Recent Development of the Bag-of-Feature Model in Visual Recognition

Lei Wang
Faculty of Informatics
University of Wollongong
20-Dec-2011 @ PCM
Outline

- Introduction
- Codebook generation
- Feature coding
- Feature pooling
- Conclusion

Images courtesy of related papers and authors
Introduction

Object recognition (Caltech 256)  
Object recognition (PASCAL VOC 2011)

Action recognition (UIUC Sports)  
Scene recognition (Lazebnik, CVPR06)

Unusual event detection (Zhong, CVPR04)  
Medical image retrieval (Wang, TMI07)
Bag-of-feature model is borrowed from text analysis
Introduction

Bag-of-feature model is borrowed from text analysis

Interest point detection
or
Dense sampling

The cropped detected regions
Introduction

A close-up view
Introduction

A close-up view
Introduction

Extract features from all training/test images

\[ x \in \mathbb{R}^d \]
Cluster all features to generated “Visual Words”
Introduction

Generated “Visual Words”

Word 1: ...

Word 2: ...

Word 3: ...

Word 4: ...

... ...

Word k: ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...
Introduction

From an image to a histogram

\[ [n_1, n_2, \ldots, n_k] \in \mathbb{R}^k \]

The number of occurrence of 1st “word” in this image

\[ [0, 0, 1, \ldots, 0] \in \mathbb{R}^k \]

\[ [1, 0, 0, \ldots, 0] \in \mathbb{R}^k \]

\[ [0, 1, 0, \ldots, 0] \in \mathbb{R}^k \]
Introduction

Classifying images

$x = w^T x + b$
A Bag-of-Feature Image Categorization System

- Image database
- Feature extraction
- Codebook generation
- Classification
- Feature pooling
- Feature coding
Introduction

Image Database

- PASCAL VOC (28K+ images, 20 classes)
- Caltech-101 (9K+ images), Caltech-256 (30K+ images)
- Scene-15 (4K+ images)
- Graz-02 (4 classes)
- ImageNet (14M+ images, 20K+ concepts)

...
Feature Detection & Description

• Harris-Affine, Hessian-Affine, MSER, IBR & EBR, Salient region detector
• SIFT, HOG, SURF, CENTRIST, filter-based, ...

• K. Mikolajczyk et al., A Comparison of Affine Region Detectors, IJCV, VOL. 65, NO. 1/2, 2005
Introduction

Codebook Generation

- *k*-means clustering, Vocabulary tree
- Gaussian mixture model
- Randomized clustering forests
- Information loss minimization
- Latent mixture model
- Compact codebooks
- ...

...
Introduction

Feature Coding

- Hard assignment, Soft assignment
- Kernel codebook
- Fisher kernels
- Sparse coding
- Locality-constrained linear coding
- Localized soft assignment
- ...

Image courtesy of Wang, CVPR2010
Introduction

Feature Pooling

- Sum pooling
- Max pooling
- Mix-order pooling

... ... ...

\[
\begin{bmatrix}
0, 0, 1, \ldots, 0 \\
1, 0, 0, \ldots, 0 \\
0, 1, 0, \ldots, 0
\end{bmatrix} \in \mathbb{R}^k
\]

\[ n_1, n_2, \ldots, n_m \] \in \mathbb{R}^k
Classification Method

- **Nonlinear & linear Support Vector Machines**
- Nearest-neighbor classifier
- Boosting
- Naïve Bayes classifier, Hierarchical Bayesian model
- ...

\[ y = w^T x + b \]
Local Invariant Features, such as SIFT (Lowe, ICCV99); Video Google (Sivic, CVPR03); Bag-of-keypoints (Csurka, SLCV@ECCV04)

Vocabulary tree (Nister, CVPR06); Randomized Clustering Forests (Moosmann, NIPS06); Spatial Pyramid Matching (Lazebnik, CVPR06)

Pyramid Match Kernel (Grauman, ICCV05); Dense sampling (Jurie, ICCV05); Compact Codebook (Winn, ICCV05)

Sparse coding for BoF (Yang, CVPR09); Local Coordinate Coding (Yu, NIPS09)

Local Soft-assignment & Mix-order pooling (Liu, ICCV11); Comparative Study on BoF model (Chatfield, BMVC, 2011); Relative Attribute (ICCV, 2011)

Locality-constrained Linear Coding for BoF (Wang, CVPR10); Coding & pooling scheme comparison (Boureau, CVPR10);

Kernel Codebook (van Gemert, ECCV08); In Defense of Nearest Neighbor Classifier (Boiman, CVPR08)

Comparative Study (Zhang, IJCV07); Fisher Kernels (Perronnin, CVPR07)

Video Google (Sivic, CVPR03); Bag-of-keypoints (Csurka, SLCV@ECCV04)
Outline

- Introduction
- Codebook generation
- Feature coding
- Feature pooling
- Conclusion

Images courtesy of related papers and authors
Codebook generation

A Bag-of-Feature Image Categorization System

Image database → Feature extraction → Codebook generation

Codebook generation → Feature coding

Feature coding → Feature pooling

Feature pooling → Classification
Codebook Generation

- $k$-means clustering
- Vocabulary tree
- Randomized clustering forests
- Information loss minimization
- Compact codebooks
**Codebook generation**

