Computer Vision - Lecture 10
Sliding-Window based Object Detection
3.12.2009

Bastian Leibe
RWTH Aachen
http://www.umic.rwth-aachen.de/multimedia
leibe@umic.rwth-aachen.de

Recap: Subspace Methods

Recap: Obj. Detection by Distance TO Eigenspace

Scan a window $\omega$ over the image and classify the window as object or non-object as follows:
- Project window to subspace and reconstruct as earlier.
- Compute the distance between $\omega$ and the reconstruction (reprojection error).
- Local minima of distance over all image locations $\Rightarrow$ object locations.
- Repeat at different scales.
- Possibly normalize window intensity such that $|\omega|=1$.

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.

Course Outline

- Image Processing Basics
- Segmentation & Grouping
- Recognition
  - Global Representations
  - Subspace Representations (remainder)
- Object Categorization I
  - Sliding Window based Object Detection
- Local Features & Matching
- Object Categorization II
- Part based Approaches
- 3D Reconstruction
- Motion and Tracking

Recap: Eigenfaces
Recap: Restrictions of PCA

- PCA minimizes projection error
- PCA is "unsupervised" no information on classes is used
- Discriminating information might be lost

Fischer’s Linear Discriminant Analysis (FLD)

- FLD is an enhancement to PCA
  - Constructs a discriminant subspace that minimizes the scatter between images of the same class and maximizes the scatter between different class images
  - Also sometimes called LDA...

Mean Images

- Let \( X_1, X_2, \ldots, X_c \) be the classes in the database and let each class \( X_i \), \( i = 1, 2, \ldots, c \) have \( k \) images \( x_{ij} \), \( j = 1, 2, \ldots, k \).
- We compute the mean image \( \mu_i \) of each class \( X_i \) as:
  \[
  \mu_i = \frac{1}{k} \sum_{j=1}^{k} x_{ij}
  \]
- Now, the mean image \( \mu \) of all the classes in the database can be calculated as:
  \[
  \mu = \frac{1}{c} \sum_{i=1}^{c} \mu_i
  \]

Scatter Matrices

- We calculate the within-class scatter matrix as:
  \[
  S_w = \sum_{i=1}^{c} \sum_{j=x_{ij}}(x_{ij} - \mu_i)(x_{ij} - \mu_i)^T
  \]
- We calculate the between-class scatter matrix as:
  \[
  S_b = \sum_{i=1}^{c} N_i(\mu_i - \mu)(\mu_i - \mu)^T
  \]

Visualization

- We maximize distance between classes
- Minimize distance within a class
- Criterion: \( J(w) = w^T S_w w + \sum_{i=1}^{c} \lambda_i S_{w_i} w \)
- \( S_b \) ... between-class scatter matrix
- \( S_w \) ... within-class scatter matrix
- Vector \( w \) is a solution of a generalized eigenvalue problem
- Classification function:
  \[
  g(x) = w^T x + w_0 \geq 0
  \]
FLD Computation

- Maximization of
  \[ J(w) = \frac{w^T S_b w}{w^T S_w w} \]
  is given by solution of generalized eigenvalue problem
  \[ S_b w = \lambda S_w w \]
- Defining \( v = S_b^{-1} w \) we get
  \[ S_b^{-1} S_w^{-1} v = \lambda v \]
  which is a regular eigenvalue problem.
- For the c-class case we obtain (at most) c-1 projections.

Face Recognition Difficulty: Lighting

- The same person can appear dramatically different when light sources illuminate the face from different directions.
- Idea:
  - Use FLD to find class-specific linear projections that compensate for lighting/facial expression.

Application: Fisherfaces

- Singularity problem
  - The within-class scatter is always singular for face recognition, since #training images << #pixels
  - This problem is overcome by applying PCA first
  \[
  W_{PC}^T = W_{PC}^T W_{PC}^{-1} \\
  W_{PC} = \text{arg max } \|W^T S_{PC} W\| \\
  W_{AI} = \text{arg max } \|W^T S_{AI} W_{PC} W\| \\
  \]

Fisherfaces: Experiments

Example Application: Fisherfaces

- Visual discrimination task
  - Training data:
    - \( C_1 \): Subjects with glasses
    - \( C_2 \): Subjects without glasses
  - Test:
    - glasses?
    - Take each image as a vector of pixel values and apply FLD...
Fisherfaces: Interpretability

- Example Fisherface for recognition “Glasses/NoGlasses”

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

- 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**
  - 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.*

**Other Types of Categories**

- **Functional Categories**
  - e.g. chairs = “something you can sit on”

- **Ad-hoc categories**
  - e.g. “something you can find in an office environment”
Challenges: Robustness

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

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

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.

