach: Proposal-driven detection

- Generate and evaluate a few hundred region proposals
  - Proposal mechanism can take advantage of low-level perceptual organization cues
  - Proposal mechanism can be category-specific or category-independent, hand-crafted or trained
  - Classifier can be slower but more powerful
Multiscale Combinatorial Grouping

- Use hierarchical segmentation: start with small superpixels and merge based on diverse cues.
Region Proposals for Detection (Eval)

Pascal SegVOC12

- MCG-Our
- SCG-Our
- CPMC [6]
- CI [17]
- GOP [12]
- GLS [13]
- RIGOR [14]
- ShSh [16]
- SeSe [11]
- RP [23]
- EB [24]
- BING [25]
- Obj [15]
- Quadtree

P. Arbelaez. et al., **Multiscale Combinatorial Grouping**, CVPR 2014
Region Proposals for Detection

- Feature extraction: color SIFT, codebook of size 4K, spatial pyramid with four levels = 360K dimensions

J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders, Selective Search for Object Recognition, IJCV 2013
Another proposal method: EdgeBoxes

- Box score: number of edges in the box minus number of edges that overlap the box boundary
- Uses a trained edge detector
- Uses efficient data structures (incl. integral images) for fast evaluation
- Gets 75% recall with 800 boxes (vs. 1400 for Selective Search), is 40 times faster

R-CNN: Region proposals + CNN features

Source: R. Girshick

R-CNN details

- **Regions**: ~2000 Selective Search proposals
- **Network**: AlexNet *pre-trained* on ImageNet (1000 classes), *fine-tuned* on PASCAL (21 classes)
- **Final detector**: warp proposal regions, extract fc7 network activations (4096 dimensions), classify with linear SVM
- **Bounding box regression** to refine box locations
- **Performance**: mAP of **53.7%** on PASCAL 2010 (vs. **35.1%** for Selective Search and **33.4%** for Deformable Part Models)
R-CNN pros and cons

- **Pros**
  - Accurate!
  - Any deep architecture can immediately be “plugged in”

- **Cons**
  - Not a single end-to-end system
    - Fine-tune network with softmax classifier (log loss)
    - Train post-hoc linear SVMs (hinge loss)
    - Train post-hoc bounding-box regressions (least squares)
  - Training is slow (84h), takes a lot of disk space
    - 2000 CNN passes per image
  - Inference (detection) is slow (47s / image with VGG16)
Fast R-CNN

Forward whole image through ConvNet

Region proposals

Conv5 feature map of image

RoI Pooling layer

Fully-connected layers

Bounding-box regressors

Linear + softmax

Linear

Softmax classifier

FCs

Source: R. Girshick

R. Girshick, Fast R-CNN, ICCV 2015
RoI pooling

- “Crop and resample” a fixed-size feature representing a region of interest out of the outputs of the last conv layer
  - Use nearest-neighbor interpolation of coordinates, max pooling

Source: R. Girshick, K. He
RoI pooling illustration

Input

0.88 0.44 0.14 0.16 0.37 0.77 0.96 0.27
0.19 0.45 0.57 0.16 0.63 0.29 0.71 0.70
0.66 0.26 0.82 0.64 0.54 0.73 0.59 0.26
0.85 0.34 0.76 0.84 0.29 0.75 0.62 0.25
0.32 0.74 0.21 0.39 0.34 0.03 0.33 0.48
0.20 0.14 0.16 0.13 0.73 0.65 0.96 0.32
0.19 0.69 0.09 0.86 0.88 0.07 0.01 0.48
0.83 0.24 0.97 0.04 0.24 0.35 0.50 0.91
Prediction

- For each RoI, network predicts probabilities for C+1 classes (class 0 is background) and four bounding box offsets for C classes
Fast R-CNN training

Log loss + smooth L1 loss

Multi-task loss

Linear + softmax

Linear

Trainable

FCs

ConvNet

Source: R. Girshick, Fast R-CNN, ICCV 2015
Multi-task loss

- Loss for ground truth class $y$, predicted class probabilities $P(y)$, ground truth box $b$, and predicted box $\hat{b}$:

$$L(y, P, b, \hat{b}) = -\log P(y) + \lambda \mathbb{I}[y \geq 1] L_{\text{reg}}(b, \hat{b})$$

  softmax loss

  regression loss

- Regression loss: *smooth L1 loss* on top of log space offsets relative to proposal

$$L_{\text{reg}}(b, \hat{b}) = \sum_{i=\{x,y,w,h\}} \text{smooth}_{L_1}(b_i - \hat{b}_i)$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 
0.5x^2 & \text{if } |x| < 1 \\
|x| - 0.5 & \text{otherwise}
\end{cases}$$
Bounding box regression

- Ground truth box
- Target offset to predict*
- Predicted offset
- Region proposal (a.k.a default box, prior, reference, anchor)
- Predicted box

*Typically in transformed, normalized coordinates
## Fast R-CNN results

<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.

