Part-based and local feature models for generic object recognition

May 28th, 2015

Yong Jae Lee
UC Davis
Announcements

• PS2 grades up on SmartSite
• PS2 stats:
  – Mean: 80.15
  – Standard Dev: 22.77
• Vote on piazza for last lecture content
Support Vector Machines (SVMs)

- Discriminative classifier based on optimal separating line (for 2d case)
- Maximize the *margin* between the positive and negative training examples
SVMs for recognition

1. Define your representation for each example.

2. Select a kernel function.

3. Compute pairwise kernel values between labeled examples.

4. Use this “kernel matrix” to solve for SVM support vectors & weights.

5. To classify a new example: compute kernel values between new input and support vectors, apply weights, check sign of output.
Example: learning gender with SVMs

Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Moghaddam and Yang, Face & Gesture 2000.
Face alignment processing

Processed faces

Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.
Learning gender with SVMs

- Training examples:
  - 1044 males
  - 713 females

- Experiment with various kernels, select Gaussian RBF

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

Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.
## Classifier Performance

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Error Rate</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Overall</td>
<td>Male</td>
<td>Female</td>
</tr>
<tr>
<td>SVM with RBF kernel</td>
<td>3.38%</td>
<td>2.05%</td>
<td>4.79%</td>
</tr>
<tr>
<td>SVM with cubic polynomial kernel</td>
<td>4.88%</td>
<td>4.21%</td>
<td>5.59%</td>
</tr>
<tr>
<td>Large Ensemble of RBF</td>
<td>5.54%</td>
<td>4.59%</td>
<td>6.55%</td>
</tr>
<tr>
<td>Classical RBF</td>
<td>7.79%</td>
<td>6.89%</td>
<td>8.75%</td>
</tr>
<tr>
<td>Quadratic classifier</td>
<td>10.63%</td>
<td>9.44%</td>
<td>11.88%</td>
</tr>
<tr>
<td>Fisher linear discriminant</td>
<td>13.03%</td>
<td>12.31%</td>
<td>13.78%</td>
</tr>
<tr>
<td>Nearest neighbor</td>
<td>27.16%</td>
<td>26.53%</td>
<td>28.04%</td>
</tr>
<tr>
<td>Linear classifier</td>
<td>58.95%</td>
<td>58.47%</td>
<td>59.45%</td>
</tr>
</tbody>
</table>

Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.
Gender perception experiment: How well can humans do?

• Subjects:
  – 30 people (22 male, 8 female)
  – Ages mid-20’s to mid-40’s

• Test data:
  – 254 face images
  – Low res (6 males, 4 females)
  – High res versions

• Task:
  – Classify as male or female, forced choice
  – No time limit

Moghaddam and Yang, Face & Gesture 2000.
Gender perception experiment:
How well can humans do?

Stimuli →

Results →

<table>
<thead>
<tr>
<th></th>
<th>High-Res</th>
<th>Low-Res</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>6.54%</td>
<td>30.7%</td>
</tr>
</tbody>
</table>

σ = 3.7%
• SVMs performed better than any single human test subject, at either resolution.
Hardest examples for humans

Top five human misclassifications

Moghaddam and Yang, Face & Gesture 2000.
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?
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: Pros and cons

• Pros
  – Many publicly available SVM packages: [http://www.kernel-machines.org/software](http://www.kernel-machines.org/software)
  – Kernel-based framework is very powerful, flexible
  – Often a sparse set of support vectors – compact at test time
  – Work very well in practice, even with very small training sample sizes

• Cons
  – No “direct” multi-class SVM, must combine two-class SVMs
  – Can be tricky to select best kernel function for a problem
  – Computation, memory
    • During training time, must compute matrix of kernel values for every pair of examples
    • Learning can take a very long time for large-scale problems
Previously

• Discriminative classifiers
  – Boosting
  – Nearest neighbors
  – Support vector machines

• Useful for object recognition when combined with “window-based” or holistic appearance descriptors
Global window-based appearance representations

