Sliding window detection

January 29, 2009

Kristen Grauman
UT-Austin

Schedule


Plan for today

- Lecture
  - Sliding window detection
  - Contrast-based representations
  - Face and pedestrian detection via sliding window classification
- Papers: HoG and Viola-Jones
- Demo
  - Viola-Jones detection algorithm

Tasks

- Detection: Find an object (or instance of object category) in the image.
- Recognition: Name the particular object (or category) for a given image/subimage.

- How is the object (class) going to be modeled or learned?
- Given a new image, how to make a decision?
Earlier: Knowledge-rich models for objects


Earlier: Knowledge-rich models for objects

Later: Statistical models of appearance

- Objects as appearance patches
  - E.g., a list of pixel intensities
- Learning patterns directly from image features

Eigenfaces (Turk & Pentland, 1991)
For what kinds of recognition tasks is a holistic description of appearance suitable?

Appearance-based descriptions
- Appropriate for classes with more rigid structure, and when good training examples available
Appearance-based descriptions

Scene recognition based on global texture pattern. [Oliva & Torralba (2001)]

What if the object of interest may be embedded in “clutter”? 
Sliding window object detection

If object may be in a cluttered scene, slide a window around looking for it.

Car/non-car Classifier

No, not a car.
Detection via classification

• Consider all subwindows in an image
  – Sample at multiple scales and positions

• Make a decision per window:
  – “Does this contain object category X or not?”

Detection via classification

Fleshing out this pipeline a bit more, we need to:
1. Obtain training data
2. Define features
3. Define classifier

Training examples

Feature extraction

Car/non-car Classifier
Detector evaluation

How to evaluate a detector?

When do we have a **correct** detection?

Is this correct?

\[
\frac{\text{Area intersection}}{\text{Area union}} > 0.5
\]

Summarize results with an **ROC curve**: show how the number of correctly classified positive examples varies relative to the number of incorrectly classified negative examples.

* Image: gim.unmc.edu/dxtests/ROC3.htm
Feature extraction:
global appearance

Simple holistic descriptions of image content
- grayscale / color histogram
- vector of pixel intensities

Eigenfaces:
global appearance description

An early appearance-based approach to face recognition

- Generate low-dimensional representation of appearance with a linear subspace.
- Project new images to “face space”.
- Recognition via nearest neighbors in face space.

Turk & Pentland, 1991
Feature extraction: global appearance

- Pixel-based representations sensitive to small shifts

- Color or grayscale-based appearance description can be sensitive to illumination and intra-class appearance variation

Cartoon example: an albino koala

Gradient-based representations

- Consider edges, contours, and (oriented) intensity gradients
Gradient-based representations

- Consider edges, contours, and (oriented) intensity gradients

- Summarize local distribution of gradients with histogram
  - Locally orderless: offers invariance to small shifts and rotations
  - Contrast-normalization: try to correct for variable illumination

Gradient-based representations: Histograms of oriented gradients

Map each grid cell in the input window to a histogram counting the gradients per orientation.
Gradient-based representations: SIFT descriptor

Lowe, ICCV 1999

Gradient-based representations: biologically inspired features

Convolve with Gabor filters at multiple orientations
Pool nearby units (max)
Intermediate layers compare input to prototype patches

Serre, Wolf, Poggio, CVPR 2005
Mutch & Lowe, CVPR 2006
Gradient-based representations: Rectangular features

Compute differences between sums of pixels in rectangles
Captures contrast in adjacent spatial regions
Similar to Haar wavelets, efficient to compute

Viola & Jones, CVPR 2001

Gradient-based representations: shape context descriptor

Count the number of points inside each bin, e.g.:

Count = 4

Count = 10

Log-polar binning: more precision for nearby points, more flexibility for farther points.

Local descriptor

Belongie, Malik & Puzicha, ICCV 2001
Classifier construction

- How to compute a decision for each subwindow?

Discriminative vs. generative models

- Generative: separately model class-conditional and prior densities
- Discriminative: directly model posterior

Plots from Antonio Torralba 2007

K. Grauman, B. Leibe
Discriminative vs. generative models

- **Generative:**
  - + possibly interpretable
  - + can draw samples
  - - models variability unimportant to classification task
  - - often hard to build good model with few parameters

- **Discriminative:**
  - + appealing when infeasible to model data itself
  - + excel in practice
  - - often can’t provide uncertainty in predictions
  - - non-interpretable

Discriminative methods

- **Nearest neighbor**
  - Shakhnarovich, Viola, Darrell 2003
  - Berg, Berg, Malik 2005...

