Multiclass object recognition
Sharing parts and transfer learning

Sharat Chikkerur
Outline

- Historical perspective and motivation

- Discriminative approach
  - A. Torralba, K. Murphy, W. Freeman, Sharing visual features for multiclass and multiview object detection, IEEE PAMI 2007

- Bayesian approach
  - (Prelude) R. Fergus, P. Perona, A. Zisserman, Object recognition by unsupervised scale-invariant learning, CVPR 03
  - L. Fei-Fei, R. Fergus and P. Perona. One-Shot learning of object categories. PAMI, 2006
Perspective: Template vs parts

- **Dense representation**
- Useful for rigid objects
- Less robust
- Appearance only
- Objects share features

- **Sparse representation**
- Rigid and articulate objects
- More robust
- Appearance and shape
- Objects share parts

Summary: Part-based representation make more sense!
Motivation: Sharing parts
**Benefits**

- Learning is faster
  - Features are reused
  - Time complexity $\sim O(\log n)$ instead of $O(n)$
- Better generalization
  - Individual parts share training data across classes
  - Robust to inter-class variation

**Challenges**

- Identity of shared parts/classes unknown
- Sharing may not follow tree structure
- Exhaustive search $\sim O(2^P)$
How do you share parts?

- **Create** a universal dictionary of parts
  - Serre et al 07 (HMAX), Ke and Sukhtankar (PCA SIFT)
- **Learn** the shared dictionary of parts
  - Discriminative
    - Embed sharing into optimization
      - Discriminative dictionary (Marial et al 08)
      - Joint boosting (Torralba et al 07)
  - Generative
    - Use unlabeled data to learn **prior**
      - Constellation model (Fei-Fei et al 06)
Discriminative approach

A. Torralba, K. Murphy, W. Freeman, Sharing visual features for multiclass and multiview object detection, IEEE PAMI 2007
Recap: Part representation

Feature (Appearance + Position)
Recap: Boosting

- An additive model for combining weak classifiers

\[ H(v) = \sum_{m=1}^{M} h_m(v) \quad J = E\left[e^{-zH(v)}\right] \]

- Weak classifier: \( h_m(v_i) = a\delta(v_i^f > \theta) + b\delta(v_i^f \leq \theta) \)

- Algorithm:

  1) Initialize the weights \( w_i = 1 \) and set \( H(v_i) = 0, i = 1..N \).
  2) Repeat for \( m = 1, 2, \ldots, M \)
     a) Fit stump: \( h_m(v_i) = a\delta(v_i^f > \theta) + b\delta(v_i^f \leq \theta) \)
     b) Update class estimates for examples \( i = 1, \ldots, N \):
        \( H(v_i) := H(v_i) + h_m(v_i) \)
     c) Update weights for examples \( i = 1, \ldots, N \):
        \( w_i := w_i e^{-z_i h_m(v_i)} \)
Choosing a weak classifier

1) Initialize the weights \( w_i = 1 \) and set \( H(v_i) = 0, \ i = 1..N \).
2) Repeat for \( m = 1, 2, \ldots, M \)
   a) Fit stump: \( h_m(v_i) = a \delta(v_i^f > \theta) + b \delta(v_i^f \leq \theta) \)
   b) Update class estimates for examples \( i = 1, \ldots, N \):
      \( H(v_i) := H(v_i) + h_m(v_i) \)
   c) Update weights for examples \( i = 1, \ldots, N \):
      \( w_i := w_i e^{-z_i h_m(v_i)} \)

- For each feature
  - Evaluate the weighted error \( J_{wse} = \sum_{i=1}^{N} w_i (z_i - h_m(v_i))^2 \)
  - Pick the feature with minimum error
Joint Boosting

- An additive model that jointly optimizes for all classes

\[ H(v, c) = \sum_{m=1}^{M} h_m(v, c) \]

\[ J = \sum_{c=1}^{C} E \left[ e^{-z^c H(v, c)} \right] \]

