Part Models and Pixel Labeling

Computer Vision

CS 543 / ECE 549

University of Illinois

Derek Hoiem
Influential Works in Detection

• Sung-Poggio (1994, 1998) : ~2412 citations
  – Basic idea of statistical template detection, bootstrapping to get “face-like” negative examples, multiple whole-face prototypes (in 1994)

  – “Parts” at fixed position, neural network based detector, non-maxima suppression, simple cascade, rotation, pretty good accuracy, fast

• Viola-Jones (2001, 2004) : ~27,000
  – Haar-like features, Adaboost as feature selection, hyper-cascade, very fast, easy to implement

• Dalal-Triggs (2005) : ~18000
  – Careful feature engineering, excellent results, HOG feature, online code

• Felzenszwalb-Huttenlocher (2000): ~2100
  – Efficient way to solve part-based detectors

  – Excellent template/parts-based blend

• Girshick-Donahue-Darrell-Malik R-CNN (2014- ): ~4700
  – Region proposals + fine-tuned CNN features (marks significant advance in accuracy over hog-based methods)
CNN (Deep Learning) for Classification

- **Operations**
  - Convolution with learned filters, ReLU
    - Note: after first layer, filters operate on high dimensional feature maps, not intensities
  - Spatial pooling and downsample 2x
  - “Bottlenecks” to limit parameters
  - “Skip connections” to simplify optimization, enable ensemble behavior
  - Classification (with multilabel or softmax loss)

- **Tricks to train**
  - Batchnorm
  - ADAM / momentum
  - Data augmentation
  - Set parameters appropriately (weight initialization, learning rate schedule, momentum, weight decay)

Each filter: 3x3x512
Final feature map: 7x7x512

Fig: Deep Residual Learning for Image Recognition
He et al. CVPR 2006
CNN for Detection

• Classifier network produces a set of feature maps

• Each cell proposes bounding boxes that might be objects

• Features are pooled into bbox regions and classified into object categories or background

Faster R-CNN (Ren et al. 2016)
Object bounding box detections

Faster RCNN detections
Today’s class

• Object part models

• Pixel labeling
Part/keypoint Prediction

Semantic Segmentation


https://www.cityscapes-dataset.com/examples/#fine-annotations
Semantic Segmentation

http://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html
Deformable objects

Images from Caltech-256
Deformable objects

Images from D. Ramanan’s dataset
Compositional objects
Parts-based Models

Define object by collection of parts modeled by

1. Appearance
2. Spatial configuration
How to model spatial relations?

• One extreme: fixed template
How to model spatial relations?

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

- CNNs have flexible models through spatial pooling
How to model spatial relations?

• Articulated parts model
  – Object is configuration of parts
  – Each part is detectable
How to model spatial relations?

• Tree-shaped model
Pictorial Structures Model

Part = oriented rectangle  Spatial model = relative size/orientation

Felzenszwalb and Huttenlocher 2005
Pictorial Structures Model

\[ P(L|I, \theta) \propto \left( \prod_{i=1}^{n} p(I|l_i, u_i) \prod_{(v_i, v_j) \in E} p(l_i, l_j|c_{ij}) \right) \]

Appearance likelihood

Geometry likelihood
Modeling the Appearance

• Any appearance model could be used
  – HOG Templates, etc.
  – Here: rectangles fit to background subtracted binary map

• Can train appearance models independently (easy, not as good) or jointly (more complicated but better)

\[
P(L|I, \theta) \propto \left( \prod_{i=1}^{n} p(I|l_i, u_i) \prod_{(v_i, v_j) \in E} p(l_i, l_j|c_{ij}) \right)
\]

Appearance likelihood

Geometry likelihood
Part representation

- Background subtraction
Pictorial structures model

Optimization is tricky but can be efficient

$$L^* = \arg \min_L \left( \sum_{i=1}^{n} m_i(l_i) + \sum_{(v_i, v_j) \in E} d_{ij}(l_i, l_j) \right)$$

