Recognition

Computer Vision I
CSE252A
Lecture 20
Computer Vision Overview

- Physics-based computer vision (computational photography)
  - Color
  - BRDF
  - Photometric stereo
  - De-noising, image processing
- Structure from motion
  - Multiple cameras, multiple views
  - Reconstruction, 3D
  - Movie industry
- Learning based Computer Vision
  - Object detection
  - Object recognition
  - Image classification
  - Segmentation
    - Image parsing (supervised)
    - Segmentation (unsupervised)
Detection / Recognition

Given a database of objects and an image determine what, if any of the objects are present in the image.
Recognition

Given a database of objects and an image determine what, if any of the objects are present in the image.
Problem:
Recognizing instances
Recognizing categories
Where are the coral heads and which ones are healthy and which are bleached?

Input Image

Segmented/labeled Image

Bleached  Healthy  Partially Bleached
Object Recognition: The Problem

Given: A database $D$ of “known” objects and an image $I$:

1. Determine which (if any) objects in $D$ appear in $I$
2. Determine the pose (rotation and translation) of the object

WHAT AND WHERE!!!
Recognition Challenges

• Within-class variability
  – Different objects within the class have different shapes or different material characteristics
  – Deformable
  – Articulated
  – Compositional

• Pose variability
  – 2-D Image transformation (translation, rotation, scale)
  – 3-D Pose Variability (perspective, orthographic projection)

• Lighting
  – Direction (multiple sources & type)
  – Color
  – Shadows

• Occlusion – partial

• Clutter in background -> false positives

Object detection / classification is HARD. State of the art sometimes makes you sad.
An (incomplete) directory of the image classification pipeline

1) Pre-processing (I=>I)
   - Contrast normalization
   - Color corrections
   - Resize
   - Rgb2grey

2) Feature extraction (I=>x)
   - Direct
     - Pixel values (patches)
     - SIFT
     - HOG
   - Statistical (bag of visual words)
     - Extract features
     - Map to dictionary
     - Pool across image region
     - Multiple layers
     - Deep-learning (ANN)
   - Geometric [objects]
     - Cuboids
     - Poselets

3) Classification (x => k)
   - Generative methods
     - Gaussian Mixture Models
   - Discriminative methods
     - Nearest Neighbour
     - SVM (kernelized or linear liblinear)
   - Boosting
     - Artificial Neural Networks
   - Decision trees
   - Random forests
Example: Face Detection

- Scan window over image.
- Classify window as either:
  - Face
  - Non-face
• So, what are the features?
• So, what is the classifier?
Simplest feature: Image as a Feature Vector

• Consider an n-pixel image to be a point in an n-dimensional space, $\mathbf{x} \in \mathbb{R}^n$.
• Each pixel value is a coordinate of $\mathbf{x}$.
More features

- Filter with multiple filters (bank of filters)
- Histogram of colors
- Histogram of Gradients (HOG)
- Haar wavelets
- Scale Invariant Feature Transform (SIFT)
- Speeded Up Robust Feature (SURF)
We will cover two types of features

- Bag of Visual Words
- Eigenfaces
Bag of Visual Words Framework

- Very common pipeline for image classification.
- Typically use the following steps
  - Feature extraction
  - Encode using a dictionary
  - Pool
- Originated in text classification where
  - The words themselves are the ‘features’
  - The dictionary is an actual dictionary
  - The pooling is a simple count of word occurrences
- **BOVW characteristics**
  - Ignores spatial layout
  - Robust to translation
  - Used for e.g. bikes, cars, busses, flowers, corals, foods.
Let’s look at an example

• From our work on Coral Reefs image annotation…
**Step 1. Preprocess**

**Input Image**

**Resize**

**Normalize Contrast**
Step 2A. Filtering \( F_P \in \mathbb{R}^8 \)

\[ I_P \in \mathbb{R}^1 \]


Computer Vision I
Step 2b. Filtering: $F_P \in \mathbb{R}^{24}$

$\mathcal{I}_P \in \mathbb{R}^3$


Computer Vision I
STEP 3. MAP TO VISUAL WORD

\[ w_i \in \mathbb{R}^{24} \]

\[ T_P = \arg \min_i ||w_i - F_p||_2 \]

\[ F_p \in \mathbb{R}^{24} \]