- Weak classifier:

\[ h_m(v, c) = \begin{cases} 
    a_S & \text{if } v_i^f > \theta \text{ and } c \in S(n) \\
    b_S & \text{if } v_i^f \leq \theta \text{ and } c \in S(n) \\
    k_S^c & \text{if } c \notin S(n) 
\end{cases} \]

\[
\begin{pmatrix}
H(v,1) \\
H(v,2) \\
H(v,3) \\
H(v,4) \\
H(v,5)
\end{pmatrix} =
\begin{pmatrix}
h_1(v,1) \\
h_1(v,2) \\
h_1(v,3) \\
h_1(v,4) \\
h_1(v,5)
\end{pmatrix} +
\begin{pmatrix}
h_2(v,1) \\
h_2(v,2) \\
h_2(v,3) \\
h_2(v,4) \\
h_2(v,5)
\end{pmatrix} + \cdots
1) Initialize the weights $w_i^c = 1$ and set $H(v_i, c) = 0$, $i = 1..N$, $c = 1..C$.
2) Repeat for $m = 1, 2, \ldots, M$
   a) Repeat for $n = 1, 2, \ldots, 2^C - 1$
      i) Fit shared stump:

      $$
      h_m^n(v_i, c) = \begin{cases} 
      a_S & \text{if } v_i^f > \theta \text{ and } c \in S(n) \\
      b_S & \text{if } v_i^f \leq \theta \text{ and } c \in S(n) \\
      k_c & \text{if } c \notin S(n) 
      \end{cases}
      $$

      ii) Evaluate error

      $$
      J_{wse}(n) = \sum_{c=1}^{C} \sum_{i=1}^{N} w_i^c (z_i^c - h_m^n(v_i, c))^2
      $$

   b) Find best subset: $n^* = \arg \min_n J_{wse}(n)$.
   c) Update the class estimates

      $$
      H(v_i, c) := H(v_i, c) + h_m^{n^*}(v_i, c)
      $$

   d) Update the weights

      $$
      w_i^c := w_i^c e^{-z_i^c h_m^{n^*}(v_i, c)}
      $$
Example

Joint boosting

Independent

Feature sharing
Greedy approach

- Exhaustive search of all classes \( \sim O(2^C) \)
- Greedy approach
  - Select the class with best reduction in error
  - Insert next class with lowest error
  - Continue till all classes are selected
  - Select the best member from the set
  - Complexity \( \sim O(C^2) \)
Typical behavior

- Independent features/ pairs $\sim O(N)$
- Shared features $\sim O(\log N)$
Application: Object categorization

- Data: 21 object categories
- 2000 candidate features (extracted by random sampling)
- 50 training examples per category
Object categorization: Performance
Object categorization: Shared features

| screen | poster | car frontal | chair | keyboard | bottle | car side | mouse | mouse pad | can | trashcan | head | person | mug | speaker | traffic light | one way Sign | do not enter | stop Sign | light | cpu |
|--------|--------|-------------|-------|----------|--------|----------|-------|-----------|-----|-----------|------|---------|-----|---------|---------------|-------------|-------------|-----------|-------|------|-----|

The table represents a matrix of shared features among different objects.
Summary

- Joint boosting allows learning of shared parts (even non-tree structures)
- Learning time reduces from $O(N)$ to $O(\log N)$
  - Allows scaling to large number of categories
- Reduces training sample size (per class)
- Useful for multi-class as well as multi-view recognition
- Wish-list?
  - Automatic scale selection for features
  - Handling occlusion
Bayesian approach
Bayes 101: Coin tossing

- **MLE**
  - Let p: probability of heads
  - Data: we observed H heads and T tails.
  - Inference: What is the chance of next head?
    - \( P(\text{Head}) = p = H/(H+T) \)

- **Bayesian**
  - Let p: probability of heads (unknown!), \( p \sim f(p) \)
  - \( P(\text{Head}) = \int P(\text{Head}|p) \ f(p) \ \text{dp} \)
  - Data: we observed H heads and T tails
  - \( p \sim f(p|D) \), still not fixed!
    - \( P(\text{Head}|D) = \int P(\text{Head}|p) \ f(p|D) \ \text{dp} \)
Learning parameters: Conjugate priors

- Conjugate prior: Functional form of the prior and posterior distribution are identical
- With no data, we assume that the coin is likely to be fair
- Uncertainty based on hyperparameters
- After we observe data
  - D = (H-heads, T-tails)
  - the uncertainty in h is altered

\[ p(h) \sim B(a,b) \]

\[ p(h|D) \sim B(a+H,b+T) \]
Transfer learning

- **Discriminative**
  - Given data: Learn shared parameters
  - New data: Use all old parameters (+ new)

