Object Detection

Shuran Song
Task: Generic object detection
Task: Generic object detection

In this lecture we focus on 2D image object detection but object detection can also be in 3D.
Outline

• Basic object detection algorithms
  – Rigid Template
  – Part Based detector
• Features
• Data set
• New techniques for object detection
Object detection algorithms
Outline

• Rigid Template
  – HOG for human detection [Dalal Triggs]
  – Exemplar SVM detector

• Part Based detector
  – Deformable Part Model
  – Poselets
  – 3D Cuboid detector
Histograms of oriented gradients for human detection

[DalalTriggs]
Human detection with HOG: Basic Steps

1. Map image to feature Space (HOG)
Human detection with HOG: Basic Steps

1. Map image to feature Space (HOG)
2. Training with positive and negative (linear SVM)

positive training examples

negative training examples
Human detection with HOG: Basic Steps

1. Map image to feature Space (HOG)
2. Training with positive and negative (linear SVM)
3. Testing : scan image in all scale and all location
   Binary classification on each location

Scale-space pyramid

Human detection with HOG: Basic Steps

1. Map image to feature Space (HOG)
2. Training with positive and negative (linear SVM)
3. Testing : scan image in all scale and all location
   Binary classification on each location
Problem:
Bounding box size is different for the same object (different depth)

Solution 1:
Resize the box and do multiple convolution?
Not ideal:
It will change the feature dimension, need to retrain the SVM for each scale.
Solution 2: Resize the image and do multiple convolution? → image pyramid

- Image is smaller ~ box is bigger
- Image is larger ~ box is smaller
Human detection with HOG: Basic Steps

1. Map image to feature Space (HOG)
2. Training with positive and negative (linear SVM)
3. Testing: scan image in all scale and all location
   Binary classification on each location
Human detection with HOG: Basic Steps

1. Map image to feature Space (HOG)
2. Training with positive and negative (linear SVM)
3. Testing : scan image in all scale and all location
4. Report box : non-maximum suppression
Summary of Basic object detection Steps

Training:
Train a classifier describe the detection target

Testing:
Detection by binary classification on all location
HOG descriptor
HOG: Gradients

- Compress image to 64x128 pixels
- Convolution with \([-1 0 1] \ [-1; 0; 1]\) filters
- Compute gradient magnitude + direction
- For each pixel: take the color channel with greatest magnitude as final gradient
HOG: Cell histograms

- Divide the image to cells, each cell 8x8 pixels
- Snap each pixel’s direction to one of 18 gradient orientations
- Build histogram pre-cell using magnitudes
Histogram interpolation example

- Interpolated trilinearly:
  - Bilinearly into spatial cells
  - Linearly into orientation bins
1. contrast sensitive features: 18 orientation -> 18 dim
2. contrast insensitive features: 9 orientation -> 9 dim
   Normalize 4 times by its neighbor blocks, and average them

3. texture features: sum of the magnitude over all orientation and normalize 4 time, not average -> 4 dim

In total each cell : 18+9+4 dimension of feature
Final Descriptor

- Concatenation the normalized histogram

Visualization:
HOG Descriptor:

1. **Compute gradients** on an image region of 64x128 pixels

2. **Compute histograms** on ‘cells’ of typically 8x8 pixels (i.e. 8x16 cells)

3. **Normalize histograms** within overlapping blocks of cells

4. **Concatenate histograms**

   It is a typical procedure of feature extraction!
Feature Engineering

• Developing a feature descriptor requires a lot of engineering
  – Testing of parameters (e.g. size of cells, blocks, number of cells in a block, size of overlap)
  – Normalization schemes

• An extensive evaluation was performed to make these design decisions

• It’s not only the idea, but also the engineering effort
Problem?

Single, rigid template usually not enough to represent a category.

• Many object categories look very different from different viewpoints, or style

• Many objects (e.g. humans) are articulated, or have parts that can vary in configuration
Solution:

- Exemplar SVM: Ensemble of Exemplar-SVMs for Object Detection and Beyond

- Part Based Model
Exemplar-SVM

• Still a rigid template, but train a separate SVM for each positive instance

For each category it can has exemplar with different size aspect ratio
Benefit from Exemplar–SVM?

- Handel the intra-category variance naturally, without using complicated model.
- Compare to nearest neighbor approach: make use of negative data and train a discriminative object detector
- Explicit correspondence from detection result to training exemplar
Benefit from Exemplar-SVM?

- Explicit correspondence from detection result to training exemplar

We not only know it is train, but also its orientation and type!
Benefit from Exemplar-SVM?

