Generic object recognition, classifiers, detection with sliding windows

Aykut Erdem
Dept. of Computer Engineering
Hacettepe University

Outline

• Generic object recognition
• Classifiers
• Object category detection
• Sliding-windows based models

Instance-level recognition problem

John’s car

Generic categorization problem
Visual categorization

• Given a small number of training images of a category, recognize a-priori unknown instances of that category and assign the correct category label.
• Many different ways to categorize

Training Images

Many different ways to categorize label.
Given a small number of training images of a category, recognize a-priori unknown instances of that category and assign the correct category label.
Many different ways to categorize.

Testing phase

Training phase

Part I: Image features

Testing

Training

Training Images

Training Labels

Image Features

Classifier Training

Trained Classifier

Test Image

Image Features

Trained Classifier

Prediction

Outdoor

Image Features

Classifier Training

Trained Classifier

Image Features

Classifier Training

Trained Classifier

Training Images

Training Labels

Image Features

Classifier Training

Trained Classifier

Training Images

Training Labels

Image Features

Classifier Training

Trained Classifier

Image Features

Classifier Training

Trained Classifier

Feature quantization and concatenation
Choice of representation

- Window-based
- Part-based

Right features depend on what you want to know

- Object: 2D shape
  - Local shape info, shading, shadows, texture
- Scene: overall layout
  - Linear perspective, gradients
- Material properties: albedo, feel, hardness, ...
  - Color, texture
- Motion
  - Optical flow, tracked points

General Principles of Representation

- Coverage
  - Ensure that all relevant info is captured
- Conciseness
  - Minimize number of features without sacrificing coverage
- Directness
  - Ideal features are independently useful for prediction

Image representations

- Templates
  - Intensity, gradients, etc.
- Histograms
  - Color, texture, SIFT descriptors, etc.
- Average of features
Image Representations: Histograms

Global histogram
• Represent distribution of features
  - Color, texture, depth, ...

Image Representations: Histograms

Clustering

Use the same cluster centers for all images

Image Representations: Histograms

Histogram: Probability or count of data in each bin

Joint histogram
• Requires lots of data
• Loss of resolution to avoid empty bins

Marginal histogram
• Requires independent features
• More data/bin than joint histogram

Computing histogram distance

$$\text{histint}(h_i, h_j) = 1 - \sum_{m=1}^{K} \min(h_i(m), h_j(m))$$

Histogram intersection (assuming normalized histograms)

$$\chi^2(h_i, h_j) = \frac{1}{2} \sum_{m=1}^{K} \frac{(h_i(m) - h_j(m))^2}{h_i(m) + h_j(m)}$$

Chi-squared Histogram matching distance

Cars found by color histogram matching using chi-squared
Histograms: Implementation issues

- Quantization
  - Few bins: fast but applicable only with few dimensions
  - Many bins: need more data, coarser representation

- Matching
  - Few bins: histogram intersection or Euclidean may be faster
  - Many bins: chi-squared often works better
  - Earth mover’s distance is good for when nearby bins represent similar values

What kind of things do we compute histograms of?

- Color
  - L*a*b* color space
  - HSV color space

- Texture (filter banks or HOG over regions)

What kind of things do we compute histograms of?

- Histograms of descriptors
  - SIFT - Lowe IJCV 2004

- “Bag of words”
But what about layout?

All of these images have the same color histogram

Spatial pyramid

Compute histogram in each spatial bin

Image Categorization: Bag of Words

Training
1. Extract keypoints and descriptors for all training images
2. Cluster descriptors
3. Quantize descriptors using cluster centers to get “visual words”
4. Represent each image by normalized counts of “visual words”
5. Train classifier on labeled examples using histogram values as features

Testing
1. Extract keypoints/descriptors and quantize into visual words
2. Compute visual word histogram
3. Compute label or confidence using classifier
Often used features

• Scene: GIST, Spatial pyramid BoW, color
• Object: Spatial pyramid BoW, HOG, color
• Material: texture, color

Things to remember about representation

• Most features can be thought of as templates, histograms (counts), or combinations
• Think about the right features for the problem
  − Coverage
  − Concision
  − Directness

Outline

• Generic object recognition
• Classifiers
  • Object category detection
  • Sliding-windows based models

Classifiers
A classifier maps from the feature space to a label.

