A Unified Approach to Object Category Recognition

Dr. Rahul Sukthankar
Intel Research and Carnegie Mellon

UCF Vision Class Guest Lecture
Nov 13, 2008

Collaborators: L. Yang, R. Jin, F. Jurie
Details in IEEE CVPR 2008 paper
Object Category Recognition

- sheep? ✗
- bus?  ✓
- cat?  ✗
- bicycle? ✓
- car?  ✓
- cow?  ✗
- dog?  ✗
- horse? ✗
- mbike? ✓
- person? ✓
Standard Approach
(adopted from text IR)

[Fei-Fei et al., 2005];
[Sivic et al., 2005];
and many others

Feature extraction and representation
(e.g., SIFT)

Classification
(e.g., SVMs)

“Bag of visual words”

- sheep? ×
- bus? ✓
- cat? ×
- bicycle? ✓
- car? ✓
- cow? ×
- dog? ×
- horse? ×
- mbike? ✓
- person? ✓

SVM

Quantization + histogram
Codebook Construction by Clustering

Limitation (I):
Universal dictionary $\rightarrow$ category independent
Unsupervised clustering $\rightarrow$ ignores labeling information
Limitation (I):
Universal dictionary → category independent

Unsupervised clustering → ignores labeling information

[Farquar et al., 2005] class-specific dictionaries (MAP)
[Perronnin et al., 2006] class-specific histograms
Limitation (I):
Universal dictionary → category independent

Unsupervised clustering → ignores labeling information

[Moosmann et al., 2006] discriminative random forests
**Limitation (II):**

Every SIFT feature forced into one cluster → failure to capture partial similarity

Difficulty in deciding the number of clusters → wrong choice leads to poor dictionaries
Limitation (II):

Every SIFT feature forced into one cluster → failure to capture partial similarity

Difficulty in deciding the number of clusters → wrong choice leads to poor dictionaries

[Perronnin et al., 2007] soft assignment to words

[Heller & Ghahramani, 2007] overlapping clustering
Codebook Construction by Clustering

Limitation (II):

Every SIFT feature forced into one cluster → failure to capture partial similarity

Difficulty in deciding the number of clusters → wrong choice leads to poor dictionaries

[Winn et al., 2005] pair-wise word merging

[Liu et al., 2007] discriminative cluster refinement

Feature extraction and representation (e.g., SIFT)

Quantization + histogram

Classification (e.g., SVMs)

“Bag of visual words”
Codebook Construction by Clustering

Limitation (III):
Codebook may not be discriminative to differentiate object categories

Separated!
Understanding Clustering

- Clustering is a special coding

Diagram:
- C1: \( x_1, x_2 \)
- C2: \( x_3, x_4 \)
- C3: \( x_5, x_6 \)
Understanding Clustering

• Clustering is a special coding
  – Two coding vectors are either identical or orthogonal
  – Two coding vectors differ by at most two bits

• More general coding
  – Error Correcting Output Code (ECOC)

\[
\begin{array}{cccc}
  & C1 & C2 & C3 \\
 x_1 & 1 & 0 & 0 \\
 x_2 & 1 & 0 & 0 \\
 x_3 & 0 & 1 & 0 \\
 x_4 & 0 & 1 & 0 \\
 x_5 & 0 & 0 & 1 \\
 x_6 & 0 & 0 & 1 \\
\end{array}
\]
Understanding Clustering

- Clustering is a special coding
  - Two coding vectors are either identical or orthogonal
  - Two coding vectors differ by at most two bits
- Our approach: coding by thresholded projections
Understanding Clustering

• Clustering is a special coding
  – Two coding vectors are either identical or orthogonal
  – Two coding vectors differ by at most two bits

• Our approach: coding by thresholded projections

<table>
<thead>
<tr>
<th></th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x3</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x4</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x5</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x6</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Understanding Clustering

• Clustering is a special coding
  – Two coding vectors are either identical or orthogonal
  – Two coding vectors differ by at most two bits
• Our approach: coding by thresholded projections
Understanding Clustering

- Clustering is a special coding
  - Two coding vectors are either identical or orthogonal
  - Two coding vectors differ by at most two bits
- Our approach: coding by thresholded projections
Understanding Clustering

• Clustering is a special coding
  – Two coding vectors are either identical or orthogonal
  – Two coding vectors differ by at most two bits

• Our approach: coding by thresholded projections
  – Non-orthogonal codes – chosen for maximal class separation
  – Key questions: how to select the projections P and thresholds b?