- These examples are truly global; each pixel in the window contributes to the representation.
- Classifier can account for relative relevance...
- *When might this not be ideal?*
Part-based and local feature models for recognition

Main idea:

Rather than a representation based on holistic appearance, decompose the image into:

- local parts or patches, and
- their relative spatial relationships
Part-based and local feature models for recognition

We’ll look at three forms:
1. **Bag of words** (no geometry)
2. **Implicit shape model** (star graph for spatial model)
3. **Constellation model** (fully connected graph for spatial model)
Bag-of-words model

• Summarize entire image based on its distribution (histogram) of word occurrences.
  – Total freedom on spatial positions, relative geometry.
  – Vector representation easily usable by most classifiers.
Bag-of-words model

Our in-house database contains 1776 images in seven classes: faces, buildings, trees, cars, phones, bikes and books. Fig. 2 shows some examples from this dataset.

Csurka et al. Visual Categorization with Bags of Keypoints, 2004
Words as parts

All local features

Local features from two selected clusters occurring in this image

Csurka et al. 2004
Naïve Bayes model for classification

\[ c^* = \arg \max_c p(c \mid w) \propto p(c) p(w \mid c) = p(c) \prod_{n=1}^{N} p(w_n \mid c) \]

Object class decision
Prior prob. of the object classes
Image likelihood given the class

What assumptions does the model make, and what are their significance?
## Confusion matrix

<table>
<thead>
<tr>
<th>True classes</th>
<th>faces</th>
<th>buildings</th>
<th>trees</th>
<th>cars</th>
<th>phones</th>
<th>bikes</th>
<th>books</th>
</tr>
</thead>
<tbody>
<tr>
<td>faces</td>
<td>76</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>13</td>
</tr>
<tr>
<td>buildings</td>
<td>2</td>
<td>44</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>trees</td>
<td>3</td>
<td>2</td>
<td>80</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>cars</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>75</td>
<td>3</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>phones</td>
<td>9</td>
<td>15</td>
<td>1</td>
<td>16</td>
<td>70</td>
<td>14</td>
<td>11</td>
</tr>
<tr>
<td>bikes</td>
<td>2</td>
<td>15</td>
<td>12</td>
<td>0</td>
<td>8</td>
<td>73</td>
<td>0</td>
</tr>
<tr>
<td>books</td>
<td>4</td>
<td>19</td>
<td>0</td>
<td>6</td>
<td>7</td>
<td>2</td>
<td>69</td>
</tr>
</tbody>
</table>

Example bag of words + Naïve Bayes classification results for generic categorization of objects
Clutter…or context?
Sampling strategies

Specific object

Category

Kristen Grauman
Sampling strategies

Sparse, at interest points

Dense, uniformly

Randomly

- To find specific, textured objects, sparse sampling from interest points more reliable.
- Multiple complementary interest operators offer more image coverage.
- For object categorization, dense sampling offers better coverage.

[See Nowak, Jurie & Triggs, ECCV 2006]
Local feature correspondence for generic object categories

\[ X = \{ \vec{x}_1, \ldots, \vec{x}_m \} \quad Y = \{ \vec{y}_1, \ldots, \vec{y}_n \} \]
Local feature correspondence for generic object categories

• Comparing bags of words histograms coarsely reflects agreement between local “parts” (patches, words).
• *But* choice of quantization directly determines what we consider to be similar…
Partially matching sets of features

X = \{\bar{x}_1, \ldots, \bar{x}_m\} \quad Y = \{\bar{y}_1, \ldots, \bar{y}_n\}

\min_{\pi: X \to Y} \sum_{x_i \in X} \|x_i - \pi(x_i)\|

Optimal match: \(O(m^3)\)
Greedy match: \(O(m^2 \log m)\)
Pyramid match: \(O(m)\) \((m=\text{num pts})\)

Previous work: Indyk & Thaper, Bartal, Charikar, Agarwal & Varadarajan, ...
Pyramid match: main idea

Feature space partitions serve to “match” the local descriptors within successively wider regions.

\[ \mathcal{R}^d \]

\[ X = \{ \bar{x}_1, \ldots, \bar{x}_m \} \]

\[ Y = \{ \bar{y}_1, \ldots, \bar{y}_n \} \]

[Grauman & Darrell, ICCV 2005]
Pyramid match: main idea

$\mathbb{R}^d$

$X = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$  $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$

$H_X$  $H_Y$

$\mathcal{I}(H_X, H_Y) = \sum_j \min(H_X(j), H_Y(j))$

= 3

Histogram intersection counts number of possible matches at a given partitioning.

