Visual Recognition: Objects, Actions and Scenes

Lecture 5:
Category-level object localization

Ivan Laptev
ivan.laptev@inria.fr

Slides from Andrew Zisserman
Visual Recognition and Machine Learning Summer School, 2010-2013

Includes slides from: Ondra Chum, Alyosha Efros, Mark Everingham, Pedro Felzenszwalb, Rob Fergus, Kristen Grauman, Bastian Leibe, Fei-Fei Li, Marcin Marszalek, Pietro Perona, Deva Ramanan, Bernt Schiele, Jamie Shotton, Andrea Vedaldi
What we would like to do...

- Visual scene understanding
- **What** is in the image and **where**

- Object categories, identities, properties, activities, relations, …
Recognition Tasks

• **Image Classification**
  – Does the image contain an aeroplane? (last lecture)

• **Object Class Detection/Localization**
  – Where are the aeroplanes (if any)?

• **Object Class Segmentation**
  – Which pixels are part of an aeroplane (if any)?
**Things vs. Stuff**

*Thing (n)*: An object with a specific size and shape.

*Stuff (n)*: Material defined by a homogeneous or repetitive pattern of fine-scale properties, but has no specific or distinctive spatial extent or shape.

Ted Adelson, Forsyth et al. 1996.

Slide: Geremy Heitz
Recognition Task

• **Object Class Detection/Localization**
  – Where are the aeroplanes (if any)?

• **Challenges**
  – Imaging factors e.g. lighting, pose, occlusion, clutter
  – Intra-class variation

• **Compared to Classification**
  – Detailed prediction e.g. bounding box
  – Location usually provided for training
Challenges: Scale
Challenges: Background Clutter
Challenges: Occlusion and truncation
Challenges: Intra-class variation
Object detection: naïve attempt

Find the chair in this image

Output of normalized correlation

This is a chair

Slide credit: A. Torralba
Object detection: naïve attempt

Find the chair in this image

Pretty much garbage
Simple template matching is not going to make it

Slide credit: A. Torralba
Object Category Recognition by Learning

- Difficult to define model of a category. Instead, learn from example images
Level of Supervision for Learning

Image-level label

Bounding box

Pixel-level segmentation

“Parts”
Preview of typical results

- aeroplane
- bicycle
- car
- cow
- horse
- motorbike
Detection by Classification

• Basic component: binary classifier
Detection by Classification

- Detect objects in clutter by **search**

- **Sliding window**: exhaustive search over position and scale
Detection by Classification

- Detect objects in clutter by **search**

- **Sliding window**: exhaustive search over position and scale
Detection by Classification

- Detect objects in clutter by **search**

- **Sliding window**: exhaustive search over position and scale (can use same size window over a spatial pyramid of images)
Window (Image) Classification

- Features usually engineered
- Classifier learnt from data

Training Data

Car/Non-car

\[ P(c|x) \propto F(x) \]
Problems with sliding windows ...

- aspect ratio
- granularity (finite grid)
- partial occlusion
- multiple responses

See work by

- Christoph Lampert et al CVPR 08, ECCV 08
Objective: detect (localize) standing humans in an image

sliding window classifier

train a binary classifier on whether a window contains a standing person or not

Histogram of Oriented Gradients (HOG) feature

although HOG + SVM originally introduced for pedestrians has been used very successfully for many object categories
Feature: Histogram of Oriented Gradients (HOG)

- tile 64 x 128 pixel window into 8 x 8 pixel cells
- each cell represented by histogram over 8 orientation bins (i.e. angles in range 0-180 degrees)
Histogram of Oriented Gradients (HOG) continued

- Adds a second level of overlapping spatial bins re-normalizing orientation histograms over a larger spatial area

- Feature vector dimension (approx) = 16 x 8 (for tiling) x 8 (orientations) x 4 (for blocks) = 4096
Window (Image) Classification

- HOG Features
- Linear SVM classifier

Training Data

Feature Extraction

Classifier

\[ F(x) \]

pedestrian/Non-pedestrian

\[ P(c|x) \propto F(x) \]
Averaged examples
Advantages of linear SVM:

\[ f(x) = w^T x + b \]