<table>
<thead>
<tr>
<th></th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>x2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>x3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>x4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>x5</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>x6</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Anatomy of a Visual Bit

\[ g_k(x, y) = I(x^T w_k^y - b_k^y) = \begin{cases} 1 & x^T w_k^y > b_k^y \\ 0 & x^T w_k^y \leq b_k^y \end{cases} \]

“Is this feature relevant to the ‘bus’ category?”

• Weakly-supervised learning of visual bits
• Applying visual bits to object category recognition
# Image Classification using Visual Bits

<table>
<thead>
<tr>
<th>Feature</th>
<th>$g_1(x,a)$</th>
<th>$g_2(x,a)$</th>
<th>...</th>
<th>$g_7(x,a)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>1</td>
<td>1</td>
<td>..</td>
<td>...</td>
</tr>
<tr>
<td>$x_2$</td>
<td>1</td>
<td>0</td>
<td>..</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$x_n$</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**Category $a$**

Feature-level representation
Image Classification using Visual Bits

Category $a$

<table>
<thead>
<tr>
<th>$g_1(x,a)$</th>
<th>$g_2(x,a)$</th>
<th>...</th>
<th>$g_T(x,a)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>1</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>$x_2$</td>
<td>1</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>$x_n$</td>
<td>0</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>$X$</td>
<td>2</td>
<td>1</td>
<td>...</td>
</tr>
</tbody>
</table>

image representation
Image Classification using Visual Bits

<table>
<thead>
<tr>
<th>Category $a$</th>
<th>$g_1(X,a)$</th>
<th>$g_2(X,a)$</th>
<th>$\ldots$</th>
<th>$g_T(X,a)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>2</td>
<td>1</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
</tbody>
</table>

Classifier for Cat. $a$

$$f_a(X) = \sum_{k=1}^{T} \alpha_k g_k(X,a)$$
### Image Classification using Visual Bits

#### Category a

<table>
<thead>
<tr>
<th></th>
<th>$g_1(X,a)$</th>
<th>$g_2(X,a)$</th>
<th>...</th>
<th>$g_T(X,a)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>2</td>
<td>1</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Classifier for Cat. a

$$ f_a(X) = \sum_{k=1}^{T} \alpha_k g_k(X, a) $$

#### Category b

<table>
<thead>
<tr>
<th></th>
<th>$g_1(X,b)$</th>
<th>$g_2(X,b)$</th>
<th>...</th>
<th>$g_T(X,b)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Classifier for Cat. b

$$ f_b(X) = \sum_{k=1}^{T} \alpha_k g_k(X, b) $$

#### Category z

<table>
<thead>
<tr>
<th></th>
<th>$g_1(X,z)$</th>
<th>$g_2(X,z)$</th>
<th>...</th>
<th>$g_T(X,z)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Classifier for Cat. z

$$ f_z(X) = \sum_{k=1}^{T} \alpha_k g_k(X, z) $$
# Image Classification using Visual Bits

## Classifier for Cat. $a$

$$f_a(X) = \sum_{k=1}^{T} \alpha_k g_k(X, a)$$

<table>
<thead>
<tr>
<th>Category $a$</th>
<th>$g_1(X,a)$</th>
<th>$g_2(X,a)$</th>
<th>...</th>
<th>$g_T(X,a)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>2</td>
<td>1</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

## Classifier for Cat. $b$

$$f_b(X) = \sum_{k=1}^{T} \alpha_k g_k(X, b)$$

<table>
<thead>
<tr>
<th>Category $b$</th>
<th>$g_1(X,b)$</th>
<th>$g_2(X,b)$</th>
<th>...</th>
<th>$g_T(X,b)$</th>
</tr>
</thead>
</table>

## Classifier for Cat. $z$

$$f_z(X) = \sum_{k=1}^{T} \alpha_k g_k(X, z)$$

<table>
<thead>
<tr>
<th>Category $z$</th>
<th>$g_1(X,z)$</th>
<th>$g_2(X,z)$</th>
<th>...</th>
<th>$g_T(X,z)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Learn visual bit functions $g(x, a)$ and weights $\alpha$ together.

