Pyramids of Features
For Categorization

Presented by Greg Griffin
Project Partner Will Coulter
Pyramids of Features For Categorization

Presented by Greg Griffin
Project Partner Will Coulter
Buckets of Features For Categorization

Presented by Greg Griffin
Project Partner Will Coulter
This talk is mostly about this paper:

Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories

Svetlana Lazebnik
slazebni@uiuc.edu
Beckman Institute
University of Illinois

Cordelia Schmid
Cordelia.Schmid@inrialpes.fr
INRIA Rhône-Alpes
Montbonnot, France

Jean Ponce
ponce@cs.uiuc.edu
Ecole Normale Supérieure
Paris, France

With a little bit about benchmarking:

CALTECH 256
Images \quad \rightarrow \quad \text{Features} \quad \rightarrow \quad A \ Number

\[
\begin{align*}
\vec{X}_1 \ldots \vec{X}_{N_x} \\
\vec{Y}_1 \ldots \vec{Y}_{N_y} \\
K(X,Y)
\end{align*}
\]
Images

Features

A Number

\[ \tilde{X}_1 \ldots \tilde{X}_{N_x} \]

\[ \tilde{Y}_1 \ldots \tilde{Y}_{N_y} \]

\[ K(X, Y) \]

“How well do they match”
“Weak Features”
“Weak Features”

(I think?)

“…points whose gradient magnitude in a given direction exceeds a minimum threshold.”

This is just their toy example
They use SIFT descriptors as “Strong Features”.
But you could use any features you want!
Images → Features → A Number
$K^L(X,Y)$
Start By Matching Reds

\[ I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i)) \]

\[ K^L(X,Y) \]
Features

Start By Matching Reds

\[ I^\ell = \sum_{i=1}^{D} \min(H^\ell_X(i), H^\ell_Y(i)) \]

\[ K^L(X,Y) \]
Start By Matching Reds

\[ I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i)) \]

\[ K^L(X,Y) \]
Start By Matching Reds

\[ I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i)) \]

\[ K^L(X,Y) \]
Start By Matching Reds

\[ I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i)) \]

\[ K^L(X,Y) \]
\[ I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i)) \]

\[ K^L(X,Y) \]

Start by matching reds.
$I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i))$

$K^L(X,Y)$

Start By Matching Reds

<table>
<thead>
<tr>
<th>$\ell$</th>
<th>$D$</th>
<th>$I^\ell$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>
Start By Matching Reds

\[ I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i)) \]

\[ K^L(X, Y) \]
Start By Matching Reds

\[ I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i)) \]

\[ \kappa^L(X^1, Y^1) = I^L + \sum_{\ell=0}^{L-1} \frac{1}{2^{L-\ell}} I^\ell = 6 + \frac{8}{2^1} + \frac{10}{2^2} \]

<table>
<thead>
<tr>
<th>\ell</th>
<th>D</th>
<th>I^\ell</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>6</td>
</tr>
</tbody>
</table>

\[ L = 2 \]

\[ K^L(X, Y) \]
Start by matching reds

\[ I^\ell = \sum_{i=1}^{D} \min(H_X^\ell (i), H_Y^\ell (i)) \]

\[ \kappa^L (X^1, Y^1) = I^L + \sum_{\ell=0}^{L-1} \frac{1}{2^{L-\ell}} I^\ell = 6 + \frac{8}{2^1} + \frac{10}{2^2} \]