- **Training (Learning)**
  - Very efficient packages for the linear case, e.g. LIBLINEAR for batch training and Pegasos for on-line training.
  - Complexity \( O(N) \) for \( N \) training points (cf \( O(N^3) \) for general SVM)

- **Testing (Detection)**

  - Non-linear
    \[ f(x) = \sum_{i=1}^{S} a_i k(x_i, x) + b \]
    \[ S = \text{# of support vectors} \]
    \[ = (\text{worst case}) N \]
    \[ \text{size of training data} \]

  - Linear
    \[ f(x) = \sum_{i=1}^{S} w_i x_i^T x + b \]
    \[ = w^T x + b \]
    Independent of size of training data
Review: Binary classification

Given training data \((x_i, y_i)\) for \(i = 1 \ldots N\), with \(x_i \in \mathbb{R}^d\) and \(y_i \in \{-1, 1\}\), learn a classifier \(f(x)\) such that

\[
f(x_i) \begin{cases} 
\geq 0 & y_i = +1 \\
< 0 & y_i = -1 
\end{cases}
\]

i.e. \(y_i f(x_i) > 0\) for a correct classification.
Review: Linear classifiers

A linear classifier has the form

$$f(x) = \mathbf{w}^\top \mathbf{x} + b$$

- in 2D the discriminant is a line
- \( \mathbf{w} \) is the normal to the plane, and \( b \) the bias
- \( \mathbf{w} \) is known as the weight vector
Review: Linear classifiers

A linear classifier has the form

$$f(x) = w^\top x + b$$

- In 3D, the discriminant is a plane, and in nD, it is a hyperplane.
Review: Linear classifiers

- Find linear function (*hyperplane*) to separate positive and negative examples

\[ x_i \text{ positive} : \quad x_i \cdot w + b \geq 0 \]
\[ x_i \text{ negative} : \quad x_i \cdot w + b < 0 \]

Which hyperplane is best?
Review: Linear classifiers - margin

- Generalization is not good in this case:

- Better if a margin is introduced:
Support vector machines

• Find a hyperplane that maximizes the margin between positive and negative examples

\[ x_i \text{ positive (} y_i = 1) : \quad x_i \cdot w + b \geq 1 \]
\[ x_i \text{ negative (} y_i = -1) : \quad x_i \cdot w + b \leq -1 \]

For support, vectors, \( x_i \cdot w + b = \pm 1 \)

The margin is \( \frac{2}{\|w\|} \)

• For more details on SVM see e.g. tutorial by C.H. Lampert
  https://sites.google.com/site/christophlampert/files/Lampert-Tutorial-Kernel_Methods-DAGM.pdf?attredirects=0
Training: “Jittering” of positive samples

- Jitter annotation to increase the set of positive trainingsamples
Test: Non-maximum suppression (NMS)

- Scanning-window detectors typically result in multiple responses for the same object

To remove multiple responses, a simple greedy procedure called “Non-maximum suppression” is applied:

NMS:  
1. Sort all detections by detector confidence  
2. Choose most confident detection $d_i$; remove all $d_j$ s.t. $overlap(d_i,d_j) > T$  
3. Repeat Step 2. until convergence
Learned model

\[ f(x) = w^T x + b \]
What do negative weights mean?

\[ wx > 0 \]

\[ (w^+ - w^-)x > 0 \]

\[ w^+ > w^-x \]

Complete system should compete pedestrian/pillar/doorway models

Discriminative models come equipped with own bg

(avoid firing on doorways by penalizing vertical edges)
Why does HOG + SVM work so well?