- **Support Vector Machines**
  - Guyon, Vapnik
  - Heisele, Serre, Poggio, 2001...

- **Neural networks**
  - LeCun, Bottou, Bengio, Haffner 1998
  - Rowley, Baluja, Kanade 1998...

- **Boosting**
  - Viola, Jones 2001,
  - Torralba et al. 2004,
  - Opelt et al. 2006...

- **Conditional Random Fields**
  - McCallum, Freitag, Pereira 2000; Kumar, Hebert 2003...

*Slide adapted from Antonio Torralba*
Boosting

- Build a strong classifier by combining number of “weak classifiers”, which need only be better than chance
- Sequential learning process: at each iteration, add a weak classifier
- Flexible to choice of weak learner
  - including fast simple classifiers that alone may be inaccurate
- We’ll look at Freund & Schapire’s AdaBoost algorithm
  - Easy to implement
  - Base learning algorithm for Viola-Jones face detector

AdaBoost: Intuition

Consider a 2-d feature space with positive and negative examples.

Each weak classifier splits the training examples with at least 50% accuracy.

Examples misclassified by a previous weak learner are given more emphasis at future rounds.
AdaBoost: Intuition

Final classifier is combination of the weak classifiers.
AdaBoost Algorithm

Start with uniform weights on training examples

For T rounds

- Evaluate weighted error for each feature, pick best.

Re-weight the examples:
- Incorrectly classified ⇒ more weight
- Correctly classified ⇒ less weight

Final classifier is combination of the weak ones, weighted according to the error they had.

[Freund & Schapire 1995]

Example: Face detection

- Frontal faces are a good example of a class where global appearance models + a sliding window detection approach fit well:
  - Regular 2D structure
  - Center of face almost shaped like a “patch”/window
Feature extraction

“Rectangular” filters

Feature output is difference between adjacent regions

Efficiently computable with integral image: any sum can be computed in constant time

Avoid scaling images \(\rightarrow\) scale features directly for same cost

Value at \((x,y)\) is sum of pixels above and to the left of \((x,y)\)

\[
D = 1 + (2 + 3) = 1 + (A + B + C + D) - (A + C + A + B)
\]

Large library of filters

Considering all possible filter parameters: position, scale, and type:

- 180,000+ possible features associated with each 24 x 24 window

Use AdaBoost both to select the informative features and to form the classifier
AdaBoost for Efficient Feature Selection

- Image features = weak classifiers
- For each round of boosting:
  - Evaluate each rectangle filter on each example
  - Sort examples by filter values
  - Select best threshold for each filter (min error)
    - Sorted list can be quickly scanned for the optimal threshold
  - Select best filter/threshold combination
  - Weight on this features is a simple function of error rate
  - Reweight examples


AdaBoost for feature+classifier selection

- Want to select the single rectangle feature and threshold that best separates positive (faces) and negative (non-faces) training examples, in terms of weighted error.

Resulting weak classifier:

$$ h_i(x) = \begin{cases} +1 & \text{if } f_i(x) > \theta_i \\ -1 & \text{otherwise} \end{cases} $$

For next round, reweight the examples according to errors, choose another filter/threshold combo.

Viola & Jones, CVPR 2001
Cascading classifiers for detection

For efficiency, apply less accurate but faster classifiers first to immediately discard windows that clearly appear to be negative; e.g.,

- Filter for promising regions with an initial inexpensive classifier
- Build a chain of classifiers, choosing cheap ones with low false negative rates early in the chain

Fleuret & Geman, IJCV 2001
Rowley et al., PAMI 1998
Viola & Jones, CVPR 2001

Viola 2003

Slide credit: Paul Viola
Cascading classifiers for detection

- A 1 feature classifier achieves 100% detection rate and about 50% false positive rate.
- A 5 feature classifier achieves 100% detection rate and 40% false positive rate (20% cumulative)
  - using data from previous stage.
- A 20 feature classifier achieve 100% detection rate with 10% false positive rate (2% cumulative)

Viola-Jones Face Detector: Summary

- Train with 5K positives, 350M negatives
- Real-time detector using 38 layer cascade
- 6061 features in final layer
- [Implementation available in OpenCV: http://www.intel.com/technology/computing/opencv/]

Slide credit: Paul Viola

Viola 2003
Viola-Jones Face Detector: Results

First two features selected

K. Grauman, B. Leibe
Viola-Jones Face Detector: Results

K. Grauman, B. Leibe
Profile Features

Detecting profile faces requires training separate detector with profile examples.