• For each $l_1$, find best $l_2$:

$$\text{Best}_2(l_1) = \min_{l_2} \left[ m_2(l_2) + d_{12}(l_1, l_2) \right]$$

• Remove $v_2$, and repeat with smaller tree, until only a single part

• For $k$ parts, $n$ locations per part, this has complexity of $O(kn^2)$, but can be solved in $\sim O(kn)$ using generalized distance transform
Distance Transform

• For each pixel \( p \), how far away is the nearest pixel \( q \) of set \( G \)
  
  \[ f(p) = \min_{q \in G} d(p, q) \]

  – \( G \) is often the set of edge pixels
Distance Transform - Applications

- Set distances – e.g. Hausdorff Distance
- Image processing – e.g. Blurring
- Robotics – Motion Planning
- Alignment
  - Edge images
  - Motion tracks
  - Audio warping
- Deformable Part Models
Generalized Distance Transform

- Original form: \( f(p) = \min_{q \in G} d(p, q) \)
- General form: \( f(p) = \min_{q \in [1, N]} m(q) + d(p, q) \)

- For many deformation costs, \( O(N^2) \rightarrow O(N) \)
  - Quadratic \( d(p, q) = \alpha(p - q)^2 + \beta(p - q) \)
  - Abs Diff \( d(p, q) = \alpha |p - q| \)
  - Min Composition \( d(p, q) = \min(d_1(p, q), d_2(p, q)) \)
  - Bounded \( d_\tau(p, q) = \begin{cases} d(p, q) & : |p - q| < \tau \\ \infty & : |p - q| \geq \tau \end{cases} \)
Results for person matching
Results for person matching
Enhanced pictorial structures

- Learn spatial prior
- Color models from soft segmentation (initialized by location priors of each part)
Parts can be hard to find on their own

Which patch corresponds to a body part?
Which patch corresponds to a body part?
Sequential structured prediction

• Can consider pose estimation as predicting a set of related variables (called structured prediction)
  – Some parts easy to find (head), some are hard (wrists)

• One solution: jointly solve for most likely variables (DPM, pictorial structures)

• Another solution: iteratively predict each variable based in part on previous predictions
Pose machines

Image Location $z$

Image Features

$g_1$

Input Image

Local image evidence is weak
Certain parts are easier to detect than others

Ramakrishna et al. ECCV 2014
Example results
General principle

• “Auto-context” (Tu CVPR 2008): instead of fancy graphical models, create feature from past predictions and repredict

• Can view this as an “unrolled belief propagation” (Ross et al. 2011)
One more approach: parallel structured prediction

• Back to CNNs
  – CNN model is a sequence of iterative feature processing
  – Last feature layer stores features that encode key information for all predictions
  – In parallel, predict bounding boxes, category, parts, and keypoints from last feature layer
Mask R-CNN – He Gxioxari Dollar Girshick

• Same network as Faster R-CNN, except
  – Bilinearly interpolate when extracting 7x7 cells of ROI features for better alignment of features to image
  – Instance segmentation: produce a 28x28 mask for each object category
  – Keypoint prediction: produce a 56x56 mask for each keypoint (aim is to label single pixel as correct keypoint)
Top performing object detector, keypoint segmenter, instance segmenter

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>AP^bb</th>
<th>AP^bb_{50}</th>
<th>AP^bb_{75}</th>
<th>AP^bb_{S}</th>
<th>AP^bb_{M}</th>
<th>AP^bb_{L}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster R-CNN++ [19]</td>
<td>ResNet-101-C4</td>
<td>34.9</td>
<td>55.7</td>
<td>37.4</td>
<td>15.6</td>
<td>38.7</td>
<td>50.9</td>
</tr>
<tr>
<td>Faster R-CNN w FPN [27]</td>
<td>ResNet-101-FPN</td>
<td>36.2</td>
<td>59.1</td>
<td>39.0</td>
<td>18.2</td>
<td>39.0</td>
<td>48.2</td>
</tr>
<tr>
<td>Faster R-CNN by G-RMI [21]</td>
<td>Inception-ResNet-v2 [37]</td>
<td>34.7</td>
<td>55.5</td>
<td>36.7</td>
<td>13.5</td>
<td>38.1</td>
<td>52.0</td>
</tr>
<tr>
<td>Faster R-CNN w TDM [36]</td>
<td>Inception-ResNet-v2-TDM</td>
<td>36.8</td>
<td>57.7</td>
<td>39.2</td>
<td>16.2</td>
<td>39.8</td>
<td>52.1</td>
</tr>
<tr>
<td>Faster R-CNN, RoIAlign</td>
<td>ResNet-101-FPN</td>
<td>37.3</td>
<td>59.6</td>
<td>40.3</td>
<td>19.8</td>
<td>40.2</td>
<td>48.8</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td>ResNet-101-FPN</td>
<td>38.2</td>
<td>60.3</td>
<td>41.7</td>
<td>20.1</td>
<td>41.1</td>
<td>50.2</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td>ResNeXt-101-FPN</td>
<td>39.8</td>
<td>62.3</td>
<td>43.4</td>
<td>22.1</td>
<td>43.2</td>
<td>51.2</td>
</tr>
</tbody>
</table>