• Similar to SIFT, records spatial arrangement of histogram orientations
• Compare to learning only edges:
  – Complex junctions can be represented
  – Avoids problem of early thresholding
  – Represents also soft internal gradients
• Older methods based on edges have become largely obsolete

• HOG gives fixed length vector for window, suitable for feature vector for SVM
Chamfer Matching

- Match points between template and image
- Measure mean distance
- Template edgel matches nearest image edgel

\[ D(T, I) = \frac{1}{|T|} \sum_{p \in T} \min_{q \in I} d(p, q) \]

- Distance transform reduces min operation to array lookup
- Computable in linear time
- Localize by sliding window search

[Gavrila & Philomin, 1999]
Chamfer Matching

• In practice performs poorly in clutter
• Unoriented edges are not discriminative enough (too easy to find…)

[Gavrila & Philomin, 1999]
Training a sliding window detector

• Object detection is inherently asymmetric: much more “non-object” than “object” data

• Classifier needs to have very low false positive rate
• Non-object category is very complex – need lots of data
Bootstrapping

1. Pick negative training set at random
2. Train classifier
3. Run on training data
4. Add false positives to training set
5. Repeat from 2

• Collect a finite but diverse set of non-object windows
• Force classifier to concentrate on hard negative examples
• For some classifiers can ensure equivalence to training on entire data set
Example: train an upper body detector

- **Training data** – used for training and validation sets
  - 33 Hollywood2 training movies
  - 1122 frames with upper bodies marked

- **First stage training (bootstrapping)**
  - 1607 upper body annotations jittered to 32k positive samples
  - 55k negatives sampled from the same set of frames

- **Second stage training (retraining)**
  - 150k hard negatives found in the training data

---

![Example images](image_url)
Training data – positive annotations
Positive windows

Note: common size and alignment
Jittered positives
Jittered positives
Random negatives
Random negatives
Window (Image) first stage classification

- Jittered positives
- Random negatives

HOG Feature Extraction

\[ f(x) = w^T x + b \]

- Linear SVM Classifier

- Find high scoring false positives detections
- These are the hard negatives for the next round of training
- Cost = \# training images x inference on each image
Hard negatives
Hard negatives
First stage performance on validation set
Precision – Recall curve

- **Precision**: % of returned windows that are correct
- **Recall**: % of correct windows that are returned

![Diagram showing precision-recall curve with Venn diagrams and classifier score decreasing feature]
First stage performance on validation set
Performance after retraining
Effects of retraining
Side by side

before retraining

after retraining
Side by side

before retraining

after retraining
Side by side

before retraining

after retraining
Tracked upper body detections
Tracked upper body detections
Tracked upper body person detections

Combined face, upper body and full body detectors “vote” for upper body bounding boxes. Detections are tracked and smoothed over video.
[Lezama, MVA thesis 2010]
Notes

• Training (bootstrapping, retraining) done in a more principled way in Structured Output learning with the cutting plane algorithm

• An object detector can be learnt from one positive example
  
  – See Exemplar SVM by Malisiewicz, Gupta, Efros, ICCV 2011
Accelerating Sliding Window Search

• Sliding window search is slow because so many windows are needed e.g. \( x \times y \times \text{scale} \approx 100,000 \) for a 320\( \times \)240 image

• Most windows are clearly not the object class of interest

• Can we speed up the search?
Cascaded Classification

• Build a sequence of classifiers with increasing complexity

More complex, slower, lower false positive rate

• Reject easy non-objects using simpler and faster classifiers
Cascaded Classification

- Slow expensive classifiers only applied to a few windows ➔ significant speed-up

- Controlling classifier complexity/speed:
  - Number of support vectors [Romdhani et al, 2001]
  - Number of features [Viola & Jones, 2001]
  - Type of SVM kernel [Vedaldi et al, 2009]
Summary: Sliding Window Detection

• Can convert any image classifier into an object detector by sliding window. Efficient search methods available.

• Requirements for invariance are reduced by searching over e.g. translation and scale.

• HOG+SVM is an efficient object detector with good properties.

• “Hard negative” mining can cope with huge numbers of negative windows.

\[ f(x) = w^T x + b \]
Last lecture: Sliding Window Detection

- Can convert any image classifier into an object detector by sliding window. Efficient search methods available.

- Requirements for invariance are reduced by searching over e.g. translation and scale.

- HOG+SVM is an efficient object detector with good properties.

But...

