Computer Vision - Lecture 10

Sliding-Window based Object Detection

2.12.2010

Bastian Leibe
RWTH Aachen
http://www.mmp.rwth-aachen.de

leibe@umic.rwth-aachen.de
Announcements

• Exercise sheet 4 is already available
  - Subspace methods for object recognition [last Thu’s topic]
  - Sliding-window object detection [today’s topic]
  - The exercise will be next Thursday.
  ⇒ Submit your results by Tuesday night.

• Application: Face detection & identification

B. Leibe
Announcements

- An updated version of last week’s slides is available
  - Made notation more consistent
Course Outline

• Image Processing Basics
• Segmentation & Grouping
• Recognition
  ➢ Global Representations
  ➢ Subspace Representations
• Object Categorization I
  ➢ Sliding Window based Object Detection
• Local Features & Matching
• Object Categorization II
  ➢ Part based Approaches
• 3D Reconstruction
• Motion and Tracking
Recap: Subspace Methods

Subspace methods

Reconstructive
PCA, ICA, NMF

= +a1 +a2 +a3 + ...
representation

Discriminative
FLD, SVM, CCA

classification
regression

Slide credit: Ales Leonardis
Recap: Obj. Detection by Distance TO Eigenspace

- For each test image, compute the reprojection error
  - An \( n \)-pixel image \( x \in \mathbb{R}^n \) can be projected to the low-dimensional feature space \( y \in \mathbb{R}^m \) by
    \[
y = Ux
    \]
  - From \( y \in \mathbb{R}^m \), the reconstruction of the point is \( U^Ty \)
  - The error of the reconstruction is
    \[
    \|x - U^TUx\|
    \]
- Accept a detection if this error is low.
  - Assumption: subspace is optimized to the target object (class).
  - Other classes are not represented well \( \Rightarrow \) large error.

Slide credit: Peter Belhumeur
Recap: Obj Identification by Distance IN Eigenspace

- Objects are represented as coordinates in an \( n \)-dim. eigenspace.
- Example:
  - 3D space with points representing individual objects or a manifold representing parametric eigenspace (e.g., orientation, pose, illumination).
- Estimate parameters by finding the NN in the eigenspace

Slide adapted from Ales Leonardis
Recap: Eigenfaces
Recap: Restrictions of PCA

- PCA minimizes projection error

- PCA is "unsupervised" no information on classes is used
- Discriminating information might be lost
Recap: Linear Discriminant Analysis (FLD)

- Maximize distance between classes
- Minimize distance within a class

Criterion: \[ J(w) = \frac{w^T S_B w}{w^T S_W w} \]

\( S_B \) ... between-class scatter matrix
\( S_W \) ... within-class scatter matrix

The optimal solution for \( w \) can be obtained as:

\[ w \propto S_W^{-1}(m_2 - m_1) \]

Classification function:

\[ y(x) = w^T x + w_0 \]

\( \begin{array}{c} \geq 0 \text{ Class 1} \\ < 0 \text{ Class 2} \end{array} \)
Recap: Multiple Discriminant Analysis

- Generalization to $K$ classes

$$J(W) = \frac{|W^T S_B W|}{|W^T S_W W|}$$

- Solution given by generalized eigenvalue problem

$$S_B w_i = \lambda_i S_W w_i$$

- Defining $v = S_{\frac{1}{2}} B w$, we get

$$S_{\frac{1}{2}} S_W^{-1} S_{\frac{1}{2}} v = \lambda v$$

which is a regular eigenvalue problem.

⇒ Solve to get eigenvectors of $v$, then from that of $w$.

- For the $K$-class case we obtain (at most) $K-1$ projections.
  - (i.e. eigenvectors corresponding to non-zero eigenvalues.)
Recap: Fisherfaces

- Example Fisherface for recognition “Glasses/NoGlasses“
Topics of This Lecture

- **Object Categorization**
  - Problem Definition
  - Challenges

- **Sliding-Window based Object Detection**
  - Detection via Classification
  - Global Representations
  - Classifier Construction

- **Classification with Boosting**
  - AdaBoost
  - Viola-Jones Face Detection

- **Classification with SVMs**
  - Support Vector Machines
  - HOG Detector
Identification vs. Categorization
Identification vs. Categorization

- Find *this particular* object
- Recognize ANY car
- Recognize ANY cow
Object Categorization - Potential Applications

There is a wide range of applications, including:

- Autonomous robots
- Navigation, driver safety
- Consumer electronics
- Content-based retrieval and analysis for images and videos
- Medical image analysis
Object Categorization

- Task Description
  - “Given a small number of training images of a category, recognize a-priori unknown instances of that category and assign the correct category label.”