Table 3. Object detection single-model results (bounding box AP), vs. state-of-the-art on test-dev. Mask R-CNN uses

<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>

Table 1. Instance segmentation mask AP on COCO test-dev. MNC [10] and FCIS [26] are the winners of the COCO 2015 and 2016

<table>
<thead>
<tr>
<th>Method</th>
<th>AP^kp</th>
<th>AP^kp_{50}</th>
<th>AP^kp_{75}</th>
<th>AP^kp_{M}</th>
<th>AP^kp_{L}</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMU-Pose+++ [6]</td>
<td>61.8</td>
<td>84.9</td>
<td>67.5</td>
<td>57.1</td>
<td>68.2</td>
</tr>
<tr>
<td>G-RMI [31]†</td>
<td>62.4</td>
<td>84.0</td>
<td>68.5</td>
<td><strong>59.1</strong></td>
<td>68.1</td>
</tr>
<tr>
<td>Mask R-CNN, keypoint-only</td>
<td>62.7</td>
<td>87.0</td>
<td>68.4</td>
<td>57.4</td>
<td>71.1</td>
</tr>
<tr>
<td>Mask R-CNN, keypoint &amp; mask</td>
<td><strong>63.1</strong></td>
<td><strong>87.3</strong></td>
<td><strong>68.7</strong></td>
<td>57.8</td>
<td><strong>71.4</strong></td>
</tr>
</tbody>
</table>

Table 4. Keypoint detection AP on COCO test-dev. Ours
Example detections and instance segmentations
Example detections and instance segmentations
Example keypoint detections
What if you want to label every pixel?

“Stuff” can be hard to capture with bounding boxes

https://www.cityscapes-dataset.com/examples/#fine-annotations
Fully convolutional networks for semantic segmentation – Long Shelhamer Darrel 2015

- Use network trained for classification as pre-trained network for pixel labeling
- Convert fully connected layers into convolutions
- Add features from earlier conv layers to improve resolution
- Fine-tune for pixel labeling task
“Fully convolutional” results

• Takes advantage of pre-training from classification

• Applied to objects and scenes (NYUv2)

• But feature pooling reduces spatial sensitivity and resolution
Dilated Convolutions – Yu Kolton 2016

• Replacing last two pooling layers with “dilated convolution” that filters a sparse 3x3 grid of pixels