HOG+SVM essentially does template matching, it cannot model non-rigid object deformations
Class of model: Pictorial Structure

• Intuitive model of an object
• Model has two components
  1. parts (2D image fragments)
  2. structure (configuration of parts)
• Dates back to Fischler & Elschlager 1973

Is this complexity of representation necessary?
Which features?
Restrict deformations
Problem of background clutter

- Use a sub-window
  - At correct position, no clutter is present
  - Slide window to detect object
  - Change size of window to search over scale
Recap: Bag of (visual) Words representation

• Detect affine invariant local features (e.g. affine-Harris)

• Represent by high-dimensional descriptors, e.g. 128-D for SIFT

• How to summarize sliding window content in a fixed-length vector for classification?

1. Map descriptors onto a common vocabulary of **visual words**

2. Represent image as a histogram over visual words – a **bag of words**
Local region descriptors and visual words

- Normalize regions to fixed size and shape
- Describe each region by a SIFT descriptor
- Vector quantize into visual words, e.g. using k-means

NB: aff. detectors/SIFT/visual words originally for view point invariant matching
Visual Words

Cluster = Visual Word

Local Descriptors

Vector Quantize (K-means)
Example Visual Words
Intuition

- Visual words represent “iconic” image fragments
- Feature detectors and SIFT give invariance to local rotation and scale
- Discarding spatial information gives configuration invariance
Learning from positive ROI examples

Bag of Words

Feature Vector
Sliding window detector

- Classifier: SVM with linear kernel
- BOW representation for ROI

Example detections for dog

Lampert et al CVPR 08
Discussion: ROI as a Bag of Visual Words

• Advantages
  – No explicit modelling of spatial information => high level of invariance to position and orientation in image
  – Fixed length vector => standard machine learning methods applicable

• Disadvantages
  – No explicit modelling of spatial information => less discriminative power
  – Inferior to state of the art performance
Beyond BOW I: Pictorial Structure

- Intuitive model of an object
- Model has two components
  1. parts (2D image fragments)
  2. structure (configuration of parts)
- Dates back to Fischler & Elschlager 1973
Object Detection with Discriminatively Trained Part Based Models

Pedro F. Felzenszwalb, David Mcallester, Deva Ramanan, Ross Girshick
PAMI 2010

Matlab code available online:
http://www.cs.brown.edu/~pff/latent/
Approach

- Mixture of deformable part-based models
  - One component per “aspect” e.g. front/side view
- Each component has global template + deformable parts
- Discriminative training from bounding boxes alone
Example Model

• One component of person model
Starting Point: HOG Filter

- Search: sliding window over position and scale
- Feature extraction: HOG Descriptor
- Classifier: Linear SVM

Filter $F$

Score of $F$ at position $p$ is $F \cdot \varphi(p, H)$

$\varphi(p, H) =$ concatenation of HOG features from subwindow specified by $p$

Dalal & Triggs [2005]
Object Hypothesis

- Position of root + each part
- Each part: HOG filter (at higher resolution)

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

\(p_0\) : location of root
\(p_1, \ldots, p_n\) : location of parts

Score is sum of filter scores minus deformation costs
Score of a Hypothesis

\[
\text{score}(p_0, \ldots, p_n) = \sum_{i=0}^{n} F_i \cdot \phi(H, p_i) - \sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2)
\]

Appearance term

Spatial prior

- Linear classifier applied to feature subset defined by hypothesis

- Concatenation of filters and deformation parameters

- Concatenation of HOG features and part displacement features
Part Detection

Response of filter in l-th pyramid level

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

cross-correlation

Transformed response

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

max-convolution, computed in linear time (spreading, local max, etc)
System

feature map

feature map at twice the resolution

response of root filter

response of part filters

transformed responses

color encoding of filter response values

combined score of root locations
Training

• Training data = images + bounding boxes
• Need to learn: model structure, filters, deformation costs
Latent SVM (MI-SVM)

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

SVM objective

“Hinge loss” on one training example

• Which component?
• Where are the parts?
Latent SVM Training

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

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

- Optimization:
  - Initialize \( \beta \) and iterate:
    - Pick best \( z \) for each positive example
    - Optimize \( \beta \) with \( z \) fixed

- Local minimum: needs good initialization
  - Parts initialized heuristically from root

Alternation strategy
Person Model

- Root filters
- Coarse resolution
- Part filters
- Finer resolution
- Deformation models