- Which categories are feasible visually?
  - Extensively studied in Cognitive Psychology, e.g. [Brown’58]
Visual Object Categories

• Basic Level Categories in human categorization
  [Rosch 76, Lakoff 87]
  - The highest level at which category members have similar perceived shape
  - The highest level at which a single mental image reflects the entire category
  - The level at which human subjects are usually fastest at identifying category members
  - The first level named and understood by children
  - The highest level at which a person uses similar motor actions for interaction with category members
Visual Object Categories

- Basic-level categories in humans seem to be defined predominantly visually.
- There is evidence that humans (usually) start with basic-level categorization before doing identification.

⇒ Basic-level categorization is easier and faster for humans than object identification!
⇒ Most promising starting point for visual classification
How many object categories are there?

Source: Fei-Fei Li, Rob Fergus, Antonio Torralba.

Biederman 1987
Other Types of Categories

- Functional Categories
  - e.g. chairs = “something you can sit on”
Challenges: Robustness

- Illumination
- Object pose
- Clutter
- Occlusions
- Intra-class appearance
- Viewpoint

Slide credit: Kristen Grauman
Challenges: Robustness

- Detection in crowded, real-world scenes
  - Learn object variability
  - Changes in appearance, scale, and articulation
  - Compensate for clutter, overlap, and occlusion

B. Leibe

[Leibe, Seemann, Schiele, CVPR’05]
Topics of This Lecture

- Object Categorization
  - Problem Definition
  - Challenges

- Sliding-Window based Object Detection
  - Detection via Classification
  - Global Representations
  - Classifier Construction

- Classification with Boosting
  - AdaBoost
  - Viola-Jones Face Detection

- Classification with SVMs
  - Support Vector Machines
  - HOG Detector
Detection via Classification: Main Idea

- Basic component: a binary classifier

Slide credit: Kristen Grauman
Detection via Classification: Main Idea

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

- Essentially, this is a brute-force approach with many local decisions.

Slide credit: Kristen Grauman
What is a Sliding Window Approach?

- Search over space and scale

- Detection as subwindow classification problem

- “In the absence of a more intelligent strategy, any global image classification approach can be converted into a localization approach by using a sliding-window search.”

B. Leibe
Detection via Classification: Main Idea

Fleshing out this pipeline a bit more, we need to:

1. Obtain training data
2. Define features
3. Define classifier

---

Slide credit: Kristen Grauman

B. Leibe
Feature extraction: Global Appearance

Simple holistic descriptions of image content
- Grayscale / color histogram
- Vector of pixel intensities

Slide credit: Kristen Grauman
Eigenfaces: Global Appearance Description

This can also be applied in a sliding-window framework...

Generate low-dimensional representation of appearance with a linear subspace.

Project new images to “face space”.

Detection via distance TO eigenspace

Identification via distance IN eigenspace

Slide credit: Kristen Grauman
Feature Extraction: Global Appearance

- Pixel-based representations are 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

Slide credit: Kristen Grauman

B. Leibe
Gradient-based Representations

- **Idea**
  - Consider edges, contours, and (oriented) intensity gradients
Gradient-based Representations

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

• Summarize local distribution of gradients with histogram
  - Locally orderless: offers invariance to small shifts and rotations
  - Still more spatial information than single global histogram
  - Contrast-normalization: try to correct for variable illumination

Slide credit: Kristen Grauman
Gradient-based Representations: Histograms of Oriented Gradients (HoG)

Map each grid cell in the input window to a histogram counting the gradients per orientation.

Code available: 
http://pascal.inrialpes.fr/soft/olt/

[Dalal & Triggs, CVPR 2005]
Classifier Construction

- How to compute a decision for each subwindow?

Image feature

<table>
<thead>
<tr>
<th>car</th>
<th>non-car</th>
<th>car</th>
<th>non-car</th>
<th>car</th>
<th>non-car</th>
</tr>
</thead>
</table>

Slide credit: Kristen Grauman
Discriminative Methods

- Learn a decision rule (classifier) assigning image features to different classes
Classifier Construction: Many Choices...

**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 credit: Kristen Grauman

B. Leibe

Slide adapted from Antonio Torralba
Boosting

- Build a strong classifier $H$ by combining a number of “weak classifiers” $h_1, \ldots, h_M$, 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


Slide credit: Kristen Grauman
AdaBoost: Intuition

Consider a 2D 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.

