Sparse-Coded Features for Image Retrieval

Tiezheng Ge  Qifa Ke  Jian Sun
University of Science and Technology of China  Microsoft Bing  Microsoft Research Asia
Problem Statement

- Retrieve images representing the same object/scene
Previous Work

• Applying local feature

  - Bag of Visual Word (BoW) [Sivic & Zisserman 03]
    - Large (hierarchical) vocabularies [Nister & Stewenius 03]
    - Hamming embed [Jegou et al 08]
    - Geometry preserving [Zhang et al 11]
    - Query expansion [Chum et al 07, Arandjelovic & Zisserman 12]

  - Aggregation based method
    - VLAD [Jegou et al 10, Arandjelovic & Zisserman 13]
    - Fisher Kernel [Perronnin et al 07, 08, 10, Douze et al 11]
Previous Work

- Aggregation formulation: **Coding & Pooling**

  - Local descriptor \( x \) → Coding → Mapped Vector \( g(x) \) → Pooling → Representation Vector \( G(X) \)

- Try more coding method!
Sparse coding for image search

• For image classification:
  ScSPM [Yang et al 09]   LLC [Wang et al 10]

• Formulation:

  \[
  \min_u |x - uV|^2_2 + \lambda |u|_1
  \]
  s.t.  \(u \succeq 0\)

  Encoding:  \(g(x) = u\)

  Pooling:  \text{max pooling}
Sparse coding for image search

• Slightly differs from classification:
  - Interest/key points, not dense sampled ones.
  - No SPM (shift and rotation)
Sparse coding for image search

- Active
Sparse coding for image search

- **Active**

  \[
  x_1 \quad g(x_1) = [0.1 \quad 0 \quad 0.5]
  \]
  \[
  x_2 \quad g(x_2) = [0.3 \quad 0.6 \quad 0.2]
  \]
  \[
  x_3 \quad g(x_3) = [0 \quad 0.4 \quad 0]
  \]
Sparse coding for image search

- Active
  \[ x_1 \quad g(x_1) = [0.1 \quad 0 \quad 0.5] \]
  \[ x_2 \quad g(x_2) = [0.3 \quad 0.6 \quad 0.2] \]
  \[ x_3 \quad g(x_3) = [0 \quad 0.4 \quad 0] \]

Pooling \( G(X) = [0.3 \quad 0.6 \quad 0.5] \)
Sparse coding for image search

- Active
  \[ x_1 \ g(x_1) = [0.1 \ 0 \ 0.5] \]
  \[ x_2 \ g(x_2) = [0.3 \ 0.6 \ 0.2] \]
  \[ x_3 \ g(x_3) = [0 \ 0.4 \ 0] \]

\[ G(X) = [0.3 \ 0.6 \ 0.5] \]

\[ x_1, x_2 \text{ are active} \]

\[ G(x) \text{ depends solely on } x_1, x_2 \]

Active points:
- \( x_1 \)
- \( x_2 \)
- \( x_3 \)
Sparse coding for image search

• Active

\[
\begin{align*}
  x_1 \ g(x_1) &= [0.1 \ 0 \ 0.5] \\
  x_2 \ g(x_2) &= [0.3 \ 0.6 \ 0.2] \\
  x_3 \ g(x_3) &= [0 \ 0.4 \ 0] \\
\end{align*}
\]

Pooling \( G(X) = [0.3 \ 0.6 \ 0.5] \)

\( x_1, x_2 \) are active

G(x) depends solely on \( x_1, x_2 \)

Active points

\( x_1 \)

\( x_2 \)
Sparse coding for image search

- Co-active

\[ x_1 \cdot g(x_1) = [0.1 \quad 0 \quad 0.5] \]
Sparse coding for image search

- Co-active

\[ x_1 \ g(x_1) = [0.1 \ 0 \ 0.5] \]
\[ y_1 \ g(y_1) = [0.2 \ 0.1 \ 0.6] \]
Sparse coding for image search

• Co-active

\[ x_1 \ g(x_1) = [0.1 \ 0 \ 0.5] \]
\[ y_1 \ g(y_1) = [0.2 \ 0.1 \ 0.6] \]

\( x_1, y_1 \) are co-active

Should be true active pair
Sparse coding for image search

• Co-active

\[ x_1 \ g(x_1) = [0.1 \ 0 \ 0.5] \]
\[ y_1 \ g(y_1) = [0.2 \ 0.1 \ 0.6] \]

\[ x_1, y_1 \text{ are co-active} \]