We can do even more.
Training Exemplar–SVM

Objective Function:

$$
\Omega_E(w, b) = \|w\|^2 + C_1 h(w^T x_E + b) + C_2 \sum_{x \in \mathcal{N}_E} h(-w^T x - b)
$$

- \( h(x) = \max(1-x, 0) \)
- Distance to the margin for the support vectors

\( h(x) = 0 \)
\( h(x) = 1-x \)
Training Exemplar–SVM

Objective Function:

$$\Omega_E(w, b) = ||w||^2 + C_1 h(w^T x_E + b) + C_2 \sum_{x \in N_E} h(-w^T x - b)$$

Exemplar represented by

$$\sim 100$$ HOG Cells
Training Exemplar–SVM

Objective Function:

\[ \Omega_E(w, b) = \|w\|^2 + C_1 h(w^T x_E + b) + C_2 \sum_{x \in \mathcal{N}_E} h(-w^T x - b) \]

\( x_E \)
Exemplar represented by 
~100 HOG Cells

\( \mathcal{N}_E \)
Windows from images not containing any in-class instances
Training Exemplar–SVM

Objective Function:

$$\Omega_E(w, b) = \|w\|^2 + C_1 h(w^T x_E + b) + C_2 \sum_{x \in \mathcal{N}_E} h(-w^T x - b)$$

Learn the $w$ that minimize the objective function, equivalent to maximize the margin
Hard Negative Mining

$$\Omega_E(w, b) = ||w||^2 + C_1 h(w^T x_E + b) + C_2 \sum_{x \in \mathcal{N}_E} h(-w^T x - b)$$

$$\mathcal{N}_E$$

Windows from images not containing any in-class instances: but there is too many!
2,000 images x 10,000 windows per image = 20M negatives

Find ones that you get wrong by a search, and train on these hard ones
Hard Negative Mining

*Input*: Positive: exemplar E  
Negative: images and bounding boxes for this category  
\[ N = \{(J_1, B_1), (J_2, B_2), \ldots (J_m, B_m)\} \]

*Initialize*: random pick \( m \) patches \( N_{\text{random}} \) from \( N \) that not overlap with  
\[ [SV, b, w] = \text{trainSVM}(E, N_{\text{random}}) \]

*Hard negative mining*

While: \( i \neq m \) or \( N_{\text{hard}} \) not empty  
for \( i = 1 \) to \( n \) do  
\[ D = \text{detect}(b, w, J_i) \]
\[ N_i = D.\text{conf} > \text{threshold} \& D \text{ not overlap with } B_i \]
Add \( N_i \) to \( N_{\text{hard}} \)
if \( |N_{\text{hard}}| > \text{memory-limit} \), then break;
end
\[ [SV_{\text{new}}, b_{\text{new}}, w_{\text{new}}] = \text{trainSVM}(E, [N_{\text{random}}, SV]) \]
\[ SV = [SV; SV_{\text{new}}] \]
end
Embarrassingly Parallel

- Each exemplar performs its own hard negative mining
- Solve many convex learning problems
- Parallel training on cluster
Calibration

• Idea: make different exemplar detector's score compatible.

• Good exemplar vs Bad exemplar
Calibration

- Idea: make different exemplar detector's score compatible.

1. On Validation set:
   Sort detection by score,
   classify true/false detection

2. sigmoid function

   Target Score

   Detection Score

   Detection Score

- All detector's scores are from 0 to 1.
- The calibrated score reflects each detector's precision
Calibration

- **Idea**: make different exemplar detector's score compatible.

Which one is “better” Exemplar?
### Performance

<table>
<thead>
<tr>
<th>Method</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traditional NN + Calibration</td>
<td>0.110</td>
</tr>
<tr>
<td>Local Distance Function + Calibration</td>
<td>0.157</td>
</tr>
<tr>
<td><strong>Exemplar-SVMs + Calibration</strong></td>
<td>0.198</td>
</tr>
<tr>
<td><strong>Exemplar-SVMs + Co-occurrence</strong></td>
<td>0.227</td>
</tr>
<tr>
<td>One SVM per category (Dalal and Triggs 2005)</td>
<td>0.097</td>
</tr>
<tr>
<td>Deformable Part Model (Felzenszwalb et al 2010)</td>
<td>0.266</td>
</tr>
</tbody>
</table>
## Performance