[Grauman & Darrell, ICCV 2005]
Pyramid match kernel

\[ K_\Delta(X, Y) = \sum_{i=0}^{L} 2^{-i} \mathcal{I} \left( H_X^{(i)}, H_Y^{(i)} \right) - \mathcal{I} \left( H_X^{(i-1)}, H_Y^{(i-1)} \right) \]

- For similarity, weights inversely proportional to bin size (or may be learned)
- Normalize these kernel values to avoid favoring large sets

[Grumman & Darrell, ICCV 2005]
Pyramid match kernel

Optimal match: $O(m^3)$
Pyramid match: $O(mL)$

$X = \{ \tilde{x}_1, \ldots, \tilde{x}_m \}$  $Y = \{ \tilde{y}_1, \ldots, \tilde{y}_n \}$

[Grauman & Darrell, ICCV 2005]
Highlights of the pyramid match

- Linear time complexity
- Formal bounds on expected error
- Mercer kernel
- Data-driven partitions allow accurate matches even in high-dim. feature spaces
- Strong performance on benchmark object recognition datasets
Example recognition results:
Caltech-101 dataset

- 101 categories
- 40-800 images per class
Recognition results:
Caltech-101 dataset

- Jain, Huynh, & Grauman (2007)
- Grauman & Darrell (2005)
- Wang et al. (2006)
- Serre et al. (2007)
- Frome et al. (2007)
- Zhang et al. (2006)
- Berg (2005)
- Berg et al. (2005)
- Holub et al. (2005)
Unordered sets of local features: **No** spatial layout preserved!

Too much?  
Too little?
Spatial pyramid match

- Make a pyramid of bag-of-words histograms.
- Provides some loose (global) spatial layout information

\[ K^L(X, Y) = \sum_{m=1}^{M} \kappa^L(X_m, Y_m) \]

Sum over PMKs computed in *image coordinate* space, one per word.

[Lazebnik, Schmid & Ponce, CVPR 2006]
Spatial pyramid match

Captures scene categories well---texture-like patterns but with some variability in the positions of all the local pieces.
Spatial pyramid match

Captures scene categories well---texture-like patterns but with some variability in the positions of all the local pieces.

<table>
<thead>
<tr>
<th>Level</th>
<th>Single-level</th>
<th>Pyramid</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (1 × 1)</td>
<td>72.2 ±0.6</td>
<td></td>
</tr>
<tr>
<td>1 (2 × 2)</td>
<td>77.9 ±0.6</td>
<td>79.0 ±0.5</td>
</tr>
<tr>
<td>2 (4 × 4)</td>
<td>79.4 ±0.3</td>
<td><strong>81.1</strong> ±0.3</td>
</tr>
<tr>
<td>3 (8 × 8)</td>
<td>77.2 ±0.4</td>
<td>80.7 ±0.3</td>
</tr>
</tbody>
</table>
Part-based and local feature models for recognition

We’ll look at three forms:

1. **Bag of words** (no geometry)
2. **Implicit shape model** (star graph for spatial model)
3. **Constellation model** (fully connected graph for spatial model)
Shape representation in part-based models

“Star” shape model

- e.g. implicit shape model
- Parts mutually independent

N image features, P parts in the model
Implicit shape models

- Visual vocabulary is used to index votes for object position [a visual word = “part”]

B. Leibe, A. Leonardis, and B. Schiele, *Combined Object Categorization and Segmentation with an Implicit Shape Model*, ECCV Workshop on Statistical Learning in Computer Vision 2004
Implicit shape models

