Class 4: Part-Based and Hierarchical Models

Rogerio Feris, Feb 14, 2013
EECS 6890 – Topics in Information Processing
Spring 2013, Columbia University
http://rogerioferis.com/VisualRecognitionAndSearch
Reminder: Project Proposal Presentation

Required Content:

- Summarize the problem and main idea of the project
- Do a literature search and briefly describe related work
- Describe your technical approach – what algorithms you plan to explore? Which programming language you plan to utilize?
- Describe your plan for experiments and quantitative evaluation. What dataset you plan to use?
- How the work will be split among team members?
- Provide a list of tasks and an estimated time for completing each task. We understand the schedule may change as the project is developed.

See more details at:
Project proposal Reminder

Important Dates:

- **Tuesday, February 19 (11:59pm)** – deadline for sending the project proposal presentation slides.

- **Thursday, February 21** – project proposal class presentation (10 minutes per group)

- Project Report coming soon!

Contact us for feedback!
What we have seen so far

Low-Level Representation: Feature Detection and Description

Classical Descriptors: SIFT and SURF

Modern Descriptors for Real-Time Applications: FAST, BRISK, ...

Check the resources page for state-of-the-art implementations: http://rogerioferis.com/VisualRecognitionAndSearch/Resources.html
What we have seen so far

Mid-level Representation: feature coding and pooling

Classical Bag-of-Word Models

Modern higher-level representations:
Fisher vector and super-vector encoding, Sparse Coding, ...

[K. Chatfield et al, The devil is in the details: an evaluation of recent feature encoding methods, BMVC 2011]
Pooling is great to achieve invariance to image transformations, more compact representations, and better robustness to noise and clutter.

But important spatial relationship of image patches may be lost due to pooling.

Example: bag-of-words
Encoding structure

Bag of words

Deformable Part Models [P. Felzenszwalb et al, PAMI 2010]

Structureless

Rigid

Dalal and Triggs, CVPR 2005
Plan for Today

- Part-based Models
  - Focus on Deformable Part Models
- Connections to Hierarchical Models
  - Brief Intro to Convolutional Nets
Class Notation

- Useful tool (source code)

- Important Comment

- Pros and Cons

- Break
History of Part and Structure Approaches

- Fischler & Elschlager 1973
- Yuille ‘91
- Brunelli & Poggio ‘93
- Lades, v.d. Malsburg et al. ‘93
- Cootes, Lanitis, Taylor et al. ‘95
- Amit & Geman ‘95, ‘99
- Felzenszwalb & Huttenlocher ’00, ’04, ’08
- Crandall & Huttenlocher ’05, ’06
- Leibe & Schiele ’03, ’04
- Many papers since 2000

Slide credit: Rob Fergus
Why Parts?

Useful to handle intra-class variation

Objects may be globally different, but have parts in common

Slide credit: Rob Fergus
Why Parts?

Useful to handle slight variations in object pose

[Heisele et al, CVPR 2001]
Why Parts?

Useful to handle occlusions

[P. Felzenszwalb et al, PAMI 2010]
Why Parts?

- Difficult to handle low-resolution
- Models can be more complex and more computationally expensive
- May throw away important image information present in global representations
Part-based Models

Different Connectivity Structures

- **a) Constellation**
  - Fergus et al. ‘03
  - Fei-Fei et al. ‘03
  - Csurka ‘04
  - Vasconcelos ‘00

- **b) Star shape**
  - Crandall et al. ‘05
  - Fergus et al. ‘05

- **c) k-fan (k = 2)**
  - Crandall et al. ‘05

- **d) Tree**
  - Felzenszwalb & Huttenlocher ‘00

- **e) Bag of features**
  - Bouchard & Triggs ‘05

- **f) Hierarchy**
  - Carneiro & Lowe ‘06

From [Carneiro & Lowe, ECCV 2006]
Part-based Models

Different Connectivity Structures

Constellation Model [Fergus et al, 2003]

Efficient Pictorial Structures  [Felzenszwalb & Huttenlocher, 2000]
Part-based Models

Different Connectivity Structures

Implicit Shape Model [Leibe et al, 2004]

Poselets [Bourdev et al, 2009]
Object Detection with Deformable Part-based Models (DPM) [P. Felzenszwalb et al, PAMI 2010]

Pay attention and think what makes DPM work so well (discussion later)

PASCAL VOC "Lifetime Achievement" Prize in 2010.
Object Detection: Learning Stage

Positive Samples

Negative Samples

Learning

Object Model

Root

Parts

Deformation
Object Detection: Test Time

Click for sliding window animation

- Sliding Window Approach
  - Model is applied at every position/scale of the image to check the presence of the object
Deformable Part-Based Models