135 WORDS IN ‘DICTIONARY’

TEXTON MAP
STEP4A. HISTOGRAMS

Normalized histogram count

135 bins

Normalized histogram count
STEP4B. HISTOGRAMS AT MULTIPLE SCALES

Normalized histogram count
Bag of Visual Words - elements

- Base-feature extraction
- Dictionary learning / encoding
- Pooling
BOVW – Feature extraction

• Image patch (color or grey)
• Filter bank
• Histogram of Gradients
• Scale Invariant Feature Transform
• Color histogram
• Local Binary Patterns (LBP)
**BOVW – Dictionary**

- Create dictionary by using large amounts of extracted base-features from training data.
- Encoding method is often connected to the dictionary type

<table>
<thead>
<tr>
<th>Method</th>
<th>Dictionary learning</th>
<th>Encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vector quantization (k=1)</td>
<td></td>
<td>Hard encoding (nearest neighbor)</td>
</tr>
<tr>
<td>Soft VW (k=1)</td>
<td></td>
<td>Soft encoding (weight vector)</td>
</tr>
<tr>
<td>Sparse Coding</td>
<td></td>
<td>Same but fixed D.</td>
</tr>
<tr>
<td>Fisher Encoding</td>
<td>Gaussian mixture model</td>
<td>Gradient in parameter space</td>
</tr>
</tbody>
</table>

\[
\min_{D,s(i)} \sum_i \|Ds(i) - x(i)\|_2^2
\]

subject to \(\|D(j)\|_2^2 = 1, \forall j\) and \(\|s(i)\|_0 \leq k, \forall i\)
BOVW – Pooling

- Mean pooling (histogram)
- Max pooling
- Spatial pyramid kernel
Face detection and recognition

Face detection slides borrowed from S. Lazebnik. Who, in turn…

Many slides adapted from K. Grauman and D. Lowe
Face detection and recognition

Detection

Recognition

“Sally”
Consumer application: iPhoto 2009

http://www.apple.com/ilife/iphoto/
Consumer application: iPhoto 2009

- Can be trained to recognize pets!

Consumer application: iPhoto 2009

Things iPhoto thinks are faces
More fails
The space of all face images

• When viewed as vectors of pixel values, face images are extremely high-dimensional
  – 100x100 image = 10,000 dimensions
• However, relatively few 10,000-dimensional vectors correspond to valid face images
• We want to effectively model the subspace of face images
The space of all face images

- We want to construct a low-dimensional linear subspace that best explains the variation in the set of face images.
Principal Component Analysis

• Given: N data points \( x_1, \ldots, x_N \) in \( \mathbb{R}^d \)

• We want to find a new set of features that are linear combinations of original ones:

\[
u(x_i) = u^T(x_i - \mu)
\]

(\( \mu \): mean of data points)

• What unit vector \( u \) in \( \mathbb{R}^d \) captures the most variance of the data?

Forsyth & Ponce, Sec. 22.3.1, 22.3.2
Principal Component Analysis

- Direction that maximizes the variance of the projected data:

\[
\text{var}(u) = \frac{1}{N} \sum_{i=1}^{N} u^T (x_i - \mu)(u^T (x_i - \mu))^T
\]

\[
= u^T \left[ \sum_{i=1}^{N} (x_i - \mu)(x_i - \mu)^T \right] u
\]

\[
= u^T \Sigma u
\]

The direction that maximizes the variance is the eigenvector associated with the largest eigenvalue of \(\Sigma\).
Principal component analysis

• The direction that captures the maximum covariance of the data is the eigenvector corresponding to the largest eigenvalue of the data covariance matrix.

• Furthermore, the top $k$ orthogonal directions that capture the most variance of the data are the $k$ eigenvectors corresponding to the $k$ largest eigenvalues.
Eigenfaces: Key idea

• Assume that most face images lie on a low-dimensional subspace determined by the first $k$ ($k<d$) directions of maximum variance

• Use PCA to determine the vectors or “eigenfaces” $u_1, \ldots, u_k$ that span that subspace