- **Bayesian**
  - Given data: Learn priors (“assumptions”)
  - New data: Update priors
A prelude
Object class recognition by unsupervised scale-invariant learning
Constellation model

Torralba et al. ~100 parts

Motorbike shape model

Fergus et al < 10 parts
Generative model/Bayesian detection

- Generative model for shape, appearance and scale

\[
p(X, S, A | \theta) = \sum_{h \in H} p(X, S, A | h | \theta) = \sum_{h \in H} p(A | X, S, h, \theta) p(X | S, h, \theta) p(S | h, \theta) p(h | \theta)
\]

- \( H \) encodes the mapping from part (P) to interest point (N)
- Example: \( N=10, P=4, h=[0\ 3\ 4\ 5] \) or \( h=[3\ 1\ 2\ 10] \). \(|h| \sim O(N^P)\)
- Bayesian detection:

\[
R = \frac{p(\text{Object} | X, S, A)}{p(\text{No object} | X, S, A)} \\
= \frac{p(X, S, A | \text{Object}) p(\text{Object})}{p(X, S, A | \text{No object}) p(\text{No object})} \\
\approx \frac{p(X, S, A | \theta) p(\text{Object})}{p(X, S, A | \theta_{bg}) p(\text{No object})}
\]
Factorization

- Appearance

\[
p(A|X, S, h, \theta) \quad \frac{p(A|X, S, h, \theta)}{p(A|X, S, h, \theta_{bg})} = \prod_{p=1}^{P} \left( \frac{G(A(h_p)|c_p, V_p)}{G(A(h_p)|c_{bg}, V_{bg})} \right)^{d_p}
\]

- Shape

\[
p(X|S, h, \theta) \quad \frac{p(X|S, h, \theta)}{p(X|S, h, \theta_{bg})} = G(X(h)|\mu, \Sigma) \quad \alpha^f
\]

- Scale

\[
p(S|h, \theta) \quad \frac{p(S|h, \theta)}{p(S|h, \theta_{bg})} = \prod_{p=1}^{P} G(S(h_p)|t_p, U_p)^{d_p} r^f
\]

- Occlusion

\[
p(h|\theta) \quad \frac{p(h|\theta)}{p(h|\theta_{bg})} = \frac{p_{Poiss}(n|M)}{p_{Poiss}(N|M)} \frac{1}{nC_r(N, f)} p(d|\theta)
\]
Representation and learning

- Position/Shape
  - Candidate part locations are obtained using Kadir-Brady interest point detector

- Appearance
  - Modeled using 11x11 pixels around the interest point (PCA used for reducing dimension)

- Learning (EM)

\[ \theta = \{ \mu, \Sigma, c, V, M, p(d|\theta), t, U \} \]

\[ \hat{\theta}_{ML} = \arg \max_{\theta} p(X, \tilde{S}, A | \theta) \]
Example: faces
Example: motorbikes
## Comparison (Caltech 4)

- Models are class-specific

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Total size of dataset</th>
<th>Object width (pixels)</th>
<th>Motorbike model</th>
<th>Face model</th>
<th>Airplane model</th>
<th>Cat model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motorbikes</td>
<td>800</td>
<td>200</td>
<td>92.5</td>
<td>50</td>
<td>51</td>
<td>56</td>
</tr>
<tr>
<td>Faces</td>
<td>435</td>
<td>300</td>
<td>33</td>
<td>96.4</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>Airplanes</td>
<td>800</td>
<td>300</td>
<td>64</td>
<td>63</td>
<td>90.2</td>
<td>53</td>
</tr>
<tr>
<td>Spotted Cats</td>
<td>200</td>
<td>80</td>
<td>48</td>
<td>44</td>
<td>51</td>
<td>90.0</td>
</tr>
</tbody>
</table>

- Models are robust to scale variation

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Total size of dataset</th>
<th>Object size range (pixels)</th>
<th>Pre-scaled performance</th>
<th>Unscaled performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motorbikes</td>
<td>800</td>
<td>200-480</td>
<td>95.0</td>
<td>93.3</td>
</tr>
<tr>
<td>Airplanes</td>
<td>800</td>
<td>200-500</td>
<td>94.0</td>
<td>93.0</td>
</tr>
<tr>
<td>Cars (Rear)</td>
<td>800</td>
<td>100-550</td>
<td>84.8</td>
<td>90.3</td>
</tr>
</tbody>
</table>
Bayesian approach