Different types of classification

- Exemplar-based: transfer category labels from examples with most similar features
  - What similarity function? What parameters?
- Linear classifier: confidence in positive label is a weighted sum of features
  - What are the weights?
- Non-linear classifier: predictions based on more complex function of features
  - What form does the classifier take? Parameters?
- Generative classifier: assign to the label that best explains the features (makes features most likely)
  - What is the probability function and its parameters?

Note: You can always fully design the classifier by hand, but usually this is too difficult. Typical solution: learn from training examples.

Many classifiers to choose from

- SVM
- Neural networks
- Naïve Bayes
- Bayesian network
- Logistic regression
- Randomized Forests
- Boosted Decision Trees
- K-nearest neighbor
- RBMs
- Etc.

Which is the best one?

Recognition task and supervision

- Images in the training set must be annotated with the “correct answer” that the model is expected to produce

Contains a motorbike
Spectrum of supervision

Definition depends on task

Generalization

- How well does a learned model generalize from the data it was trained on to a new test set?

Generalization

- Components of generalization error
  - **Bias**: How much the average model over all training sets differ from the true model?
    - Error due to inaccurate assumptions/simplifications made by the model
  - **Variance**: How much models estimated from different training sets differ from each other

- Underfitting: model is too “simple” to represent all the relevant class characteristics
  - High bias and low variance
  - High training error and high test error

- Overfitting: model is too “complex” and fits irrelevant characteristics (noise) in the data
  - Low bias and high variance
  - Low training error and high test error

Bias-Variance Trade-off

- Models with too few parameters are inaccurate because of a large bias (not enough flexibility).
- Models with too many parameters are inaccurate because of a large variance (too much sensitivity to the sample).
Bias-Variance Trade-off

$$E(\text{MSE}) = \text{noise}^2 + \text{bias}^2 + \text{variance}$$

- Unavoidable error
- Error due to incorrect assumptions
- Error due to variance of training samples

See the following for explanations of bias-variance (also Bishop’s “Neural Networks” book):

- [http://www.inf.ed.ac.uk/teaching/courses/mlsc/Notes/Lecture4/BiasVariance.pdf](http://www.inf.ed.ac.uk/teaching/courses/mlsc/Notes/Lecture4/BiasVariance.pdf)

Remember...

- No classifier is inherently better than any other: you need to make assumptions to generalize

- Three kinds of error
  - Inherent: unavoidable
  - Bias: due to over-simplifications
  - Variance: due to inability to perfectly estimate parameters from limited data

Very brief tour of some classifiers

- SVM
- Neural networks
- Naïve Bayes
- Bayesian network
- Logistic regression
- Randomized Forests
- Boosted Decision Trees
- K-nearest neighbor
- RBMs
- etc.

Generative vs. Discriminative Classifiers

Generative Models
- Represent both the data and the labels
- Often, makes use of conditional independence and priors
- Examples
  - Naïve Bayes classifier
  - Bayesian network
- Models of data may apply to future prediction problems

Discriminative Models
- Learn to directly predict the labels from the data
- Often, assume a simple boundary (e.g., linear)
- Examples
  - Logistic regression
  - SVM
  - Boosted decision trees
- Often easier to predict a label from the data than to model the data
**K-nearest neighbor**

![K-nearest neighbor diagram](image)

**Nearest neighbors: pros and cons**

- **Pros:**
  - Simple to implement
  - Flexible to feature / distance choices
  - Naturally handles multi-class cases
  - Can do well in practice with enough representative data

- **Cons:**
  - Large search problem to find nearest neighbors
  - Storage of data
  - Must know we have a meaningful distance function

---

**Where in the World?**

A nearest neighbor recognition example


**Where in the World?**

![Image of Notre Dame Cathedral]
Where in the World?

6+ million geotagged photos by 109,788 photographers
Annotated by Flickr users

Which scene properties are relevant?
A scene is a single surface that can be represented by global (statistical) descriptors.

Which scene properties are relevant?