• Represent all face images in the dataset as linear combinations of eigenfaces

Eigenfaces example

- Training images
- \( x_1, \ldots, x_N \)
Eigenfaces example

Top eigenvectors: $u_1, \ldots, u_k$

Mean: $\mu$
Eigenfaces example

Principal component (eigenvector) $u_k$

$\mu + 3\sigma_k u_k$

$\mu - 3\sigma_k u_k$
More examples

Using 1, 2, 3, .... K eigenfaces
Eigenfaces example

• Face $\mathbf{x}$ in “face space” coordinates:

$$
\mathbf{x} \rightarrow [\mathbf{u}_1^T (\mathbf{x} - \mu), \ldots, \mathbf{u}_k^T (\mathbf{x} - \mu)] = \mathbf{w}_1, \ldots, \mathbf{w}_k
$$
Eigenfaces example

• Face $\mathbf{x}$ in “face space” coordinates:

$$\mathbf{x} \rightarrow \begin{bmatrix} \mathbf{u}_1^T (\mathbf{x} - \mu), & \ldots, & \mathbf{u}_k^T (\mathbf{x} - \mu) \end{bmatrix} = \begin{bmatrix} w_1, & \ldots, & w_k \end{bmatrix}$$

• Reconstruction:

$$\hat{\mathbf{x}} = \mu + w_1 \mathbf{u}_1 + w_2 \mathbf{u}_2 + w_3 \mathbf{u}_3 + w_4 \mathbf{u}_4 + \ldots$$
Recognition with eigenfaces

- Process labeled training images:
- Find mean $\mu$ and covariance matrix $\Sigma$
- Find $k$ principal components (eigenvectors of $\Sigma$) $u_1, \ldots, u_k$
- Project each training image $x_i$ onto subspace spanned by principal components:
  $$(w_{i1}, \ldots, w_{ik}) = (u_1^T(x_i - \mu), \ldots, u_k^T(x_i - \mu))$$

- Given novel image $x$:
- Project onto subspace:
  $$(w_1, \ldots, w_k) = (u_1^T(x - \mu), \ldots, u_k^T(x - \mu))$$
- Optional: check reconstruction error $x - \hat{x}$ to determine whether image is really a face
- Classify as closest training face in $k$-dimensional subspace

Eigenfaces summary

• Dimensionality reduction for several purposes
  – Remove image noise
  – Model occlusions such as glasses
  – Reduce dimensionality to learn more effective classifiers

• Weakness
  – Models a generative subspace, not the most discriminative.
  – Nearest Neighbor classifier is costly for large datasets.

• Basis for much other face recognition work
  – Fisherfaces (discriminative subspaces)
Feature extraction - review

• We have looked at two examples
  – Bag of visual words
  – Eigenfaces

• There are many many more!
  – E.g. Deformable part models

• Tradeoff between discrimination and invariance
• Usually where you get the most ‘bang for your buck’
• Recently, ‘Deep Learning’ has become popular.
Pattern Classification Summary

• Supervised vs. Unsupervised: Do we have labels?
• Supervised [CSE 250A, B Saul, Elkan]
  – Generative [ECE 271A Vasconcelos]
    • Bayesian
  – Discriminative [ECE 271B Vasconcelos]
    • Nearest Neighbor
    • Neural Network
    • Support Vector Machine
    • Kernel methods
• Unsupervised [CSE 250C Saul]
  – Clustering
  – Reinforcement learning
Nearest Neighbor Classifier

\{ R_j \} are set of training images.

\[ ID = \arg \min_j \text{dist}(R_j, I) \]
Comments

- Sometimes called “Template Matching”
- Variations on distance function (e.g. $L_1$, robust distances)
- Multiple templates per class- perhaps many training images per class.
- Expensive to compute $k$ distances, especially when each image is big (N dimensional).
- May not generalize well to unseen examples of class.
- Some solutions:
  - Bayesian classification
  - Dimensionality reduction
Bayesian Decision Theory
An Example

- “Sorting incoming Fish on a conveyor according to species using optical sensing”
Pattern Classification, Chapter 1
• Adopt the lightness and add the width of the fish

\[ x^T = [x_1, x_2] \]
Bayesian Decision Theory
Continuous Features
Introduction

• The sea bass/salmon example

  • State of nature is a random variable, $\omega_i$
    • $\omega_1$ – the fish is a salmon
    • $\omega_2$ – the fish is a sea bass

• Prior Probabilities
  • $P(\omega_1), P(\omega_2)$

  • $P(\omega_i) > 0$
  • $P(\omega_1) + P(\omega_2) = 1$ (exclusivity and exhaustivity)