<table>
<thead>
<tr>
<th>Method</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traditional NN + Calibration</td>
<td>0.110</td>
</tr>
<tr>
<td>Local Distance Function + Calibration</td>
<td>0.157</td>
</tr>
<tr>
<td><strong>Exemplar-SVMs + Calibration</strong></td>
<td><strong>0.198</strong></td>
</tr>
<tr>
<td><strong>Exemplar-SVMs + Co-occurrence</strong></td>
<td><strong>0.227</strong></td>
</tr>
<tr>
<td>One SVM per category (Dalal and Triggs 2005)</td>
<td>0.097</td>
</tr>
<tr>
<td><strong>Deformable Part Model (Felzenszwalb et al 2010)</strong></td>
<td><strong>0.266</strong></td>
</tr>
</tbody>
</table>

We can do better
Part Based detector

- Pictorial Structures
- Without part label
  - Deformable part model
- With part labeled
  - Poselets
  - 3D Cuboid detector
Part Based detector

Objects are represented by features of parts and spatial relations between parts

Face model by Fischler and Elschlager '73
Part Based detector

• How to defined the parts for one object category
• How to represent their spatial relation shape
• How to combine parts detection and spatial relations to obtained the final detection
DPM : Object Detection with Discriminatively Trained Part Based Models

P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan,
Object Detection with Discriminatively Trained Part Based Models, PAMI 32(9), 2010
DPM: overview

• Each category detector has mixture of deformable part models (components)
• Each component has global template + deformable parts
• Fully trained from bounding boxes alone (Latent SVM)
DPM: component

- Each category detector has a mixture of components for different aspect ratio (handle intra-class variance)
- Each component has its own DPM model
DPM: component

- root filters
  - coarse resolution
- part filters
  - finer resolution
- deformation models

Each component has a root filter $F_0$ and $n$ part models $(F_i, v_i, d_i)$
DPM: Initialization

Root filter for each component

- For each component warp all positives to have same size
- Random pick negatives with same size
- Standard SVM no latent information
DPM: Initialization

Initializing Part Filter

- Fixed number: 6 parts per component
- Choose the high-energy regions of the root filter
  (Energy: norm of positive weight in subwindow)
- Greedy approach: once part placed set to zero and find next high-energy part
DPM: Training

• Training data consists of images with labeled bounding boxes.
• Need to learn the model structure, filters and deformation costs.
DPM: Latent SVM training

Classifiers that score an example $x$ using

$$f_\beta(x) = \max_{z \in Z(x)} \beta \cdot \Phi(x, z)$$

$\beta$ are model parameters
$z$ are latent values

Training data $D = (\langle x_1, y_1 \rangle, \ldots, \langle x_n, y_n \rangle)$ $y_i \in \{-1, 1\}$

We would like to find $\beta$ such that: $y_i f_\beta(x_i) > 0$

Minimize

$$L_D(\beta) = \frac{1}{2} \| \beta \|^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i f_\beta(x_i))$$
DPM: Latent SVM training

\[ L_D(\beta) = \frac{1}{2} ||\beta||^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i f_\beta(x_i)) \]

- Convex if we fix \( z \) for positive examples

Initialize \( \beta \) and iterate:
- Fix \( \beta \) and find the best \( z \) for each training example (detection)
- Fix \( z \) and solve for \( \beta \) (standard SVM training)
DPM: Detection

\[
\text{score}(p_0, \ldots, p_n) = \sum_{i=0}^{n} F_i \cdot \phi(H, p_i) \quad \text{“data term”} \quad \sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2) \quad \text{“spatial prior”}
\]

displacements

deflection parameters

Score for one part at certain location:
filter response score – deform cost relative to root
DPM: Detection

• Define an overall score for each root location
  – Based on best placement of parts
    \[
    \text{score}(p_0) = \max_{p_1, \ldots, p_n} \text{score}(p_0, \ldots, p_n).
    \]

• High scoring root locations define detections
• Efficient computation: dynamic programming + generalized distance transforms
DPM: Detection

Response of filter in l-th pyramid level

\[ R_l(x, y) = F \cdot \phi(H, (x, y, l)) \]
DPM: Detection

Transformed response: max-convolution

\[ D_l(x, y) = \max_{dx,dy} \left( R_l(x + dx, y + dy) - d_i \cdot (dx^2, dy^2) \right) \]

Response of filter in l-th pyramid level

Root location

cost - response
DPM: Detection

Transformed response: max-convolution

\[ D_t(x, y) = \max_{dx,dy} \left( R_t(x + dx, y + dy) - d_i \cdot (dx^2, dy^2) \right) \]

- For each root location its’ best placement of this part can be read from the transformed response directly.
Combine Many Parts

\[ P_{i,l}(x, y) = \arg\max_{d_x, d_y} (R_{k,l}(x + d_x, y + d_y) - d_i \cdot \phi_d(d_x, d_y)) \]
DPM: Detection