Viola-Jones Face Detector: Results

Postprocess: suppress non-maxima
Example application

Frontal faces detected and then tracked, character names inferred with alignment of script and subtitles.

Everingham, M., Sivic, J. and Zisserman, A.
"Hello! My name is... Buffy" - Automatic naming of characters in TV video, BMVC 2006.
http://www.robots.ox.ac.uk/~vgg/research/nface/index.html

Fast face detection: Viola & Jones

Key points:
• Huge library of features
• Integral image – efficiently computed
• AdaBoost to find best combo of features
• Cascade architecture for fast detection
Discriminative methods

- Nearest neighbor
  - Shakhnarovich, Viola, Darrell 2003
  - Berg, Berg, Malik 2005...

- Neural networks
  - LeCun, Bottou, Bengio, Haffner 1998
  - Rowley, Baluja, Kanade 1998
  - ...

- Support Vector Machines
  - Guyon, Vapnik, Heisele, Serre, Poggio, 2001, ...

- Boosting
  - Viola, Jones 2001,
  - Torralba et al. 2004,
  - Opelt et al. 2006, ...

- Conditional Random Fields
  - McCallum, Freitag, Pereira 2000; Kumar, Hebert 2003
  - ...

Slide adapted from Antonio Torralba

---

Linear classifiers

- Find linear function to separate positive and negative examples

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

\[
x_i \text{ negative: } x_i \cdot w + b < 0
\]
Support Vector Machines (SVMs)

- Discriminative classifier based on optimal separating hyperplane
- Maximize the margin between the positive and negative training examples

Support vector machines

- Want line that maximizes the margin.

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

Support vector machines

- Want line that maximizes the margin.

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

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

Distance between point and line:
\[ \frac{|x_i \cdot w + b|}{\|w\|} \]

For support vectors:
\[ \frac{w^T x + b}{\|w\|} = \frac{\pm 1}{\|w\|} \]
\[ M = \frac{1}{\|w\|} - \frac{-1}{\|w\|} = \frac{2}{\|w\|} \]

Therefore, the margin is \( 2 / \|w\| \)
Finding the maximum margin line

1. Maximize margin $2/\|\mathbf{w}\|$
2. Correctly classify all training data points:
   - $\mathbf{x}_i$ positive ($y_i = 1$): $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$
   - $\mathbf{x}_i$ negative ($y_i = -1$): $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

**Quadratic optimization problem:**

Minimize $\frac{1}{2} \mathbf{w}^T \mathbf{w}$
Subject to $y_i (\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1$

• Solution: $\mathbf{w} = \sum \alpha_i y_i \mathbf{x}_i$

Finding the maximum margin line

- Solution: \( w = \sum \alpha_i y_i x_i \)
  \[ b = y_i - w \cdot x_i \] (for any support vector)
  \[ w \cdot x + b = \sum \alpha_i y_i x_i \cdot x + b \]
- Classification function:
  \[ f(x) = \text{sign}(w \cdot x + b) \]
  If \( f(x) < 0 \), classify as negative,
  if \( f(x) > 0 \), classify as positive
- Notice that it relies on an \textit{inner product} between the test point \( x \) and the support vectors \( x_i \)
- (Solving the optimization problem also involves computing the inner products \( x_i \cdot x_j \) between all pairs of training points)

C. Burges, \textit{A Tutorial on Support Vector Machines for Pattern Recognition}, Data Mining and Knowledge Discovery.

---

Non-linear SVMs

- Datasets that are linearly separable with some noise work out great:

- But what are we going to do if the dataset is just too hard?