Source: R. Girshick
Region proposal network (RPN)

- Slide a small window (3x3) over the conv5 layer
  - Predict object/no object
  - Regress bounding box coordinates with reference to anchors (3 scales x 3 aspect ratios)
One network, four losses

Classification loss
Bounding-box regression loss

Region Proposal Network
proposals

CNN

Bounding-box regression loss

feature map

Rol pooling

Source: R. Girshick, K. He
Faster R-CNN results

detection mAP on PASCAL VOC 2007, with VGG-16 pre-trained on ImageNet
Object detection progress

Before CNNs

After CNNs

Faster R-CNN

Fast R-CNN

R-CNNv1

mean Average Precision (mAP)

year
Streamlined detection architectures

- The Faster R-CNN pipeline separates proposal generation and region classification:

  - RPN
  - Region Proposals
  - RoI pooling
  - RoI features
  - Classification + Regression
  - Detections

- Is it possible do detection in one shot?
SSD

Similarly to RPN, use anchors and directly predict class-specific bounding boxes.

SSD: Results (PASCAL 2007)

- More accurate and faster than YOLO and Faster R-CNN

<table>
<thead>
<tr>
<th>Method</th>
<th>mAP</th>
<th>FPS</th>
<th>batch size</th>
<th># Boxes</th>
<th>Input resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster R-CNN (VGG16)</td>
<td>73.2</td>
<td>7</td>
<td>1</td>
<td>~ 6000</td>
<td>~ 1000 x 600</td>
</tr>
<tr>
<td>Fast YOLO</td>
<td>52.7</td>
<td>155</td>
<td>1</td>
<td>98</td>
<td>448 x 448</td>
</tr>
<tr>
<td>YOLO (VGG16)</td>
<td>66.4</td>
<td>21</td>
<td>1</td>
<td>98</td>
<td>448 x 448</td>
</tr>
<tr>
<td>SSD300</td>
<td>74.3</td>
<td>46</td>
<td>1</td>
<td>8732</td>
<td>300 x 300</td>
</tr>
<tr>
<td>SSD512</td>
<td>76.8</td>
<td>19</td>
<td>1</td>
<td>24564</td>
<td>512 x 512</td>
</tr>
<tr>
<td>SSD300</td>
<td>74.3</td>
<td>59</td>
<td>8</td>
<td>8732</td>
<td>300 x 300</td>
</tr>
<tr>
<td>SSD512</td>
<td>76.8</td>
<td>22</td>
<td>8</td>
<td>24564</td>
<td>512 x 512</td>
</tr>
</tbody>
</table>
Multi-resolution prediction

- SSD predicts boxes of different size from different conv maps, but each level of resolution has its own predictors and higher-level context does not get propagated back to lower-level feature maps.
- Can we have a more elegant multi-resolution prediction architecture?
Feature Pyramid Networks

(a) Featurized image pyramid
(b) Single feature map
(c) Pyramidal feature hierarchy
(d) Feature Pyramid Network

Feature pyramid networks

- Improve predictive power of lower-level feature maps by adding contextual information from higher-level feature maps.
- Predict different sizes of bounding boxes from different levels of the pyramid (but share parameters of predictors).

RetinaNet

- Combine feature pyramid network with focal loss to reduce the standard cross-entropy loss for well-classified examples

Review: R-CNN

Review: Fast R-CNN

R. Girshick, Fast R-CNN, ICCV 2015
Review: Faster R-CNN

Review: RPN

- Slide a small window (3x3) over the conv5 layer
  - Predict object/no object
  - Regress bounding box coordinates with reference to anchors (3 scales x 3 aspect ratios)
Review: SSD

Summary: Object detection with CNNs

- **R-CNN**: region proposals + CNN on cropped, resampled regions
- **Fast R-CNN**: region proposals + RoI pooling on top of a conv feature map
- **Faster R-CNN**: RPN + RoI pooling
- **Next generation of detectors**
  - Direct prediction of BB offsets, class scores on top of conv feature maps
  - Get better context by combining feature maps at multiple resolutions
Evaluation

- Average Precision like detection, except region IoU as opposed to box IoU.