- **Gist scene descriptor**
- **Color Histograms** - L*A*B* 4x14x14 histograms
- **Texton Histograms** - 512 entry, filter bank based
- **Line Features** - Histograms of straight line stats

Global texture: capturing the “Gist” of the scene

Capture global image properties while keeping some spatial information.

Slide credit: K. Grauman

Scene Matches

Slide credit: J. Hays
The Importance of Data

Feature Performance

Support Vector Machines (SVMs)

• Discriminative classifier based on optimal separating line (for 2d case)

• Maximize the margin between the positive and negative training examples
**Linear classifier**

Finding the linear hyperplane that separate examples of different categories

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

**Classification Margin**

- Distance from example \( x_i \) to the separator is \( r = \frac{w^T x_i + b}{|w|} \)
- Examples closest to the hyperplane are support vectors.
- **Margin** \( \rho \) of the separator is the distance between support vectors.

**Linear Separators**

- Which of the linear separators is optimal?

**Maximum Margin Classification**

- Implies that only support vectors matter; other training examples are ignorable.
Linear SVM Mathematically

- Let training set \( \{ (x_i, y_i) \}_{i=1}^{n}, x_i \in \mathbb{R}^d, y_i \in \{-1, 1\} \) be separated by a hyperplane with margin \( q \). Then for each training example \((x_i, y_i)\):
  \[
  \begin{align*}
  w^T x_i + b \leq -q/2 & \quad \text{if } y_i = -1 \\
  w^T x_i + b \geq q/2 & \quad \text{if } y_i = 1
  \end{align*}
  \]

- For every support vector \( x_s \) the above inequality is an equality. After rescaling \( w \) and \( b \) by \( q/2 \) in the equality, we obtain that distance between each \( x_s \) and the hyperplane is
  \[
  r = \frac{1}{\sum \alpha_i y_i x_i \cdot x_i}
  \]

- Then the margin can be expressed through (rescaled) \( w \) and \( b \) as:
  \[
  \rho = 2r = \frac{1}{\|w\|}
  \]

Solving the Optimization Problem

minimize \( L_p(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(w^T x_i + b) - 1\right) \)

\[\text{s.t. } \alpha_i \geq 0\]

\[
\begin{align*}
\frac{\partial L_p}{\partial w} &= 0 \quad \Rightarrow \quad w = \sum_{i=1}^{n} \alpha_i y_i x_i \\
\frac{\partial L_p}{\partial b} &= 0 \quad \Rightarrow \quad \sum_{i=1}^{n} \alpha_i y_i = 0
\end{align*}
\]

Solving the Optimization Problem

minimize \( L_p(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(w^T x_i + b) - 1\right) \)

\[\text{s.t. } \alpha_i \geq 0\]

Lagrangian Dual Problem

maximize \( \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \)

\[\text{s.t. } \alpha_i \geq 0 \text{, and } \sum_{i=1}^{n} \alpha_i y_i = 0\]
Soft Margin Classification

- What if the training set is not linearly separable?
- *Slack variables* $\xi_i$ can be added to allow misclassification of difficult or noisy examples, resulting margin called soft.

Large Margin Linear Classifier

- Formulation:

$$\text{minimize } \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i$$

such that

$$y_i (w^T x_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

- Parameter C is a trade off factor

Large Margin Linear Classifier

- Formulation: (Lagrangian Dual Problem)

$$\text{maximize } \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

such that

$$0 \leq \alpha_i \leq C$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

Linear SVMs: Recap

- The classifier is a *separating hyperplane*.
- Most “important” training points are support vectors; they define the hyperplane.
- Quadratic optimization algorithms can identify which training points $x_i$ are support vectors with non-zero Lagrangian multipliers $\alpha_i$. 

Classifiers: Kernelized SVM

What if the data is not linearly separable?