Unify code generation with discriminative classifier.
Image Classification using Visual Bits

### Category a

<table>
<thead>
<tr>
<th>$g_1(x,a)$</th>
<th>$g_2(x,a)$</th>
<th>$\ldots$</th>
<th>$g_T(x,a)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>2</td>
<td>1</td>
<td>$\ldots$</td>
</tr>
</tbody>
</table>

**Classifier for Cat. a**

$$f_a(X) = \sum_{k=1}^{T} \alpha_k g_k(X, a)$$

### Category b

<table>
<thead>
<tr>
<th>$g_1(x,b)$</th>
<th>$g_2(x,b)$</th>
<th>$\ldots$</th>
<th>$g_T(x,b)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>0</td>
<td>1</td>
<td>$\ldots$</td>
</tr>
</tbody>
</table>

**Classifier for Cat. b**

$$f_b(X) = \sum_{k=1}^{T} \alpha_k g_k(X, b)$$

### Category z

$$f_a(X) = \sum_{i=1}^{N} \alpha_i \cdot k(\ldots)$$

**Classifier for Cat. z**

$$f_a(X) = \sum_{k=1}^{T} \alpha_k g_k(X, z)$$

$\vec{g}(X,a) = (g_1(X,a), \ldots, g_T(X,a))$: visual bit vector

$k(x,x): \mathbb{R}^T \times \mathbb{R}^T \rightarrow \mathbb{R}$: kernel function
Image Classification using Visual Bits

Category a

<table>
<thead>
<tr>
<th>g_1(x,a)</th>
<th>g_2(x,a)</th>
<th>...</th>
<th>g_T(x,a)</th>
</tr>
</thead>
</table>
| X        | 2        | 1   | ...      |...

Classifier for Cat. a

\[ f_a(X) = \sum_{k=1}^{T} \alpha_k g_k(X, a) \]

Category b

<table>
<thead>
<tr>
<th>g_1(x,b)</th>
<th>g_2(x,b)</th>
<th>...</th>
<th>g_T(x,b)</th>
</tr>
</thead>
</table>
| X        | 2        | 1   | ...      |...

Classifier for Cat. b

\[ f_b(X) = \sum_{k=1}^{T} \alpha_k g_k(X, b) \]

Category z

<table>
<thead>
<tr>
<th>g_1(x,z)</th>
<th>g_2(x,z)</th>
<th>...</th>
<th>g_T(x,z)</th>
</tr>
</thead>
</table>
| X        | 2        | 1   | ...      |...

Classifier for Cat. z

\[ f_z(X) = \sum_{k=1}^{T} \alpha_k g_k(X, z) \]

Generalizes to nonlinear classifier
(can be implemented using standard SVM)

\[ f_a(X) = \sum_{i=1}^{N} \alpha_i k(\vec{g}(X, a), \vec{g}(X_i, a)) \]

\[ k(\vec{x}, \vec{x}) : \mathbb{R}^T \times \mathbb{R}^T \rightarrow \mathbb{R} \text{ kernel function} \]

\[ \vec{g}(X, a) = (g_1(X, a), \ldots, g_T(X, a)) : \text{visual bit vector} \]
Relevant Visual Bits Localize Concepts

Relevance of feature $x$ to category $y$

$$\sum_{k=1}^{T} \alpha_k g_k(x, y)$$
Unified Approach

Feature extraction and representation (e.g., SIFT)

- sheep? ✗
- bus? ✓
- cat? ✗
- bicycle? ✓
- car? ✓
- cow? ✗
- dog? ✗
- horse? ✗
- mbike? ✓
- person? ✓

SVM SVM SVM SVM SVM SVM SVM SVM SVM SVM

class-specific visual bits

class-specific visual bits

class-specific visual bits

class-specific visual bits

class-specific visual bits

code classify
Unified Approach

Feature extraction and representation (e.g., SIFT)

How to learn this discriminative representation in a weakly-supervised setting?
Learning Visual Bits
Optimization Framework

• Given visual bit functions \( g(x, a) \) and weights \( \alpha \), how to measure if they are able to classify image \( X=(x_1,\ldots,x_n) \) into cat. \((y_1, y_2, \ldots, y_K)\)

Challenge
Which features correspond to which categories, or do not correspond to any category of interest at all?
Learning Visual Bits
Optimization Framework

• Given visual bit functions $g(x, a)$ and weights $\alpha$, how to measure if they are able to classify image $X=(x_1, \ldots, x_n)$ into cat. $(y_1, y_2, \ldots, y_K)$

Solution: consider all possibilities

$$f(x_3, y_1) = \sum_{k=1}^{T} \alpha_k g_k(x_3, y_1)$$
$$f(x_3, y_2) = \sum_{k=1}^{T} \alpha_k g_k(x_3, y_2)$$
$$f(x_3, y_3) = \sum_{k=1}^{T} \alpha_k g_k(x_3, y_3)$$
Learning Visual Bits
Optimization Framework

- Given visual bit functions $g(x, a)$ and weights $\alpha$, how to measure if they are able to classify image $X=(x_1, \ldots, x_n)$ into cat. $(y_1, y_2, \ldots, y_K)$