Figure adapted from Freund and Schapire

Slide credit: Kristen Grauman
AdaBoost: Intuition

Weak Classifier 1

Weights Increased

Weak Classifier 2

Slide credit: Kristen Grauman
AdaBoost: Intuition

Final classifier is combination of the weak classifiers

Slide credit: Kristen Grauman  B. Leibe
AdaBoost: Basic Steps

- Start with uniform weighting on training examples
- For $M$ iterations
  - Select best weak classifier for this weighted data set:
    - Evaluate weighted error for each feature, pick best one.
  - Reweight the training examples:
    - Incorrectly classified $\Rightarrow$ more weight
    - Correctly classified $\Rightarrow$ less weight
- Final classifier is combination of the weak ones, weighted according to the error they had.
- Simple, eh? Let’s flesh this out in more detail...
AdaBoost: Detailed Training Algorithm

1. Initialization: Set $w_n^{(1)} = \frac{1}{N}$ for $n = 1, \ldots, N$.
2. For $m = 1, \ldots, M$ iterations
   a) Train a new weak classifier $h_m(x)$ using the current weighting coefficients $W^{(m)}$ by minimizing the weighted error function
      $$J_m = \sum_{n=1}^{N} w_n^{(m)} I(h_m(x_n) \neq t_n)$$
      $$I(A) = \begin{cases} 1, & \text{if } A \text{ is true} \\ 0, & \text{else} \end{cases}$$
   b) Estimate the weighted error of this classifier on $X$:
      $$\epsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} I(h_m(x_n) \neq t_n)}{\sum_{n=1}^{N} w_n^{(m)}}$$
   c) Calculate a weighting coefficient for $h_m(x)$:
      $$\alpha_m = \ln \left( \frac{1 - \epsilon_m}{\epsilon_m} \right)$$
   d) Update the weighting coefficients:
      $$w_n^{(m+1)} = w_n^{(m)} \exp \{ \alpha_m I(h_m(x_n) \neq t_n) \}$$

B. Leibe
AdaBoost: Recognition

- Evaluate all selected weak classifiers on test data.
  \[ h_1(x), \ldots, h_m(x) \]

- Final classifier is weighted combination of selected weak classifiers:
  \[ H(x) = \text{sign} \left( \sum_{m=1}^{M} \alpha_m h_m(x) \right) \]

- Very simple procedure!
  - Less than 10 lines in Matlab!
  - But works extremely well in practice...
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

- Now we’ll take AdaBoost and see how the Viola-Jones face detector works

Slide credit: Kristen Grauman
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 → scale features directly for same cost

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

Integral image

\[
D = 1 + 4 - (2 + 3) \\
= A + (A + B + C + D) - (A + C + A + B) \\
= D
\]

Slide credit: Kristen Grauman

B. Leibe

[Viola & Jones, CVPR 2001]
Example

Integral Image

(x, y)
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

Weak classifier: \[ \text{filter output} > \theta? \]

Slide credit: Kristen Grauman

B. Leibe

[Viola & Jones, CVPR 2001]
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:

\[ h_t(x) = \begin{cases} 
+1 & \text{if } f_t(x) > \theta_t \\
-1 & \text{otherwise} 
\end{cases} \]

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

Outputs of a possible rectangle feature on faces and non-faces.

Slide credit: Kristen Grauman
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

Cascading Classifiers for Detection

- Even if the filters are fast to compute, each new image has a lot of possible windows to search.
- 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]

Slide credit: Kristen Grauman
Cascading Classifiers

- Chain classifiers that are progressively more complex and have lower false positive rates:

![Cascading Classifiers Diagram]

Receiver operating characteristic

% False Pos

% Detection

0 50

100

50

0

50

100

Classifier 1

Classifier 2

Classifier 3

FACE

IMAGE SUB-WINDOW

F

NON-FACE

F

NON-FACE

F

NON-FACE

Slide credit: Svetlana Lazebnik
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://sourceforge.net/projects/opencvlibrary/]

Slide credit: Kristen Grauman
Viola-Jones Face Detector: Results

First two features selected

- Performance
  - 384 by 288 pixel images detected at 15 fps on a conventional 700 MHz Intel Pentium III in 2001.
  - Training time = weeks

Slide adapted from Kristen Grauman
Viola-Jones Face Detector: Results
Viola-Jones Face Detector: Results
Viola-Jones Face Detector: Results
You Can Try It At Home...

- The Viola & Jones detector was a huge success
  - First real-time face detector available
  - Many derivative works and improvements

- C++ implementation available in OpenCV [Lienhart, 2002]
  - [http://sourceforge.net/projects/opencvlibrary/](http://sourceforge.net/projects/opencvlibrary/)

- Matlab wrappers for OpenCV code available, e.g. here

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

Slide credit: Kristen Grauman
Classifier Construction: Many Choices...

**Nearest neighbor**

10^6 examples

Shakhnarovich, Viola, Darrell 2003
Berg, Berg, Malik 2005...

**Support Vector Machines**

Guyon, Vapnik
Heisele, Serre, Poggio, 2001,...