• Example prior: bass & salmon are equally likely
  • $P(\omega_1) = P(\omega_2) = \frac{1}{2}$ (uniform priors)
• Decision rule with only the prior information
  • Decide \( \omega_1 \) if \( P(\omega_1) > P(\omega_2) \) otherwise decide \( \omega_2 \)

• Use of the class–conditional information

• \( P(x | \omega_1) \) and \( P(x | \omega_2) \) describe the difference in lightness between populations of sea-bass and salmon
FIGURE 2.1. Hypothetical class-conditional probability density functions show the probability density of measuring a particular feature value $x$ given the pattern is in category $\omega_i$. If $x$ represents the lightness of a fish, the two curves might describe the difference in lightness of populations of two types of fish. Density functions are normalized, and thus the area under each curve is 1.0. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.
• Posterior, likelihood, evidence

\[ P(\omega_j | x) = \frac{P(x | \omega_j)P(\omega_j)}{P(x)} \]

(BAYES RULE)

• In words, this can be said as:
  Posterior = (Likelihood * Prior) / Evidence

• Where in case of two categories

\[ P(x) = \sum_{j=1}^{2} P(x | \omega_j)P(\omega_j) \]
FIGURE 2.2. Posterior probabilities for the particular priors $P(ω_1) = 2/3$ and $P(ω_2) = 1/3$ for the class-conditional probability densities shown in Fig. 2.1. Thus in this case, given that a pattern is measured to have feature value $x = 14$, the probability it is in category $ω_2$ is roughly 0.08, and that it is in $ω_1$ is 0.92. At every $x$, the posteriors sum to 1.0. From: Richard O. Duda, Peter E. Hart, and David G. Stork, Pattern Classification. Copyright © 2001 by John Wiley & Sons, Inc.
Intuitive decision rule given the posterior probabilities:

Given x:

if \( P(\omega_1 \mid x) > P(\omega_2 \mid x) \) \( \Rightarrow \) True state of nature = \( \omega_1 \)

if \( P(\omega_1 \mid x) < P(\omega_2 \mid x) \) \( \Rightarrow \) True state of nature = \( \omega_2 \)

Why do this?: Whenever we observe a particular \( x \), the probability of error is:

\[
P(error \mid x) = P(\omega_1 \mid x) \text{ if we decide } \omega_2
\]

\[
P(error \mid x) = P(\omega_2 \mid x) \text{ if we decide } \omega_1
\]
Plug-in classifiers

• Assume that class conditional distributions $P(x|\omega_i)$ have some parametric form - now estimate the parameters from the data.

• Common:
  – assume a normal distribution with shared covariance, different means; use usual estimates
  – Normal distribution but with different covariances;
Support Vector Machines

- Bayes classifiers and generative approaches in general try to model of the posterior, $p(\omega|x)$
- Instead, try to obtain the decision boundary directly
  - potentially easier, because we need to encode only the geometry of the boundary, not any irrelevant wiggles in the posterior.
  - Not all points affect the decision boundary
Support Vector Machines

- Set $S$ of points $x_i \in \mathbb{R}^n$, each $x_i$ belongs to one of two classes $y_i \in \{-1, 1\}$
- The goal is to find a hyperplane that divides $S$ in these two classes

Mathematically:

$S$ is separable if $\exists w \in \mathbb{R}^n, b \in \mathbb{R}$

$$y_i (w \cdot x_i + b) \geq 1$$

Separating hyperplanes:

$$w \cdot x + b = 0$$

Closest point:

$$y_i d_i = \frac{1}{w}$$
Support Vector Machines

- Optimal separating hyperplane maximizes $\frac{1}{w}$

Problem 1:
Minimize $\frac{1}{2} w \cdot w$
Subject to $y_i(w \cdot x_i + b) \geq 1, \ i = 1,2,\ldots, N$
Final Exam

• Closed book
• One cheat sheet
  – Single piece of paper any size, handwritten, no photocopying, no physical cut & paste.
• What to study
  – Basically material presented in class, and supporting material from text
  – If it was in text, but NEVER mentioned in class, it is very unlikely to be on the exam
• Question style:
  – Short answer
  – Some longer problems to be worked out.