(after non-maximum suppression)

~1 second to search all scales
Car model

Component 1

Component 2
Car detections

high scoring true positives

high scoring false positives
More detections

- horse
- sofa
- bottle
DPM: Learning from weakly-labeled data
  Only bounding box no part label need

Can we do better with part labeled?
  – Articulated Human Detection with Flexible Mixtures-of-Parts
  – Cuboid detection
  – Poselet
Articulated Human Detection with Flexible Mixtures-of-Parts

Manually labeled part, and defined tree structure
Find part type (component) for each part

Clustering by part label’s location wrt it’s parent (Not by their aspect ratio)
Learning- Structural SVM

\[
\arg \min_{w, \xi_i \geq 0} \frac{1}{2} \beta \cdot \beta + C \sum_n \xi_n
\]

s.t. \quad \forall n \in \text{pos} \quad \beta \cdot \Phi(I_n, z_n) \geq 1 - \xi_n

\forall n \in \text{neg}, \forall z \quad \beta \cdot \Phi(I_n, z) \leq -1 + \xi_n

\beta = (w, b) \quad \text{Feature (HOG)}

Positive : Image with human, z pose – from label
Negative : Image without human, z pose - any pose

Detection and pose estimation at same time
Detection

Tree structure example: 4 parts head as root, neck is body’s parent
Results
3D Cuboid Detector

Input image

Output detections

detect

Synthesized New Views
Poselet
What is a Poselet?

Poselets capture part of the pose from a given viewpoint

[Bourdev & Malik, ICCV09]
Poselets

Examples may differ visually but have common semantics

[Bourdev & Malik, ICCV09]
Poselets

One poselet one classifier not a model for whole human body
How do we train a poselet?
Finding correspondences at training time

Given part of a human pose

How do we find a similar pose configuration in the training set?
Finding correspondences at training time

We use key points to annotate the joints, eyes, nose, etc. of people.
Finding correspondences at training time

Residual Error
Training poselet classifiers

1. Given a seed patch
2. Find the closest patch for every other person
3. Sort them by residual error
4. Threshold them
Training poselet classifiers

1. Given a seed patch
2. Find the closest patch for every other person
3. Sort them by residual error
4. Threshold them
5. Use them as positive training examples to train a linear SVM with HOG features

One poselet one classifier not a model for whole human body
Test time
Goal: Extract attributes of this person

Input:
- Target person bounds
- Bounds of other nearby people
Step 1: Detect poselet activations

[Bourdev et al, ECCV10]
Step 2: Cluster the activations

Because we know the joint for each poselet
Step 3: Predict person bounds

[Bourdev et al, ECCV10]
Step 4: Identify the correct cluster

Max-flow in bipartite graph
Poselet Activations

Start with its poselet activations
Beyond detection: Attribute Classification

Person-level Attribute Classifiers

Poselet-level Attribute Classifiers

Features

Poselet Activations
Features
Local & Global Features

A set of **local features** describes image properties at one particular location in the image:

A set of **global features** provides information about the global image structure without encoding specific objects.

This feature likes images with vertical structures at the top part and horizontal texture at the bottom part (this is a typical composition of an empty street).
Image Descriptors

• HOG
• Gist scene descriptor
• LBP (Local Binary Pattern)
• SIFT (Scale-Invariant Feature Transform)
GIST:

Example visual gists

Oliva & Torralba (2001)
Certain local binary patterns are fundamental properties of texture, providing the vast majority, sometimes over 90%.
They have one thing in common: contains very few spatial transitions.
Uniformity: number of spatial transitions (bitwise 0/1 changes).

Ojala, Pietikäinen, Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. PAMI’02.
SIFT: Scale-Invariant Feature Transform

Images convolved with Gaussian of different scales

Scale (next octave)

Scale (first octave)

Gaussian

Difference of Gaussian (DOG)
Key Point Localization

Detect maxima and minima of difference-of Gaussian in scale space

For each maxima and minima we know their 2D location and scale
Select Canonical Orientation

- Create histogram of local gradient directions computed at selected scale
- Assign canonical orientation at peak of smoothed histogram
- Each key specifies stable 2D coordinates (x, y, scale, orientation)
SIFT: Scale-Invariant Feature Transform

- Compute image gradients in the chosen region
- Weight the gradient magnitude by a Gaussian
- Create array of orientation histograms
- 8 orientations x 4x4 histogram array = 128 dimensions