**Boosting**

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

**Neural networks**

LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998...

**Conditional Random Fields**

McCallum, Freitag, Pereira 2000; Kumar, Hebert 2003, ...
Linear Classifiers

Let

\[ w = \begin{bmatrix} a \\ c \end{bmatrix}, \quad x = \begin{bmatrix} x \\ y \end{bmatrix} \]

\[ ax + cy + b = 0 \]

\[ w \cdot x + b = 0 \]
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 (i.e. line for 2D case)
- Maximize the margin between the positive and negative training examples
Support Vector Machines

- Want line that maximizes the margin.

\[ \begin{align*}
\mathbf{x}_i \text{ positive } (y_i = 1): & \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1 \\
\mathbf{x}_i \text{ negative } (y_i = -1): & \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1
\end{align*} \]

For support, vectors, \( \mathbf{x}_i \cdot \mathbf{w} + b = \pm 1 \)

**Quadratic optimization problem**

Minimize \( \frac{1}{2} \mathbf{w}^T \mathbf{w} \)

Subject to \( y_i (\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 \)

Finding the Maximum Margin Line

- Solution: \( w = \sum_i \alpha_i y_i x_i \)

Finding the Maximum Margin Line

- **Solution:**
  \[ w = \sum_i \alpha_i y_i x_i \]
  \[ w \cdot x + b = \sum_i \alpha_i y_i x_i \cdot x + b \]

- **Classification function:**
  \[ f(x) = \text{sign} (w \cdot x + b) \]
  \[ = \text{sign} \left( \sum_i \alpha_i x_i \cdot x + b \right) \]
  *If* \( f(x) < 0 \), classify as neg., *if* \( f(x) > 0 \), classify as pos.

- Notice that this relies on an *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)

Questions

- What if the features are not 2d?
- What if the data is not linearly separable?
- What if we have more than just two categories?
Questions

- What if the features are not 2d?
  - Generalizes to d-dimensions - replace line with “hyperplane”

- What if the data is not linearly separable?
- What if we have more than just two categories?
Questions

• What if the features are not 2d?
  ➢ Generalizes to d-dimensions - replace line with “hyperplane”

• What if the data is not linearly separable?
  ➢ Non-linear SVMs with special kernels

• What if we have more than just two categories?
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:

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

More on that in the Machine Learning lecture...

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

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

Some Often-Used 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):**
  \[ K(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right) \]

Slide from Andrew Moore’s tutorial: [http://www.autonlab.org/tutorials/svm.html](http://www.autonlab.org/tutorials/svm.html)
Questions

• What if the features are not 2d?
  ➢ Generalizes to d-dimensions - replace line with “hyperplane”

• What if the data is not linearly separable?
  ➢ Non-linear SVMs with special kernels

• What if we have more than just two categories?
Multi-Class SVMs

- Achieve multi-class classifier by combining a number of binary classifiers

- **One vs. all**
  - Training: learn an SVM for each class vs. the rest
  - Testing: apply each SVM to test example and assign to it the class of the SVM that returns the highest decision value

- **One vs. one**
  - Training: learn an SVM for each pair of classes
  - Testing: each learned SVM “votes” for a class to assign to the test example
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.

Slide credit: Kristen Grauman
Pedestrian Detection

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

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

  Space-time rectangle features [Viola, Jones & Snow, ICCV 2003]

  SVM with HoGs [Dalal & Triggs, CVPR 2005]
Pedestrian detection with HoGs & SVMs

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

Slide credit: Kristen Grauman

B. Leibe
Summary: Sliding-Windows

- **Pros**
  - Simple detection protocol to implement
  - Good feature choices critical
  - Past successes for certain classes
  - Good detectors available (Viola & Jones, HOG, etc.)

- **Cons/Limitations**
  - High computational complexity
    - For example: 250,000 locations x 30 orientations x 4 scales = 30,000,000 evaluations!
    - This puts tight constraints on the classifiers we can use.
    - If training binary detectors independently, this means cost increases linearly with number of classes.
  - With so many windows, false positive rate better be low
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

![Sliding window vs Detector’s view](image-url)

Figure credit: Derek Hoiem
Limitations (continued)

- In practice, often entails large, cropped training set (expensive)
- Requiring good match to a global appearance description can lead to sensitivity to partial occlusions
References and Further Reading

• Read the Viola-Jones paper

• Viola-Jones Face Detector
  ➢ C++ implementation available in OpenCV [Lienhart, 2002]
    - [http://sourceforge.net/projects/opencvlibrary/](http://sourceforge.net/projects/opencvlibrary/)
  ➢ Matlab wrappers for OpenCV code available, e.g. here

• HOG Detector