Should be true active pair

\[ x_1 \]
\[ y_1 \]
Sparse coding for image search
Sparse coding for image search

- Most descriptors are active
Sparse coding for image search

- Most descriptors are active
- Many correct corresponding pairs!
Sparse coding for image search

- Most descriptors are active
- Many correct corresponding pairs!
- Sparse coding is a feature matcher
Sparse coding for image search

- Another example:
Multiple feature
Multiple feature

Feature 1 → Coding & Pooling → Vector 1
Multiple feature

Feature 1 → Coding & Pooling → Vector 1

Feature 2
Multiple feature

Feature 1 → Vector 1

Coding & Pooling

Feature 2 → Vector 2

Coding & Pooling
Multiple feature

Feature 1

Coding & Pooling

Vector 1

Coding & Pooling

Vector 2

Concatenate

Vector
Multiple feature

- Feature 1
  - Coding & Pooling
  - Vector 1

- Feature 2
  - Coding & Pooling
  - Vector 2

  Concatenate

  Vector
  PCA
  Compact Vector
Multiple feature

- Do not hurt memory & computation efficiency
Exploit feature combination:
- Detector: Harris corner, LOG (Laplacian of Gaussian)
- Descriptor: SIFT, DAISY
- Best configuration: Harris-DAISY(HD) + LOG-SIFT(LS)
Sparse-coded micro feature

- Color features
- Inspired by bag-of-colors (BOC) [Wengert et al. 11]

Dense sampling → CIE-Lab color → N*N color patch → Micro feature → N * N * 3 Dim
Sparse-coded micro feature

• Active points
Sparse-coded micro feature

- Active points

Focus on distinctive points
Sparse-coded micro feature

- Active points
Sparse-coded micro feature

- Active points

Focus on distinctive points
patches in smooth region
SC is a filter!
Single feature comparison

- Compare SC framework with VLAD and Fisher Kernel using the same local feature --- Harris-DAISY(HD, 104Dim)
- Datasets: INREA Holidays(mAP) & UKB(score/4)

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Dim</th>
<th>Holidays(mAP)</th>
<th>UKB(score/4)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>D</td>
<td>-&gt;128</td>
</tr>
<tr>
<td>Fisher(HD)</td>
<td>6656</td>
<td>0.566</td>
<td>0.530</td>
</tr>
<tr>
<td>VLAD(HD)</td>
<td>6656</td>
<td>0.559</td>
<td>0.527</td>
</tr>
<tr>
<td>SC(HD)</td>
<td>5000</td>
<td>0.599</td>
<td>0.525</td>
</tr>
</tbody>
</table>
Multiple features

- Adding more features
- HD – Harris-Daisy    LS – LOG-SIFT    Micro – Micro feature

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Dim</th>
<th>Holidays(mAP)</th>
<th>UKB(score/4)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>D</td>
<td>-&gt;128</td>
</tr>
<tr>
<td>SC(HD)</td>
<td>5000</td>
<td>0.599</td>
<td>0.525</td>
</tr>
<tr>
<td>SC(HD+LS)</td>
<td>10000</td>
<td>0.664</td>
<td>0.599</td>
</tr>
<tr>
<td>SC(HD+LS+Micro)</td>
<td>11024</td>
<td>0.767</td>
<td>0.727</td>
</tr>
</tbody>
</table>
Multiple features

- Adding more features
- HD – Harris-Daisy    LS – LOG-SIFT    Micro – Micro feature

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Dim</th>
<th>Holidays(mAP)</th>
<th>UKB(score/4)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>D</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-&gt;128</td>
<td></td>
</tr>
<tr>
<td>SC(HD)</td>
<td>5000</td>
<td>0.599</td>
<td>3.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.525</td>
<td>3.29</td>
</tr>
<tr>
<td>SC(HD+LS)</td>
<td>10000</td>
<td>0.664</td>
<td>3.50</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.599</td>
<td>3.45</td>
</tr>
<tr>
<td>SC(HD+LS+Micro)</td>
<td>11024</td>
<td>0.767</td>
<td>3.76</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.727</td>
<td>3.67</td>
</tr>
</tbody>
</table>
Scalability study

- Holidays + 4M images from *Flickr*.
Scalability study

- Holidays + 4M images from *Flickr*. 
Scalability study

- Holidays + 4M images from *Flickr*.
Scalability study

- Holidays + 4M images from Flickr.
Conclusion

• Our work:
  - Is based on local feature aggregation
  - Applies sparse coding
  - Utilize multiple features
  - Designs novel “Micro feature”
Thank you!