Implementation Details: http://www.vlfeat.org/api/sift.html#sift-intro-detector

Color-SIFT

Experiment 1: Descriptor performance on image benchmark

MAP

0.0 0.1 0.2 0.3 0.4 0.5
HOG: Histogram of oriented gradients

- Compute gradient using filter: [1; 0; +1]
- Discretize gradient orientation: contrast insensitive (9bin) & sensitive (18bin)
- Aggregate into cells with “soft binning”
- Normalize & Truncate: min(v,0.2)

- Dimension reduction

PCA of HOG features

HOG: Histogram of oriented gradients

- Compute gradient using filter: [1; 0; +1]
- Discretize gradient orientation: contrast insensitive (9bin) & sensitive (18bin)
- Aggregate into cells with “soft binning”
- Normalize & Truncate: min(v,0.2)

- Dimension reduction
- 31dimension feature vector = 9 (insensitive) +18 (sensitive) + 4
- 124dimension feature vector = concatenation of 2x2 neighboring feature vectors together

New techniques for object detection

- Finding things
- Selective search
- RCNN
Finding Things: Image Parsing with Regions and Per-Exemplar Detectors
Finding Things: Image Parsing with Regions and Per-Exemplar Detectors

Generation of our detector based data term.
Selective Search for Object Recognition

Like segmentation, we use the image structure to guide our sampling process.
Like exhaustive search, we aim to capture all possible object locations.
Deep learning in object detection

1. Input image
2. Extract region proposals (≈2k)
3. Compute CNN features
4. Classify regions
Training RCNN

1. Pre-train CNN for **image classification**
   - large auxiliary dataset (ImageNet)
   - train CNN

2. Fine-tune CNN on **target dataset** and **task**
   - fine-tune CNN
   - small target dataset (PASCAL VOC)

3. Train linear predictor for **detection**
   - region proposals
   - small target dataset (PASCAL VOC)
   - ~2000 warped windows / image
   - CNN features
   - training labels
   - per class SVM
## Result

<table>
<thead>
<tr>
<th>Method</th>
<th>VOC 2007</th>
<th>VOC 2010</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM v5 (Girshick et al. 2011)</td>
<td>33.7%</td>
<td>29.6%</td>
</tr>
<tr>
<td>UVA sel. search (Uijlings et al. 2012)</td>
<td></td>
<td>35.1%</td>
</tr>
<tr>
<td>Regionlets (Wang et al. 2013)</td>
<td>41.7%</td>
<td>39.7%</td>
</tr>
<tr>
<td><strong>Fine-tuned</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R-CNN pool$_5$</td>
<td>40.1%</td>
<td></td>
</tr>
<tr>
<td>R-CNN fc$_6$</td>
<td>43.4%</td>
<td></td>
</tr>
<tr>
<td>R-CNN fc$_7$</td>
<td>42.6%</td>
<td></td>
</tr>
<tr>
<td>R-CNN FT pool$_5$</td>
<td>42.1%</td>
<td></td>
</tr>
<tr>
<td>R-CNN FT fc$_6$</td>
<td>47.2%</td>
<td></td>
</tr>
<tr>
<td>R-CNN FT fc$_7$</td>
<td><strong>48%</strong></td>
<td><strong>43.5%</strong></td>
</tr>
</tbody>
</table>
Next generation of object detection

Observation:
sliding windows -> segmentation
designed features -> deep learned features
parts based models -> linear SVMs
Data Set

Data set is critical for the advance of computer vision.

Many algorithm’s success are based on good data set.
Both quantity and quality are important for a dataset
Data Set

The Ultimate Dataset
✓ All scene categories
✓ Large collections of realistic photos
✓ Label all objects

- Frame-object Snapshot
- Multi-object Scene
Data Set: PASCAL  2005-2012

20 classes

• Bounding box annotation
• Object in scene: realistic photos
• 20 class
• If there is not PASCAL, there may not be DPM and all the later object detection algorithm
Data Set

Not large enough?

IMAGENET

SUN
imageNet

- 20,000+ categories
- 14,000,000+ images
Scene UNderstanding Database

- 131,072 images
- 3,819 object categories
- 249,522 segmented objects
Chairs

Object centric dataset

Scene centric dataset

Typical Chairs from Image-net

Typical Chairs from SUN Database

• Location Bias
• Size Bias
• Number Bias
• Context Bias

• Occlusion Bias
• Background Bias
• Lighting Bias
• Viewpoint Bias
Object-centric vs. Scene-centric

Image-Net

SUN
Object without Scene

Object without scene really sucks!

Toilet without Restroom
• Good object detection algorithm need to consider
• What feature?
  – HOG, SIFT ...
• What model?
  – Rigid, DPM...
• What data?
  – PASCAL, imageNet, SUN...