Solution: consider all possibilities

$$f(x_3, y_1) = \sum_{k=1}^{T} \alpha_k g_k(x_3, y_1)$$

$$f(x_3, y_2) = \sum_{k=1}^{T} \alpha_k g_k(x_3, y_2)$$

$$f(x_3, y_3) = \sum_{k=1}^{T} \alpha_k g_k(x_3, y_3)$$

$$e(x_3, y_i) = \frac{\exp(f(x_3, y_i))}{\sum_{z=1}^{m} \exp(f(x_3, z))}$$
Learning Visual Bits
Optimization Framework

• Given visual bit functions \( g(x, a) \) and weights \( \alpha \), how to measure if they are able to classify image \( X = (x_1, \ldots, x_n) \) into cat. \( (y_1, y_2, \ldots, y_K) \)

Loss function for image \( X \)

\[
l(X, y_1) = \frac{n}{\sum_{j=1}^{n} e(x_j, y_1)}
\]
Learning Visual Bits
Optimization Framework

• Given visual bit functions \( g(x, a) \) and weights \( \alpha \), how to measure if they are able to classify image \( X=(x_1, \ldots, x_n) \) into cat. \((y_1, y_2, \ldots, y_K)\)

\[
\begin{align*}
\text{Loss function for image } X & \\
\quad l(X, y_1) = \frac{n}{\sum_{j=1}^{n} e(x_j, y_1)} \\
\quad l(X, y) = \sum_{y \in y} l(X, y) \\
\text{Loss function for the image collection} & \\
\mathcal{L} (\alpha_1:T, g_1:T) &= \sum_{i=1}^{N} l(X_i, y_i)
\end{align*}
\]
Learning Visual Bits
Optimization Framework

Given a collection of training images

\[ T = \{(X_i, y_i), i = 1, \ldots, N\} \]

Find optimal visual bits and combination weights by solving

\[
\min_{g_{1:T}, \alpha_{1:T}} \mathcal{L}(\alpha_{1:T}, g_{1:T}) = \sum_{i=1}^{N} l(X_i, y_i)
\]

Overview of optimization algorithm (reminiscent of boosting)

- Iterative approach: learn one visual bit \((g)\) and weight \((\alpha)\) at a time
- Employ bound optimization to decouple \(g\) and \(\alpha\)

[details in paper and supplementary material]
Results on PASCAL 2006
(AUR with 100 training examples)

• Follows methodology from [Marszalek & Schmid, 2006]
• Baselines
  – Standard: K-means (k=1000) + SVM (χ² kernel)
  – Discriminative: Extremely Randomized Clustering Forests

<table>
<thead>
<tr>
<th>Class</th>
<th>KM-SVM</th>
<th>ERCF</th>
<th>Our Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>sheep</td>
<td>0.551 ± 0.046</td>
<td>0.747 ± 0.017</td>
<td>0.842 ± 0.008</td>
</tr>
<tr>
<td>bus</td>
<td>0.618 ± 0.030</td>
<td>0.708 ± 0.024</td>
<td>0.930 ± 0.005</td>
</tr>
<tr>
<td>cat</td>
<td>0.697 ± 0.011</td>
<td>0.753 ± 0.015</td>
<td>0.759 ± 0.016</td>
</tr>
<tr>
<td>bicycle</td>
<td>0.750 ± 0.026</td>
<td>0.744 ± 0.021</td>
<td>0.782 ± 0.021</td>
</tr>
<tr>
<td>car</td>
<td>0.654 ± 0.043</td>
<td>0.731 ± 0.019</td>
<td>0.875 ± 0.007</td>
</tr>
<tr>
<td>cow</td>
<td>0.519 ± 0.026</td>
<td>0.751 ± 0.026</td>
<td>0.790 ± 0.017</td>
</tr>
<tr>
<td>dog</td>
<td>0.670 ± 0.011</td>
<td>0.706 ± 0.026</td>
<td>0.761 ± 0.012</td>
</tr>
<tr>
<td>horse</td>
<td>0.503 ± 0.016</td>
<td>0.712 ± 0.025</td>
<td>0.671 ± 0.009</td>
</tr>
<tr>
<td>motor</td>
<td>0.496 ± 0.017</td>
<td>0.733 ± 0.019</td>
<td>0.782 ± 0.013</td>
</tr>
<tr>
<td>person</td>
<td>0.551 ± 0.035</td>
<td>0.729 ± 0.015</td>
<td>0.722 ± 0.007</td>
</tr>
</tbody>
</table>
Conclusion

• Unify codebook construction + classifier training
  – Generate codebooks by iterative projection
  – Efficiently learn projection and weights together

• Impact on object category recognition
  – Learns better representations with limited training data
  – No parameters to tune