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

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:

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

Examples of kernel functions

- Linear: \( K(x_i, x_j) = x_i^T x_j \)

- Gaussian RBF: \( K(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right) \)

- Histogram intersection:

\[
K(x_i, x_j) = \sum_k \min(x_i(k), x_j(k))
\]

Using SVMs

- Good general purpose classifier
  - Generalization depends on margin, so works well with many weak features
  - No feature selection
  - Usually requires some parameter tuning

- Choosing kernel
  - Linear: fast training/testing – start here
  - RBF: related to neural networks, nearest neighbor
  - Chi-squared, histogram intersection: good for histograms (but slower, esp. chi-squared)
  - Can learn a kernel function
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

Measuring classification performance

- Confusion matrix

- **Accuracy**
  - \( \frac{TP+TN}{TP+TN+FP+FN} \)

- **True Positive Rate=Recall**
  - \( \frac{TP}{TP+FN} \)

- **False Positive Rate**
  - \( \frac{FP}{FP+TN} \)

- **Precision**
  - \( \frac{TP}{TP+FP} \)

- **F1 Score**
  - \( \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \)

ROC curve

- Receiver_operating_characteristic
  - Area under the curve (AUC)
  - Equal Error Rate (EER)

Outline

- Generic object recognition
- Classifiers
- Object category detection
- Sliding-windows based models
Object Category Detection

- Focus on object search: “Where is it?”
- Build templates that quickly differentiate object patch from background patch

Challenges in modeling the object class

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

Challenges in modeling the non-object class

- True Detections
- Bad Localization
- Confused with Similar Object
- Misc. Background
- Confused with Dissimilar Objects

General Process of Object Recognition

1. Specify Object Model: What are the object parameters?
2. Generate Hypotheses
3. Score Hypotheses
4. Resolve Detections
Specifying an object model

1. Statistical Template in Bounding Box
   - Object is some (x,y,w,h) in image
   - Features defined wrt bounding box coordinates

Specifying an object model

2. Articulated parts model
   - Object is configuration of parts
   - Each part is detectable

Specifying an object model

3. Hybrid template/parts model

Specifying an object model

4. 3D-ish model
   - Object is collection of 3D planar patches under affine transformation
General Process of Object Recognition

Specify Object Model
→ Generate Hypotheses
→ Score Hypotheses
→ Resolve Detections

Propose an alignment of the model to the image

Generating hypotheses

1. Sliding window
   - Test patch at each location and scale

2. Voting from patches/keypoints
   - Interest Points
   - Matched Codebook Entries
   - Probabilistic Voting
   - 3D Voting Space (continuous)

ISM model by Leibe et al.
Generating hypotheses

3. Region-based proposal

General Process of Object Recognition

Specify Object Model

Generate Hypotheses

Score Hypotheses

Resolve Detections

Mainly gradient-based features, usually based on summary representation, many classifiers

Resolving detection scores

1. Non-max suppression

Score = 0.1

Score = 0.8

Score = 0.8
Resolving detection scores

2. Context/reasoning

Object category detection in computer vision

• Goal: detect all pedestrians, cars, monkeys, etc in image

Basic Steps of Category Detection

1. Align
   - E.g., choose position, scale orientation
   - How to make this tractable?

2. Compare
   - Compute similarity to an example object or to a summary representation
   - Which differences in appearance are important?

Outline

• Generic object recognition
• Classifiers
• Object category detection
• Sliding-windows based models
Statistical Template

- Object model = sum of scores of features at fixed positions

\[+3 +2 -2 -1 -2.5 -0.5 \geq 7.5\]
Non-object

\[+4 +1 +0.5 +3 +0.5 -10.5 \geq 7.5\]
Object

Design challenges

- How to efficiently search for likely objects
  - Even simple models require searching hundreds of thousands of positions and scales
- Feature design and scoring
  - How should appearance be modeled? What features correspond to the object?
- How to deal with different viewpoints?
  - Often train different models for a few different viewpoints
- Implementation details
  - Window size
  - Aspect ratio
  - Translation/scale step size
  - Non-maxima suppression
Example: Dalal-Triggs pedestrian detector

1. Extract fixed-sized (64x128 pixel) window at each position and scale
2. Compute HOG (histogram of gradient) features within each window
3. Score the window with a linear SVM classifier
4. Perform non-maxima suppression to remove overlapping detections with lower scores

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

- Tested with
  - RGB
  - LAB
  - Grayscale
- Gamma Normalization and Compression
  - Square root
  - Log