A compact set of matches is preferable to widely dispersed matches

\[ K^L (X,Y) \]
Start By Matching Reds

\[
I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i))
\]

\[
K^L(X,Y) = I^\ell + \sum_{\ell=0}^{L-1} \frac{1}{2^\ell} I^{\ell+1} = 6 + \frac{8}{2^1} + \frac{10}{2^2}
\]

\[
\begin{array}{cccc}
\ell & D & I^\ell \\
0 & 1 & 10 \\
1 & 4 & 8 \\
2 & 16 & 6 \\
\end{array}
\]

\[L = 2\]
Start By Matching Reds

\[ I^{\ell} = \sum_{i=1}^{D} \min(H_{X}^{\ell}(i), H_{Y}^{\ell}(i)) \]

\[ K^{L}(X, Y) \]
\[ I^\ell = \sum_{i=1}^{D} \min(H_{X}^{\ell}(i), H_{Y}^{\ell}(i)) \]

\[ \kappa^{L}(X^{1}, Y^{1}) = I^{\ell} + \sum_{\ell=0}^{L-1} \frac{1}{2^{L-\ell}} I = 6 + \frac{8}{2^{1}} + \frac{10}{2} \]

\[ \kappa^{L}(X^{1}, Y^{1}) = I^{L} + \sum_{\ell=0}^{L-1} \frac{1}{2^{L-\ell}} (I^{\ell} - I^{\ell+1}) = 5.4 < 7.5 \]

A Sanity Check:
Features X vs. Purely Isotropic Y

\[ K^{L}(X, Y) \]

\[ \begin{array}{ccc}
\ell & D & I^{\ell} \\
0 & 1 & 10 \\
1 & 4 & 5.5 \\
2 & 16 & 1.9 \\
\end{array} \]

\[ L = 2 \]
Start By Matching Reds, Then The Blues, Then…

\[ I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i)) \]

\[ \kappa^r(X^1, Y^1) = I^r + \sum_{\ell=0}^{L-1} \frac{1}{2^{L-\ell}} I = 6 + \frac{8}{2^1} + \frac{10}{2^2} \]

\[ \kappa^L(X^1, Y^1) = I^L + \sum_{\ell=0}^{L-1} \frac{1}{2^{L-\ell}} (I^\ell - I^{\ell+1}) = 6 + \frac{8 - 6}{2} + \frac{10 - 8}{2^2} = 7.5 \]

\[ K^L(X, Y) = \sum_{m=1}^{M} \kappa^L(X^m, Y^m) = 7.5 + \ldots \]

\[ K^L(X, Y) \]
M = 8
\[ I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i)) \]
$I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i))$

<table>
<thead>
<tr>
<th>$\ell$</th>
<th>$D$</th>
<th>$I^\ell$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>8</td>
</tr>
</tbody>
</table>
\[ I^\ell = \sum_{i=1}^{D} \min(H_X^\ell(i), H_Y^\ell(i)) \]
Foreach feature $M=1 \ldots m$

Foreach level $\ell = 0 \ldots L$

Foreach cell $i=1 \ldots D$

\[
I^\ell = \sum_{i=1}^{D} \min(H^\ell_X(i), H^\ell_Y(i))
\]

\[
\kappa^L(X^m, Y^m) = I^L + \sum_{\ell=0}^{L-1} \frac{1}{2^{L-\ell}} (I^\ell - I^{\ell+1})
\]

\[
K^L(X, Y) = \sum_{m=1}^{M} \frac{1}{N^m_X N^m_Y} \kappa^L(X^m, Y^m)
\]
Training Set

15 Categories

100 Images per Category

Test Set

100-300 Images per Category

S^L(X,Y)

3.4
5.6
7.8
1.5
5.4

office
store
coast
street
suburb
Confusion Matrix

Train on 100
Test on 100-300
(per category)
Scene Database

<table>
<thead>
<tr>
<th>( L )</th>
<th>Weak features ((M = 16))</th>
<th>Strong features ((M = 200))</th>
<th>Strong features ((M = 400))</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Single-level</td>
<td>Pyramid</td>
<td>Single-level</td>
</tr>
<tr>
<td>0 ((1 \times 1))</td>
<td>45.3 ±0.5</td>
<td></td>
<td>72.2 ±0.6</td>
</tr>
<tr>
<td>1 ((2 \times 2))</td>
<td>53.6 ±0.3</td>
<td>56.2 ±0.6</td>
<td>77.9 ±0.6</td>
</tr>
<tr>
<td>2 ((4 \times 4))</td>
<td>61.7 ±0.6</td>
<td>64.7 ±0.7</td>
<td>79.4 ±0.3</td>
</tr>
<tr>
<td>3 ((8 \times 8))</td>
<td>63.3 ±0.8</td>
<td>66.8 ±0.6</td>
<td>77.2 ±0.4</td>
</tr>
</tbody>
</table>