• Enables large receptive field with few parameters

• Improves resolution
Dilated Convolutions results

(a) Image  (b) Front end  (c) + Context  (d) + CRF-RNN  (e) Ground truth

<table>
<thead>
<tr>
<th></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>ankle</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>mean IoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepLab++</td>
<td>89.1</td>
<td>38.3</td>
<td>88.1</td>
<td>63.3</td>
<td>69.7</td>
<td>87.1</td>
<td>83.1</td>
<td>85</td>
<td>85</td>
<td>29.3</td>
<td>76.5</td>
<td>56.5</td>
<td>79.8</td>
<td>85.8</td>
<td>82.4</td>
<td>57.4</td>
<td>84.3</td>
<td>54.9</td>
<td>80.5</td>
<td>64.1</td>
</tr>
<tr>
<td>DeepLab-MSc++</td>
<td>89.2</td>
<td>46.7</td>
<td>88.5</td>
<td>63.5</td>
<td>68.4</td>
<td>87.0</td>
<td>81.2</td>
<td>86.3</td>
<td>32.6</td>
<td>80.7</td>
<td>62.4</td>
<td>81.0</td>
<td>81.3</td>
<td>84.3</td>
<td>82.1</td>
<td>56.2</td>
<td>84.6</td>
<td>58.3</td>
<td>76.2</td>
<td>67.2</td>
</tr>
<tr>
<td>CRF-RNN</td>
<td>90.4</td>
<td>55.3</td>
<td>88.7</td>
<td>68.4</td>
<td>69.8</td>
<td>88.3</td>
<td>82.4</td>
<td>85.1</td>
<td>32.6</td>
<td>78.5</td>
<td>64.4</td>
<td>79.6</td>
<td>81.9</td>
<td><strong>86.4</strong></td>
<td><strong>88.1</strong></td>
<td>82.4</td>
<td>58.6</td>
<td>82.4</td>
<td>53.5</td>
<td>77.4</td>
</tr>
<tr>
<td>Front end</td>
<td>86.6</td>
<td>37.3</td>
<td>84.9</td>
<td>62.4</td>
<td>67.3</td>
<td>86.2</td>
<td>81.2</td>
<td>82.1</td>
<td>32.6</td>
<td>77.4</td>
<td>58.3</td>
<td>75.9</td>
<td>81</td>
<td>83.6</td>
<td>82.3</td>
<td>54.2</td>
<td>81.5</td>
<td>50.1</td>
<td>77.5</td>
<td>63</td>
</tr>
<tr>
<td>Context</td>
<td>89.1</td>
<td>39.1</td>
<td>86.8</td>
<td>62.6</td>
<td>68.9</td>
<td>88.2</td>
<td>82.6</td>
<td>87.7</td>
<td>33.8</td>
<td>81.2</td>
<td>59.2</td>
<td>81.8</td>
<td>87.2</td>
<td>83.3</td>
<td>83.6</td>
<td>53.6</td>
<td>84.9</td>
<td>53.7</td>
<td>80.5</td>
<td>62.9</td>
</tr>
<tr>
<td>Context + CRF</td>
<td>91.3</td>
<td>39.9</td>
<td>88.9</td>
<td>64.3</td>
<td>69.8</td>
<td>88.9</td>
<td>82.6</td>
<td>89.7</td>
<td>34.7</td>
<td>82.7</td>
<td>59.5</td>
<td>83</td>
<td>88.4</td>
<td>84.2</td>
<td>85</td>
<td>55.3</td>
<td>86.7</td>
<td>54.4</td>
<td><strong>81.9</strong></td>
<td>63.6</td>
</tr>
<tr>
<td>Context + CRF-RNN</td>
<td><strong>91.7</strong></td>
<td>39.6</td>
<td>87.8</td>
<td>63.1</td>
<td><strong>71.8</strong></td>
<td>89.7</td>
<td><strong>89.8</strong></td>
<td><strong>89.8</strong></td>
<td>37.2</td>
<td>84</td>
<td>63</td>
<td><strong>83.3</strong></td>
<td><strong>89</strong></td>
<td><strong>85.1</strong></td>
<td>56</td>
<td>87.6</td>
<td>56</td>
<td>80.2</td>
<td>64.7</td>
<td>75.3</td>
</tr>
</tbody>
</table>
Graphical models vs. sequential/parallel prediction

• Advantages of BP/graphcut/etc
  – Elegant
  – Relations are explicitly modeled
  – Exact inference in some cases

• Advantages of sequential/parallel prediction
  – Simple procedures for training and inference
  – Learns how much to rely on each prediction
  – Can model very complex relations
Things to remember

• Models can be broken down into part appearance and spatial configuration
  – Wide variety of models

• Efficient optimization can be tricky but usually possible
  – Generalized distance transform is a useful trick

• Rather than explicitly modeling contextual relations, can encode through features/classifiers
Next classes

- Tues: Object tracking with Kalman Filters
- Thurs: Action Recognition