- Visual vocabulary is used to index votes for object position [a visual word = “part”]

B. Leibe, A. Leonardis, and B. Schiele, *Combined Object Categorization and Segmentation with an Implicit Shape Model*, ECCV Workshop on Statistical Learning in Computer Vision 2004
Implicit shape models: Training

1. Build vocabulary of patches around extracted interest points using clustering
Implicit shape models: Training

1. Build vocabulary of patches around extracted interest points using clustering
2. Map the patch around each interest point to closest word
Implicit shape models: Training

1. Build vocabulary of patches around extracted interest points using clustering
2. Map the patch around each interest point to closest word
3. For each word, store all positions it was found, relative to object center
Implicit shape models: Testing

1. Given new test image, extract patches, match to vocabulary words
2. Cast votes for possible positions of object center
3. Search for maxima in voting space
4. (Extract weighted segmentation mask based on stored masks for the codebook occurrences)

*What is the dimension of the Hough space?*
Implicit shape models: Testing
Example: Results on Cows

Original image
Example: Results on Cows

Interest points
Example: Results on Cows

Matched patches
Example: Results on Cows
Example: Results on Cows

1st hypothesis
Example: Results on Cows

2nd hypothesis
Example: Results on Cows

3rd hypothesis
Detection Results

• Qualitative Performance
  - Recognizes different kinds of objects
  - Robust to clutter, occlusion, noise, low contrast
Shape representation in part-based models

“Star” shape model

- e.g. implicit shape model
- Parts mutually independent

Fully connected constellation model

- e.g. Constellation Model
- Parts fully connected

N image features, P parts in the model
Probabilistic constellation model

\[ P(\text{image} \mid \text{object}) = P(\text{appearance, shape} \mid \text{object}) \]

Part descriptors

Part locations

Candidate parts

Source: Lana Lazebnik
Probabilistic constellation model

\[ P(\text{image} \mid \text{object}) = P(\text{appearance, shape} \mid \text{object}) \]
Probabilistic constellation model

\[
P(\text{image} \mid \text{object}) = P(\text{appearance}, \text{shape} \mid \text{object})
\]

\[
= \max_h P(\text{appearance} \mid h, \text{object}) p(\text{shape} \mid h, \text{object}) p(h \mid \text{object})
\]

h: assignment of features to parts
Example results from constellation model: data from four categories

- Faces
- Motorbikes
- Airplanes
- Spotted cats

Slide from Li Fei-Fei http://www.vision.caltech.edu/feifeili/Resume.htm
Face model

Appearance: 10 patches closest to mean for each part

Fergus et al. CVPR 2003
Face model

Recognition results

Appearance: 10 patches closest to mean for each part

Test images: size of circles indicates score of hypothesis
Motorbike model

Appearance: 10 patches closest to mean for each part

Recognition results
Appearance: 10 patches closest to mean for each part

Spotted cat model

Recognition results
Shape representation in part-based models

“Star” shape model

- e.g. implicit shape model
- Parts mutually independent
- Recognition complexity: $O(NP)$
- Method: Gen. Hough Transform

N image features, P parts in the model

Fully connected constellation model

- e.g. Constellation Model
- Parts fully connected
- Recognition complexity: $O(N^P)$
- Method: Exhaustive search

Slide credit: Rob Fergus
Summary: part-based and local feature models for generic object recognition

• Histograms of visual words to capture global or local layout in the bag-of-words framework
  – Pyramid match kernels
  – Powerful in practice for image recognition

• Part-based models encode category’s part appearance together with 2d layout and allow detection within cluttered image
  – “implicit shape model”: shape based on layout of all parts relative to a reference part; Generalized Hough for detection
  – “constellation model”: explicitly model mutual spatial layout between all pairs of parts; exhaustive search for best fit of features to parts
Recognition models

Instances: recognition by alignment

Categories: Holistic appearance models (and sliding window detection)

Categories: Local feature and part-based models
Questions?

See you Tuesday!