- How about... mapping data to a higher-dimensional space:

Slide from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html
Non-linear SVMs: Feature spaces

- General idea: the original input space can be mapped to some higher-dimensional feature space where the training set is separable:

\[ \Phi: x \rightarrow \varphi(x) \]

Slide from Andrew Moore’s tutorial: http://www.autonlab.org/tutorials/svm.html

Nonlinear SVMs

- The kernel trick: instead of explicitly computing the lifting transformation \( \varphi(x) \), define a kernel function \( K \) such that

\[ K(x_i, x_j) = \varphi(x_i) \cdot \varphi(x_j) \]

- This gives a nonlinear decision boundary in the original feature space:

\[ \sum_i \alpha_i y_i K(x_i, x) + b \]

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998
Examples of General Purpose Kernel Functions

- Linear: \( K(x_i, x_j) = x_i^T x_j \)
- Polynomial of power \( p \): \( K(x_i, x_j) = (1 + x_i^T x_j)^p \)
- Gaussian (radial-basis function network):
  \[
  K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)
  \]

More on specialized image kernels -- next class.

Slide from Andrew Moore’s tutorial: http://www.autonlab.org/tutorials/svm.html

SVMs for recognition

1. Define your representation for each example.
2. Select a kernel function.
3. Compute pairwise kernel values between labeled examples.
4. Given this “kernel matrix” to SVM optimization software to identify support vectors & weights.
5. To classify a new example: compute kernel values between new input and support vectors, apply weights, check sign of output.
Pedestrian detection

- Detecting upright, walking humans also possible using sliding window’s appearance/texture; e.g.,

  SVM with Haar wavelets [Papageorgiou & Poggio, IJCV 2000]

  SVM with HoGs [Dalal & Triggs, CVPR 2005]

Pedestrian detection

- Navneet Dalal, Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005
Moving pedestrians

• What about video? Is pedestrian motion a useful feature?

Detecting Pedestrians Using Patterns of Motion and Appearance, P. Viola, M. Jones, and D. Snow, ICCV 2003.
  – Use motion and appearance to detect pedestrians
  – Generalize rectangle features for sequence data
  – Training examples = pairs of images.
Global appearance, windowed detectors:
The good things

- Some classes well-captured by 2d appearance pattern
- Simple detection protocol to implement
- Good feature choices critical
- Past successes for certain classes

Limitations

- High computational complexity
  - For example: 250,000 locations x 30 orientations x 4 scales = 30,000,000 evaluations!
  - With so many windows, false positive rate better be low
- If training binary detectors independently, means cost increases linearly with number of classes
Limitations (continued)

• Not all objects are “box” shaped

Limitations (continued)

• Non-rigid, deformable objects not captured well with representations assuming a fixed 2d structure; or must assume fixed viewpoint

• Objects with less-regular textures not captured well with holistic appearance-based descriptions
Limitations (continued)

- If considering windows in isolation, context is lost
- In practice, often entails large, cropped training set (expensive)
- Requiring good match to a global appearance description can lead to sensitivity to partial occlusions
Tools: A simple object detector with Boosting

Download

• Toolbox for manipulating dataset
• Code and dataset

Matlab code

• Gentle boosting
• Object detector using a part based model

Dataset with cars and computer monitors

http://people.csail.mit.edu/torralba/iccv2005/

From: Antonio Torralba

Tools: OpenCV

Face Detection using OpenCV

How to compile and run the facedetector is one of the frequently asked question in the OpenCV Yahoo! Groups. I’ll try to explain it to my best. Feel free to edit it if you have some more details.

Haar Like Features: What is that?

A recognition process can be much more efficient if it is based on the detection of features that encode some information about the class to be detected. This is the case of Haar-like features that encode the existence of oriented contrasts between regions in the image. A set of these features can be used to encode the contrasts exhibited by a human face and their spatial relationships. Haar-like features are so called because they are computed similar to the coefficients in Haar wavelet transforms.

The object detector of OpenCV has been initially proposed by Paul Viola and improved by Rainer Lienhart. First, a classifier (namely a cascade of boosted classifiers working with haar-like features) is trained with a few hundreds of sample views of a particular object (i.e., a face or a car), called positive examples, that are scaled to the same size (say, 20x20), and negative examples - arbitrary images of the same size.

• http://pr.willowgarage.com/wiki/OpenCV
Tools: LibSVM

- C++, Java
- Matlab interface