Key Ingredients:

- Powerful HOG (Histograms of Oriented Gradients) features
- Detection based on Deformable Parts
- Latent SVM Training
- Multiple Components
HOG Features

Histograms of Oriented Gradients (HOG)

- Bin gradients into 9 orientations over 8x8 pixel neighborhoods & normalize
  - Dalal & Triggs CVPR05

Slide credit: Deva Ramanan
HOG Features

Histograms of Oriented Gradients (HOG)

- Split detection window into non-overlapping pixel regions called cells (e.g., 8x8 pixels)
- Compute histogram of oriented gradients in each cell (e.g., 9 orientation bins)
- Group cells into larger spatial blocks and contrast-normalize each block separately
- Final feature descriptor is the concatenation of histograms for all overlapping blocks over the detection window
HOG Features

Cell = 8x8 pixels
Histograms of gradients = 9 orientation bins
Block = 2x2 cells

9x4 = 36 dimensions per cell
HOG Features

- Feature pooling (histograms) inside cells improves robustness against small object deformations.

- Fine scale derivatives (no smoothing) and fine orientation binning (e.g., 9 bins) improve performance.

- Local contrast normalization is essential for good performance (for better invariance to illumination, shadowing, ...).

- The DPM implementation uses a slightly different HOG representation than [Dalal&Triggs 2005]: both signed and unsigned gradients are taken into account and dimensionality reduction is performed.
HOG Features

VLFeat HOG Implementation
http://www.vlfeat.org/overview/hog.html

cellSize = 8;
hog = vl_hog(im, cellSize, 'verbose');

- More parameters available (orientation binning, ...)
- Both Dalal&Triggs and UoCTTI (DPM) versions available
**Filters** are rectangular templates defining *weights* for features.

Score is dot product of filter and subwindow of HOG pyramid.

\[
\text{Score of } F \text{ at position } p \text{ is } \ F \cdot \phi(p, H)
\]

\[
\phi(p, H) = \text{HOG features in subwindow specified by } p
\]

Slide credit: Pedro Felzenszwalb
HOG Filters

- [Dalal and Triggs, C VPR 2005]: HOG filter is learned using a linear SVM
- At test time, filter is convolved with feature map

Slide credit: Lana Lazebnik
Pedestrian Detection Demo

Dalal and Triggs, CVPR 2005
Deformable Part-based Models

Each model has a global template + part templates. Models are fully trained from bounding boxes alone (without specification of part locations)
Scoring an Object Hypothesis

- Pick a hypothesis $Z$ (location of root + location of parts)

$$z = (p_0, ..., p_n)$$

- $p_0$: location of root
- $p_1, ..., p_n$: location of parts

Score is sum of filter scores minus deformation costs

Multi-Scale Model: resolution of the part filters is twice the resolution of the root

Slide credit: Pedro Felzenszwalb
Scoring an Object Hypothesis

The score of a hypothesis is the sum of filter scores minus the sum of deformation costs.

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

Hypothesis Z: root location + part locations
Filters (root + parts)
Part Deformation weights
Subwindow features
Displacements with respect to part anchor

For a single hypothesis:

\[
\text{score}(z) = \beta \cdot \Psi(H, z)
\]

Concatenation of filter and deformation weights
Concatenation of subwindow features and displacements

Visual Recognition And Search  
Columbia University, Spring 2013
Detection

- Sliding window approach

- For each window location in the pyramid, compute score based on best placement of parts:

\[
score(p_0) = \max_{p_1, \ldots, p_n} score(p_0, \ldots, p_n)
\]

- High scoring root locations define detections

- *Generalized distance transforms* are used to compute the best placement of parts as a function of root location (i.e., the computation of the score above)
Detection

Example: best-scoring displacements for the “head” part

Input image
I-th pyramid level

head filter

Response of filter in I-th pyramid level:

\[ R_I(x, y) = F \cdot \phi(H, (x, y, I)) \]

Cross-correlation

Transformed Response:

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

“Default” anchor
head location

- Computed in linear time (distance transform)
- Smart Blur according to part deformation
Matching Result

(after non-maximum suppression)

~1 second to search all scales on a multi-core computer
Break
Latent SVM Learning

Positive Samples

Negative Samples

Learning

Object Model

Root  Parts  Deformation
Our classifier has the form:

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

\( \beta \) 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

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

- Maximum of convex functions is convex
- \( f_\beta(x) = \max_{z \in Z(x)} \beta \cdot \Phi(x, z) \) is convex in \( \beta \)
- \( \max(0, 1 - y_i f_\beta(x_i)) \) is convex for negative examples

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

Convex if latent values for positive examples are fixed
Latent SVM Training:

Initialize weights (beta) and iterate:

1) **Relabel positive examples**: fix beta and pick best z (latent variables) for each positive example

2) **Optimize Beta**: fix z and optimize beta (convex optimization problem)

Hard example mining during training is essential for improved performance
Multiple Components

- Multiple models (components) are utilized to deal with significant appearance variations that cannot be tackled with deformable parts (e.g., frontal view and side-view).