Handles partial occlusion/truncation
Car Model

root filters
coarse resolution

part filters
finer resolution

deformation
models
Car Detections

- high scoring true positives
- high scoring false positives
Person Detections

high scoring true positives

high scoring false positives (not enough overlap)
Precision/Recall: VOC2008 Person

- UoCTTIUCI (42.0)
- LEAR_PlusClass (19.7)
- CASIA_Det (11.2)
- XRCE_Det (9.0)
- MPI_struct (2.5)
- Jena (2.0)
Precision/Recall: VOC2008 Bicycle

- UoCTTIUCI (42.0)
- LEAR_PlusClass (34.3)
- Oxford (24.6)
- CASIA_Det (14.6)
- XRCE_Det (10.5)
- MPI_struct (8.0)
- Jena (1.4)
Comparison of Models
Segmentation as Selective Search for Object Recognition

K. van de Sande\textsuperscript{1}, J. Uijlings\textsuperscript{2}, T. Gevers\textsuperscript{1}, and A. Smeulders\textsuperscript{1}
University of Amsterdam\textsuperscript{1} and University of Trento\textsuperscript{2}
ICCV 2011

Slides by Esa Rahtu

(material taken from van de Sande’s ICCV paper and PASCAL presentations)
Main objectives

- Object recognition in ”large” datasets (VOC and ImageNet)
  - Object classification (what is it?)
  - Localization (where is it?)
  - Use computationally ”demanding” models
Motivation

- Most current approaches use exhaustive search
  - Visit every location in an image
  - Imposes computational constraints on
    - Number of possible locations -> grid/fixed aspect ratio
    - Evaluation cost per location -> simple features/classifiers
  - To go beyond this, we need something more sophisticated

Viola IJCV 2004
Dalal CVPR 2005
Felzenszwalb TPAMI 2010
Vedaldi ICCV 2009
Selective search

- Instead of using grid, "intelligently" choose locations where classifier is evaluated.

-> Selective search

Others:
Rahtu ICCV 2011
Alexe TPAMI 2012

Carreira CVPR 2010
Endres ECCV 2010

Vedaldi ICCV 2009
Alexe CVPR 2010
Main design criteria

• **High recall**
  – We do not want to lose any objects, since they cannot be recovered later.

• **Coarse locations are sufficient**
  – Accurate delineation is not necessary for recognition
  – In contrary, nearby context might be useful
    -> use bounding boxes

• **Fast to compute**
  – Necessary when operating with large datasets
    -> <10s/image
How to obtain high recall?

• Images are intrinsically hierarchical

• Segmentation at single scale are not enough
    -> hypotheses based on hierarchical grouping
Proposed method

• Start by oversegmenting the input image

“Efficient graph-based image segmentation”
Felzenszwalb and Huttenlocher, IJCV 2004
Proposed method

• Compute similarity measure between all adjacent region pairs \( a \) and \( b \) as

\[
S(a, b) = S_{size}(a, b) + S_{texture}(a, b)
\]

- **Proportion of the image area that \( a \) and \( b \) jointly occupy**
- **Histogram intersection of 8-bin gradient direction histogram computed in each color channel**

- Encourages small regions to merge early and prevents single region from gobbling up all others one by one.

- Encourages regions with similar texture (and color) to be grouped early.
Proposed method

1. Merge two most similar regions based on $S$.
2. Update similarities between the new region and its neighbors.
3. Go back to step 1. until the whole image is a single region.
Proposed method

- Take bounding boxes of all generated regions and treat them as possible object locations.
Proposed method
High recall revisited

- No single segmentation works for all cases
  -> diversify the set of segmentations
- Use different color spaces
  - RGB, Opponent color, normalized RGB, and hue
- Use different parameters in Felzenswalb method
  - $k = [100, 150, 200, 250]$ ($k =$ threshold parameter)
Evaluation of object hypotheses

• Recall is a proportion of objects that are covered by some box with >0.5 overlap
Comparison with other methods