Slightly better performance vs. grayscale
Very slightly better performance vs. no adjustment

Outperforms
**Histogram of gradient orientations**

- Votes weighted by magnitude
- Bilinear interpolation between cells

Orientation: 9 bins (for unsigned angles)

Histograms in 8x8 pixel cells

Normalize with respect to surrounding cells

\[ L2 - norm : v \rightarrow v/\sqrt{||v||^2 + \epsilon^2} \]
Viola-Jones sliding window detector

**Fast** detection through two mechanisms
- Quickly eliminate unlikely windows
- Use features that are fast to compute

0.16 = w^T x - b

\[ \text{sign}(0.16) = 1 \]

\[ \Rightarrow \text{pedestrian} \]

Cascade for Fast Detection

- Choose threshold for low false negative rate
- Fast classifiers early in cascade
- Slow classifiers later, but most examples don’t get there
Features that are fast to compute

- "Haar-like features"
  - Differences of sums of intensity
  - Thousands, computed at various positions and scales within detection window

- Integral Images
  \[ ii = \text{cumsum}(\text{cumsum}(im, 1), 2) \]
  \[ ii(x,y) = \text{Sum of the values in the grey region} \]

Feature selection with Adaboost

- Create a large pool of features (180K)
- Select features that are discriminative and work well together
  - "Weak learner" = feature + threshold + parity
    \[ h_j(x) = \begin{cases} 
    1 & \text{if } p_j f_j(x) < p_j \theta_j \\
    0 & \text{otherwise} 
    \end{cases} \]
  - Choose weak learner that minimizes error on the weighted training set
  - Reweight
Top 2 selected features

Viola-Jones details
- 38 stages with 1, 10, 25, 50 ... features
  - 6061 total used out of 180K candidates
  - 10 features evaluated on average
- Training Examples
  - 4916 positive examples
  - 10000 negative examples collected after each stage
- Scanning
  - Scale detector rather than image
  - Scale steps ~ 1.25 (factor between two consecutive scales)
  - Translation 1*scale (# pixels between two consecutive windows)
- Non-max suppression: average coordinates of overlapping boxes
- Train 3 classifiers and take vote

Viola-Jones Results
Speed = 15 FPS (in 2001)

Strengths and Weaknesses of Statistical Template Approach
Strengths
- Works very well for non-deformable objects: faces, cars, upright pedestrians
- Fast detection

Weaknesses
- Not so well for highly deformable objects
- Not robust to occlusion
- Requires lots of training data
Tricks of the trade

- Details in feature computation really matter
  - E.g., normalization in Dalal-Triggs improves detection rate by 27% at fixed false positive rate

- Template size
  - Typical choice is size of smallest detectable object

- “Jittering” to create synthetic positive examples
  - Create slightly rotated, translated, scaled, mirrored versions as extra positive examples

- Bootstrapping to get hard negative examples
  1. Randomly sample negative examples
  2. Train detector
  3. Sample negative examples that score > -1
  4. Repeat until all high-scoring negative examples fit in memory

Consumer application: iPhoto 2009

- Things iPhoto thinks are faces

Influential Works in Detection

  - Basic idea of statistical template detection (I think), bootstrapping to get “face-like” negative examples, multiple whole-face prototypes (in 1994)

  - “Parts” at fixed position, non-maxima suppression, simple cascade, rotation, pretty good accuracy, fast

  - Careful feature engineering, excellent results, cascade

- Viola-Jones (2001, 2004) : ~11,000
  - Haar-like features, Adaboost as feature selection, hyper-cascade, very fast, easy to implement

- Dalal-Triggs (2005) : ~3250
  - Careful feature engineering, excellent results, HOG feature, online code

- Felzenszwalb-Huttenlocher (2000) : ~1000
  - Efficient way to solve part-based detectors

- Felzenszwalb-McAllester-Ramanan (2008) : ~800
  - Excellent template/parts-based blend

Things to remember

- Sliding window for search

- Features based on differences of intensity (gradient, wavelet, etc.)
  - Excellent results require careful feature design

- Boosting for feature selection

- Integral images, cascade for speed

- Bootstrapping to deal with many, many negative examples