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."

Slide credit: Kristen Grauman
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

Feature extraction:
- Car/non-car Classifier

Training examples

Feature extraction

Car/non-car

Classifier

Feature Extraction: Global Appearance

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

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".

Recognition via nearest neighbors in face space.

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

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 (HoG)

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

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

B. Leibe

Classifier Construction

• How to compute a decision for each subwindow?

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...

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

• Support Vector Machines
  Guyon, Vapnik, Heilebe, 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, ...

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 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.
**AdaBoost: Intuition**

**Week Classifer 1**

**Weights Increased**

**Week Classifer 2**

---

**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 classifiers.

---

**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.

---

**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

---

**Example**

Integral Image

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

\[
\sum_{y=0}^{y} \sum_{x=0}^{x} (I(x',y') - I(x,y))
\]

\[I(x',y') = \begin{cases} +1 & \text{if } \left| y - y' \right| < \frac{1}{2} \text{ and } \left| x - x' \right| < \frac{1}{2} \\ -1 & \text{otherwise} \end{cases}
\]

Viola & Jones, CVPR 2001

---

Slide credit: Kristen Grauman
**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 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.

---

**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:**

\[ h_i(x) = \begin{cases} 
+1 & \text{if } f_i(x) > 0_i \\
-1 & \text{otherwise} 
\end{cases} \]

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

---

**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

---

**Cascading Classifiers**

- Chain classifiers that are progressively more complex and have lower false positive rates.
**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/)

**Viola-Jones Face Detector: Results**

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

**Detecting profile faces?**

Detecting profile faces requires training separate detector with profile examples.
**Viola-Jones Face Detector: Results**

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/
- 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

---

**Classifier construction: many choices...**

- 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, ...

---

**Linear Classifiers**

- Let 
  \[ \mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} \]
  \[ a \mathbf{x} + c \mathbf{y} + b = 0 \]
**Perceptual and Sensory Augmented Computing**  
**Computer Vision WS 09/10**

---

### Lines in $\mathbb{R}^2$

Let $\mathbf{w} = \begin{bmatrix} a \\ c \\ b \end{bmatrix}$ and $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

\[ ax + cy + b = 0 \]

\[ \mathbf{w} \cdot \mathbf{x} + b = 0 \]

---

### 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 (SVMs)

- Want line that maximizes the margin.

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

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

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

---

### Finding the Maximum Margin Line

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

- Classification function:

\[ f(x) = \text{sign} (\mathbf{w} \cdot \mathbf{x} + b) \]

- If $f(x) < 0$, classify as neg., if $f(x) > 0$, classify as pos.

- Notice that it relies on an inner product between the test point $\mathbf{x}$ and the support vectors $\mathbf{x}_i$

(Using the optimization problem also involves computing the inner products $\mathbf{x}_i \cdot \mathbf{x}_j$ between all pairs of training points)

---

### Finding the Maximum Margin Line

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

\[ \mathbf{w} \cdot \mathbf{x} + b = \sum \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b \]

- Classification function:

\[ f(x) = \text{sign} (\mathbf{w} \cdot \mathbf{x} + b) \]

- If $f(x) < 0$, classify as neg., if $f(x) > 0$, classify as pos.

- Notice that it relies on an inner product between the test point $\mathbf{x}$ and the support vectors $\mathbf{x}_i$

(Solving the optimization problem also involves computing the inner products $\mathbf{x}_i \cdot \mathbf{x}_j$ 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?

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...

Nonlinear SVMs

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

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

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

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

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 network):

\[ K(x_i, x_j) = \exp\left(-\frac{|x_i - x_j|^2}{2\sigma^2}\right) \]
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.

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]

Example: Pedestrian Detection with HoG and SVMs

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

• Train a linear SVM using training set of pedestrian vs. non-pedestrian windows.

Code available: http://pascal.inrialpes.fr/opencote/
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)

- 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

References and Further Reading

- Read the Viola-Jones paper
  - (first version appeared at CVPR 2001)

- Viola-Jones Face Detector
  - C++ implementation available in OpenCV [Lienhart, 2002]
    - [http://sourceforge.net/projects/opencvlibrary/]
  - Matlab wrappers for OpenCV code available, e.g. here
    - [http://www.mathworks.com/matlabcentral/fileexchange/19912]

- HOG Detector
  - Code available: [http://pascal.inrialpes.fr/software/]

Limitations (continued)

- Not all objects are “box” shaped

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