**Experiment 2: Maximum Recall of Selective Search for Recognition**

<table>
<thead>
<tr>
<th>Method</th>
<th>Max. recall (%)</th>
<th># windows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sliding Windows [13]</td>
<td>83.0</td>
<td>200 per class</td>
</tr>
<tr>
<td>Jumping Windows [27]</td>
<td>94.0</td>
<td>10,000 per class</td>
</tr>
<tr>
<td>‘Objectness’ [1]</td>
<td>82.4</td>
<td>10,000</td>
</tr>
<tr>
<td>Our hypotheses</td>
<td>96.7</td>
<td>1,536</td>
</tr>
</tbody>
</table>

**Experiment 2: Recall of Selective Search for Recognition**

![Graph showing recall vs. number of candidate windows for different methods including Sliding Windows, Jumping Windows, 'Objectness', and Our hypotheses.]
Comparison with other methods

- Segmentation accuracy:
  (although this was not the main design criteria)

<table>
<thead>
<tr>
<th></th>
<th>Max. recall (%)</th>
<th># windows</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carreira [3]</td>
<td>78.2</td>
<td>697</td>
<td>432</td>
</tr>
<tr>
<td>Endres [7]</td>
<td>82.2</td>
<td>1,989</td>
<td>226</td>
</tr>
<tr>
<td>Our hypotheses</td>
<td>79.8</td>
<td>1,973</td>
<td>8</td>
</tr>
<tr>
<td>Combination</td>
<td>90.1</td>
<td>4,659</td>
<td>666</td>
</tr>
</tbody>
</table>
Comparison continued

![Graphs showing recall vs. overlap score threshold for different methods.](image)
Object localization (training)

- Use original and mirrored ground truth as positives
- Use object hypotheses to create hard negatives
- Add two iterations of false positives

- Features: Bag-of-words using SIFT, Opponent-SIFT, RGB-SIFT, calculated at every pixel at one scale. Codebook of size 4096 and spatial pyramid with 4 levels.
- Classification: SVM with histogram intersection kernel (fast approximation)

S. Maji, A. C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In CVPR, 2008. 4
Training and testing

- Most telling window
Object localization (testing)

- Apply learned model to all object hypotheses
- Sort windows based on classification score
- Remove windows which have more than 30% overlap with a higher scoring window (i.e. NMS)

### Experiment 4: Object Recognition Accuracy on VOC2010 Test Set

<table>
<thead>
<tr>
<th>System</th>
<th>plane</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>dog</th>
<th>horse</th>
<th>motor</th>
<th>person</th>
<th>plant</th>
<th>sheep</th>
<th>sofa</th>
<th>train</th>
<th>tv</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUS</td>
<td>.491</td>
<td>.524</td>
<td>.178</td>
<td>.120</td>
<td>.306</td>
<td>.535</td>
<td>.328</td>
<td>.373</td>
<td>.177</td>
<td>.306</td>
<td>.277</td>
<td>.295</td>
<td>.519</td>
<td>.563</td>
<td>.442</td>
<td>.096</td>
<td>.148</td>
<td>.279</td>
<td>.495</td>
</tr>
<tr>
<td>This paper</td>
<td><strong>.582</strong></td>
<td>.419</td>
<td>.<strong>192</strong></td>
<td>.140</td>
<td>.143</td>
<td>.448</td>
<td>.367</td>
<td>.488</td>
<td>.129</td>
<td>.281</td>
<td><strong>.287</strong></td>
<td><strong>.394</strong></td>
<td>.441</td>
<td>.525</td>
<td>.258</td>
<td><strong>.141</strong></td>
<td><strong>.388</strong></td>
<td><strong>.342</strong></td>
<td>.431</td>
</tr>
</tbody>
</table>

- Improve state-of-the-art for 8 categories.
Object localization (testing)

- Comparison with Felzenswalb part-based model:
  - Improves with deformable categories (cat, cow, sheep,..)
  - Works less well with categories with rigid shape (car,..)
Conclusions (selective search)

• Proposed selective search strategy:
  – High recall: >96% with approx. 1500 proposals
  – Coarse locations, which seem to be sufficient
  – Fast computation time: <10s per image
  – Class independent
  – Enables to use bag-of-fetaures models
  – State-of-the art in object detection
  – Shows that conceptually simple Bag-of-features models outperforms more sophisticated deformable part models (DPM)
References


References (cont.)