Caltech 101

- minaret (97.6%)
- windsor chair (94.6%)
- joshua tree (87.9%)
- okapi (87.8%)
- cougar body (27.6%)
- beaver (27.5%)
- crocodile (25.0%)
- ant (25.0%)
Hypothesis

Pyramid Matching works well when:

• Objects are aligned and localized
  – ie. certain Caltech 101 categories
  – biased by different values of $N_{\text{test}}$

• A few common features that define the category get randomly permuted through many positions, thanks to a large dataset
  – ie. scene database
  – now and then pyramid matching gets lucky
Test

How well will Pyramid matching work?

- Objects are not well aligned, or cluttered
  - Caltech 256 is more challenging in this respect
Scene Database

<table>
<thead>
<tr>
<th>$L$</th>
<th>Weak features ($M = 16$)</th>
<th></th>
<th>Strong features ($M = 200$)</th>
<th></th>
<th>Strong features ($M = 400$)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Single-level</td>
<td>Pyramid</td>
<td>Single-level</td>
<td>Pyramid</td>
<td>Single-level</td>
<td>Pyramid</td>
</tr>
<tr>
<td>0 (1 × 1)</td>
<td>45.3 ± 0.5</td>
<td></td>
<td>72.2 ± 0.6</td>
<td></td>
<td>74.8 ± 0.3</td>
<td></td>
</tr>
<tr>
<td>1 (2 × 2)</td>
<td>53.6 ± 0.3</td>
<td>56.2 ± 0.6</td>
<td>77.9 ± 0.6</td>
<td>79.0 ± 0.5</td>
<td>78.8 ± 0.4</td>
<td>80.1 ± 0.5</td>
</tr>
<tr>
<td>2 (4 × 4)</td>
<td>61.7 ± 0.6</td>
<td>64.7 ± 0.7</td>
<td>79.4 ± 0.3</td>
<td>81.1 ± 0.3</td>
<td>79.7 ± 0.5</td>
<td>81.4 ± 0.5</td>
</tr>
<tr>
<td>3 (8 × 8)</td>
<td>63.3 ± 0.8</td>
<td>66.8 ± 0.6</td>
<td>77.2 ± 0.4</td>
<td>80.7 ± 0.3</td>
<td>77.2 ± 0.5</td>
<td>81.1 ± 0.6</td>
</tr>
</tbody>
</table>

Caltech 101

- minaret (97.6%)
- windsor chair (94.6%)
- joshua tree (87.9%)
- okapi (87.8%)
- cougar body (27.6%)
- beaver (27.5%)
- crocodile (25.0%)
- ant (25.0%)
Cluster Matching

k-means of SIFT positions
More Flexible Than A Grid?

k-means of SIFT positions + color
1. Any cluster can match any cluster
2. Clusters respect duck / water boundaries (sort of)
Tackles Alignment Problem

But… how to match efficiently?
How many clusters? How big?
Summary

Spatial Pyramid Matching is efficient and handles a range of scales, but seems to be sensitive to translation and clutter.

Cluster Matching has the potential to improve translational invariance and tolerance of clutter. But inefficient. Less principled: how many clusters are optimal? How big should they be? No scale invariance.

Can we have the best of both worlds?
Try *Sliding* Spatial Pyramids?

Slide puzzle photo from: http://www.tenfootpolesoftware.com/products/slidepuzzle/