B. Hariharan et al., *Simultaneous Detection and Segmentation*, ECCV 2014
Mask R-CNN

- Mask R-CNN = Faster R-CNN + FCN on RoIs

K. He, G. Gkioxari, P. Dollar, and R. Girshick, Mask R-CNN, ICCV 2017 (Best Paper Award)
RoIAlign vs. RoIPool

- RoIPool: nearest neighbor quantization

K. He, G. Gkioxari, P. Dollar, and R. Girshick, Mask R-CNN, ICCV 2017 (Best Paper Award)
RoIAlign vs. RoIPool

- **RoIPool**: nearest neighbor quantization
- **RoIAlign**: bilinear interpolation

Mask R-CNN

- From RoIAlign features, predict class label, bounding box, and segmentation mask

Mask R-CNN

K. He, G. Gkioxari, P. Dollar, and R. Girshick, Mask R-CNN, ICCV 2017 (Best Paper Award)
Example results
Example results
### Instance segmentation results on COCO

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>AP</th>
<th>AP$_{50}$</th>
<th>AP$_{75}$</th>
<th>AP$_S$</th>
<th>AP$_M$</th>
<th>AP$_L$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNC [10]</td>
<td>ResNet-101-C4</td>
<td>24.6</td>
<td>44.3</td>
<td>24.8</td>
<td>4.7</td>
<td>25.9</td>
<td>43.6</td>
</tr>
<tr>
<td>FCIS [26] +OHEM</td>
<td>ResNet-101-C5-dilated</td>
<td>29.2</td>
<td>49.5</td>
<td>-</td>
<td>7.1</td>
<td>31.3</td>
<td>50.0</td>
</tr>
<tr>
<td>FCIS+++ [26] +OHEM</td>
<td>ResNet-101-C5-dilated</td>
<td>33.6</td>
<td>54.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td>ResNet-101-C4</td>
<td>33.1</td>
<td>54.9</td>
<td>34.8</td>
<td>12.1</td>
<td>35.6</td>
<td>51.1</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td>ResNet-101-FPN</td>
<td>35.7</td>
<td>58.0</td>
<td>37.8</td>
<td>15.5</td>
<td>38.1</td>
<td>52.4</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td>ResNeXt-101-FPN</td>
<td>37.1</td>
<td>60.0</td>
<td>39.4</td>
<td>16.9</td>
<td>39.9</td>
<td>53.5</td>
</tr>
</tbody>
</table>

AP at different IoU thresholds

AP for different size instances

K. He, G. Gkioxari, P. Dollar, and R. Girshick, [Mask R-CNN](#), ICCV 2017 (Best Paper Award)
Unifying Semantic and Instance Segm.

Abstract

We propose and study a task we name **panoptic segmentation** (PS). Panoptic segmentation unifies the typically distinct tasks of **semantic segmentation** (assign a class label to each pixel) and **instance segmentation** (detect and segment each object instance). The proposed task requires generating a coherent scene segmentation that is rich and complete, an important step toward real-world vision systems.

While early work in computer vision addressed related image/scene parsing tasks, these are not currently popular, possibly due to lack of appropriate metrics or associated recognition challenges. To address this, we propose a novel **panoptic quality** (PQ) metric that captures performance for all classes (stuff and things) in an interpretable and unified manner. Using the proposed metric, we perform a rigorous study of both human and machine performance for PS on three existing datasets, revealing interesting insights about the task. The aim of our work is to revive the interest of the community in a more unified view of image segmentation.

1. Introduction

In the early days of computer vision, things – countable objects such as people, animals, tools – received the dominant share of attention. Questioning the wisdom of this trend, Adelson [1] elevated the importance of studying systems that recognize stuff – amorphous regions of similar texture or material such as grass, sky, road. This dichotomy between stuff and things persists to this day, reflected in both the division of visual recognition tasks and in the specialized algorithms developed for stuff and thing tasks.

Studying stuff is most commonly formulated as a task known as **semantic segmentation**, see Figure 1b. As stuff is amorphous and uncountable, this task is defined as simply assigning a class label to each pixel in an image (note that semantic segmentation treats thing classes as stuff).

In contrast, studying things is typically formulated as the task of **object detection** or **instance segmentation**, where the goal is to detect each object and delineate it with a bounding box or segmentation mask, respectively, see Figure 1c.

While seemingly related, the datasets, details, and metrics for these two visual recognition tasks vary substantially.

The schism between semantic and instance segmentation has led to a parallel rift in the methods for these tasks. Stuff classifiers are usually built on fully convolutional nets [30] with dilations [52,53] while object detectors often use object proposals [15] and are region-based [37,14]. Overall algorithmic progress on these tasks has been incredible in the past decade, yet, something important may be overlooked by focussing on these tasks in isolation.

A natural question emerges: Can there be a reconciliation between stuff and things? And what is the most effective design of a unified vision system that generates rich and coherent scene segmentations? These questions are particularly important given their relevance in real-world applications, such as autonomous driving or augmented reality.

Interestingly, while semantic and instance segmentation dominate current work, in the pre-deep learning era there...
Keypoint prediction

- Given K keypoints, train model to predict K m x m one-hot maps