L. Fei-Fei, R. Fergus and P. Perona. One-Shot learning of object categories. PAMI, 2006
Bayesian approach

- Fergus et al \((I = X, S, A)\)

\[
R = \frac{p(\text{Object} | X, S, A)}{p(\text{No object} | X, S, A)}
\]

\[
= \frac{p(X, S, A | \text{Object}) p(\text{Object})}{p(X, S, A | \text{No object}) p(\text{No object})}
\]

MLE approximation

\[
\approx \frac{p(X, S, A | \theta) p(\text{Object})}{p(X, S, A | \theta_{bg}) p(\text{No object})}
\]

- Fei-Fei et al.

\[
R \propto \frac{\int p(X, A | \theta, \mathcal{O}_{fg}) p(\theta | x_t, A_t, \mathcal{O}_{fg}) d\theta}{\int p(X, A | \theta_{bg}, \mathcal{O}_{bg}) p(\theta_{bg} | x_t, A_t, \mathcal{O}_{bg}) d\theta_{bg}}
\]

Parameter integration
Generative model for shape and appearance

- Foreground object (integrate over all hypotheses)

\[ p(\mathcal{X}, A | \theta) = \sum_{w=1}^{\Omega} \sum_{h \in H} p(\mathcal{X}, A | h, u | \theta) \]

\[ = \sum_{w=1}^{\Omega} p(w | \pi) \sum_{h \in H} p(A | h, \theta^A_w) p(\mathcal{X} | h, \theta^X_w) p(h | \theta_w) \]

- In the paper, \( \Omega=1 \) (\( \Omega>1 \) can handle pose variation)

- Background (has a single null hypothesis)

\[ p(\mathcal{X}, A | \theta_{bg}) = p(\mathcal{X}, A, h_0 | \theta_{bg}) \]

\[ = p(A | h_0, \theta^A_{bg}) p(\mathcal{X} | h_0, \theta^X_{bg}) p(h_0 | \theta_{bg}) \]
Factorization

- **Appearance**

\[
\frac{p(A|X, S, h, \theta)}{p(A|X, S, h, \theta_{bg})} = \prod_{p=1}^{P} \left( \frac{G(A(h_p)|c_p, V_p)}{G(A(h_p)|c_{bg}, V_{bg})} \right)^{d_p}
\]

Fergus et al.

\[
\frac{p(A|h, \theta^A_w)}{p(A|h_0, \theta^A_{bg})} = \prod_{p=1}^{P} \frac{G(A(h_p)|\mu^A_{p,w}, \Gamma^A_{p,w})}{G(A(h_p)|\mu^A_{bg}, \Gamma^A_{bg})}
\]

Fei Fei et al.

- **Shape**

\[
\frac{p(X|S, h, \theta)}{p(X|S, h, \theta_{bg})} = G(X(h)|\mu, \Sigma) \alpha^f
\]

Fergus et al.

\[
\frac{p(X|h, \theta^X_w)}{p(X|h_0, \theta^X_{bg})} = \alpha^{P-1} G(X(h)|\mu^X_w, \Gamma^X_w)
\]

Fei Fei et al.

- **Scale and occlusion are not modeled**
Comparison

- **MLE**
  \[ \theta^* = \theta^{ML} = \arg \max_{\theta} p(\mathbf{x}_t, A_t | \theta) \]

- **MAP**
  \[ \theta^* = \theta^{MAP} = \arg \max_{\theta} p(\mathbf{x}_t, A_t | \theta)p(\theta) \]

- **Bayesian**
  \[ \bar{\theta} = \{ \pi, \mu^\mathbf{x}, \mu^\mathbf{A}, \Gamma^\mathbf{x}, \Gamma^\mathbf{A} \} \]
  \[ p(\theta) \rightarrow p(\theta | \mathbf{x}_t, A_t, \mathcal{O}_{fg}) \]
Conjugate priors

- **Parameters**
  \[ \tilde{\theta} = \{ \pi, \mu^x, \mu^A, \Gamma^x, \Gamma^A \} \]