- Component label is added as a latent variable (in addition to part locations).

Car model:
Latent SVM Training: Initialization

Initializing Beta (concatenation of filter and deformation weights):

- For a k-component model:
  - Split examples into k sets based on bounding box aspect ratio
- Learn k root filters using standard SVMs
- Initialize parts by selecting patches from root filters:
  - Sub-windows with strong coefficients
  - Interpolate to get higher resolution filters
  - Initialize spatial model using fixed spring constants
Cat Detections

high scoring true positives

high scoring false positives (not enough overlap)
Person Detections

high scoring true positives

high scoring false positives (not enough overlap)
Car Detections

high scoring true positives

high scoring false positives
More Example Results

horse

sofa

bottle
Comparison of Car Models

class: car, year 2006

- 1 Root (0.48)
- 2 Root (0.58)
- 1 Root+Parts (0.55)
- 2 Root+Parts (0.62)
- 2 Root+Parts+BB (0.64)
Cascade DPM Detection

- Goal: Improve efficiency of DPM

- Hierarchy of models where the i-th model is defined by the first i parts from the original model

- Intuition: when detecting people, we might evaluate the score of the head part at each possible location and decide we do not need to evaluate the torso part for most locations in the image

Positions (in white) where a particular part appearance model was evaluated at the scale of the bicycle detection. The images are shown in “cascade order” from left to right and top to bottom.
Sparselets

- Goal: Improve efficiency of DPM for large number of classes
- GPU Implementation + Sparse coding of Part Filters

See Video Demo at http://www.eecs.berkeley.edu/~song/sparselets/
http://people.cs.uchicago.edu/~rbg/latent/

Stable Release Version (Includes Cascade Detection)

**Discriminatively trained deformable part models**

Version 5 (Sept. 5, 2012)
Discussion

Why DPM works?
What are the main limitations?

Check paper: How important are Deformable Parts in the Deformable Parts Model? [Divvala et al, Parts and Attributes Workshop, 2012]
Towards Richer Hierarchical Models

- Hierarchical Models – parts defined as a function of subparts
- Stochastic Image Grammars

S.C. Zhu et al. and D. Mumford
Towards Richer Hierarchical Models

- Hierarchical Models – parts defined as a function of subparts

Brief Intro to Convolutional Networks

More on large-scale classification class!
Multi-layer Perceptrons

Weights are learned with the error backpropagation algorithm
Fully connected network may lead to billions of parameters
Convolutional Neural Networks

- Locally connected neural network (each neuron has a receptive field)
- Weights are shared across locations (convolution)
- Subsampling (pooling Layer) for dealing with image transformations

Slide Credit: Ranzato
Convolutional Neural Networks

Multiple feature maps are learned (multiple convolutions with different learned filters)
Convolutional Neural Networks

Input

Feature Map 1

Slide Credit: Rob Fergus
Convolutional Neural Networks

Input → Feature Map 2

Slide Credit: Rob Fergus
Convolutional Networks

- Multiple Convolution and pooling layers define the network

- Backpropagation with constrained weights
Naïve implementation of ConvNets requires lots of training examples to work well.

Recent Advances:

1) **rectification of non-linearities and local contrast normalization** (see [Jarrett et al, What is the Best Multi-Stage Architecture for Object Recognition?, ICCV 2009]). Impressive results even with random filters!

2) **Unsupervised Pre-Training** [Geoffrey Hinton et al, 2006]
Are Deformable Part Models Related to Convolutional Networks?

- Gradient Operators (convolution)
- histogram of oriented gradients (pooling)
- Convolution with part filters
- Transformed response (pooling)
- Deformation Learning in DPM

Input Image

Output Labels

Hard-wired filters in DPM
Convolutional Networks

Torch7 (http://www.torch.ch/)

EBLearn (http://eblearn.cs.nyu.edu:21991/doku.php)
Convolutional Networks

- Imagenet Breakthrough, and excellent results on other datasets as well (MNIST, NORB, ...)

- Doesn’t seem to produce state-of-the-art results on Pascal VOC

- Unsupervised feature learning

- Difficult to set and tune parameters of the network