- **Priors**
  \[
p(\theta | x_t, A_t) = p(\pi) \prod_\omega p(\mu^x_\omega | \Gamma^x_\omega) p(\Gamma^x_\omega) p(\mu^A_\omega | \Gamma^A_\omega) p(\Gamma^A_\omega)
\]

  \[
p(\pi) = \text{Dir}(\lambda_\omega I_\Omega) \quad p(\Gamma^x_\omega) = \mathcal{W}(\Gamma^x_\omega | a^x_\omega, B^x_\omega) \quad p(\mu^x_\omega | \Gamma^x_\omega) = \mathcal{G}(\mu^x_\omega | m^x_\omega, \beta^x_\omega \Gamma^x_\omega)
\]

  Dirichlet \hspace{1cm} \text{Wishart} \hspace{1cm} \text{Normal}

- **Hyper-parameters**
  \[ \{ \lambda_\omega, a_\omega, B_\omega, m_\omega, \beta_\omega \} \]

- **Closed form solution**
  \[
p(x, A | x_t, A_t, O_{fg}) = \int p(x, A | \theta) p(\theta | x_t, A_t, O_{fg}) d\theta
\]

  \[
  \sum_{\omega=1}^{\Omega} \sum_{h=1}^{\vert H \vert} \tilde{\pi}_\omega \mathcal{S}(x_h | g^x_\omega, m^x_\omega, \Lambda^x_\omega) \mathcal{S}(A_h | g^A_\omega, m^A_\omega, \Lambda^A_\omega),
\]

  where \( g_\omega = a_\omega + 1 - d \) and \( \Lambda_\omega = \frac{\beta_\omega + 1}{\beta_\omega g_\omega} B_\omega \) and \( \tilde{\pi}_\omega = \frac{\lambda_\omega}{\sum_{\omega'} \lambda_{\omega'}} \)
Learning

- $p(x|\theta) = \sum p(x,h|\theta)p(h|\theta)$, $h$-unknown, but 'convenient'

- **Regular EM**
  - E-step: Estimate $p(h|x,\theta^n)$, $Q(\theta) = E_h \{ \log(p(x,h|\theta)) | h \}$
  - (Usually available in closed form)
  - M-step: $\Theta^{n+1} = \text{argmax} \ Q(\theta)$

- **Variational (EM)**
  - Getting $p(h|x,\theta^n)$ is hard-no closed form
  - $p(h|x,\theta^n) \sim q(x)$, approximate the posterior
Performance: Caltech 4
Caltech 4 (cont.)
Performance: Caltech 101

- Performance: \( (a) 10.4\%, 13.9\%, 17.7\% \) with 3, 6, 15 training example
- State-of-the-art: > 60\%
## Comparison

<table>
<thead>
<tr>
<th>Authors</th>
<th>Categories</th>
<th># Categories</th>
<th># Training images</th>
<th>Framework</th>
<th>Hand alignment</th>
<th>Segmented</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fergus et al. [14]</td>
<td>Assorted</td>
<td>6</td>
<td>&gt; 100</td>
<td>Gen.</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Weber et al. [40]</td>
<td>Cars, Faces</td>
<td>2</td>
<td>&gt; 100</td>
<td>Gen. + Disc.</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Viola &amp; Jones [37]</td>
<td>Faces</td>
<td>1</td>
<td>~ 10,000</td>
<td>Disc.</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Schneiderman &amp; Kanade [35]</td>
<td>Cars</td>
<td>1</td>
<td>2,000</td>
<td>Disc.</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>Rowley et al. [33]</td>
<td>Cars</td>
<td>1</td>
<td>500</td>
<td>Disc.</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>LeCun et al. [25]</td>
<td>Digits</td>
<td>10</td>
<td>60,000</td>
<td>Disc.</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>LeCun et al. [26]</td>
<td>Assorted</td>
<td>5</td>
<td>~300,000</td>
<td>Disc.</td>
<td>Y</td>
<td>N</td>
</tr>
</tbody>
</table>
Summary

- Transfer learning in a Bayesian setting
- Recipe = learning priors on given data + updating priors on new data
- Good results with just 1~5 training examples (compared to MLE approaches)
- Learning is hard (computationally)
- Wishlist
  - Handling multiple objects within the image.
Thank You!