**$k$-means Clustering**

Given a set of features $\{x_1, x_2, \cdots, x_n\}$, where $x_i \in \mathbb{R}^d$, it finds an optimal partition $C_1^*, C_2^*, \cdots, C_k^*$ via

$$\{C_1^*, C_2^*, \cdots, C_k^*\} = \arg \min \sum_{i=1}^{k} \sum_{x_j \in C_i} \|x_j - \mu_i\|^2$$

- What is the optimal $k$?
- What if $n$ or $k$ is very large?
- Any better partitioning / prototypes?
- How to incorporate class label information?
Codebook generation

\textbf{\textit{k}-means Clustering}

- What is the optimal \( k \) ?
  - Codebook combination, codeword selection, \textit{compact} codebook

- What if \( n \) or \( k \) is very large ?
  - Vocabulary tree, randomized clustering forests, fast \( k \)-means

- Better partition / prototypes ?
  - GMM, fixed-radius partition, mean-shift, \textit{Information loss minimization}, Beyond Euclidean distance

- Incorporate class label information?
  - \textit{Information loss minimization}, discriminative codebooks
Vocabulary Tree

• A tree-structured visual codebook
• built by **hierarchical** \( k \)-means clustering

Images Courtesy of Nister and Stewenius, CVPR06
Codebook generation

Vocabulary Tree

• Larger vocabulary
  – More discriminative, better retrieval performance

• Faster retrieval (sub-second time for 1M images)
  – Hierarchically assign the descriptors of a query
  – Use the inverted file approach

• Hierarchical scoring scheme for retrieval
  – Besides the leaves, other nodes are considered
  – Combine score at each internal nodes and leaf
Codebook generation

Vocabulary Tree

\[ w_i = \ln \frac{N}{N_i} \]

For each node \( i \)

\[
\begin{align*}
q_i &= n_i w_i \\
d_i &= m_i w_i
\end{align*}
\]

\[ s(q, d) = \left\| \frac{q}{\|q\|} - \frac{d}{\|d\|} \right\| \]

Illustration of the invert file structure

Images Courtesy of Nister and Stewenius, CVPR06
Codebook generation

Vocabulary Tree

Percentage of at least one in top 5

Images Courtesy of Nister and Stewenius, CVPR06
Randomized Clustering Forests

- Aims to handle large number of descriptors and large-sized codebooks
- Extremely Randomized Clustering Forests
  - Ensembles of randomly created clustering trees
  - Faster training and test
- Discriminative codebooks
  - Incorporates class-label information
  - Training “classification” trees and use them as “clustering” trees
  - More accurate classification
Randomized Clustering Forests

Illustration of image recognition with ERC-forests-based visual codebooks

Images Courtesy of Moosmann et al., NIPS06
Randomized Clustering Forests

Given a set of descriptors $\mathcal{L} = \{(x_i, y_i)\}_{i=1}^{n}$, where $x_i \in \mathbb{R}^d$, and $y_i$ is the class label of the corresponding image.

A boolean test $\mathcal{T} = \{x_i \cdot \theta\}$
Both $i$ and $\theta$ are selected \textbf{randomly}

Two key parameters:

$S_{min}$: the minimal purity of the partition using $(i, \theta)$
$T_{max}$: the maximal number of trials on $(i, \theta)$
Randomized Clustering Forests

Graz02, bike vs. none; Courtesy of Moosmann et al., NIPS06
Information Loss Minimization

• Aims to make quantized representation preserve as much as possible discriminative information
  – Being a sufficient statistic of its class label
• K-means style but using a new criterion
  – Minimize the loss on label information due to quantization
• Discriminative codebooks
  – Incorporates class-label information
Information Loss Minimization

\[ I(X; Y) - I(K; Y) = \frac{1}{N} \sum_{i=1}^{k} \sum_{x_j \in C_i} D(P_{x_j \mid \pi_i}) \]

\[ + \sum_{i=1}^{k} \sum_{x_j \in C_i} \| x_j - \mu_i \|^2 \]

Images Courtesy of Lazebnik and Raginsky, TPAMI09
Codebook generation

Information Loss Minimization

\[ I(X; Y) - I(K; Y) = \frac{1}{N} \sum_{i=1}^{k} \sum_{x_j \in C_i} D(P_{x_j} \| \pi_i) \]
Codebook generation

Information Loss Minimization

(a) Centers and partitions produced by $k$-means ($C = 32$)

(b) Centers and partitions produced by our info-loss method ($C = 32$)

Red dot: class +1
Blue dot: class -1

Images Courtesy of Lazebnik and Raginsky, TPAMI09
Codebook generation

Information Loss Minimization

<table>
<thead>
<tr>
<th></th>
<th>$C = 32$</th>
<th>$C = 64$</th>
<th>$C = 128$</th>
<th>$C = 256$</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>56.2 ± 0.9</td>
<td>60.3 ± 1.7</td>
<td>62.7 ± 0.7</td>
<td>64.9 ± 0.4</td>
</tr>
<tr>
<td>k-means</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NB</td>
<td>60.8 ± 0.9</td>
<td>62.9 ± 1.4</td>
<td>64.8 ± 0.8</td>
<td>66.6 ± 0.7</td>
</tr>
<tr>
<td>info-loss</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVM</td>
<td>59.5 ± 0.6</td>
<td>65.8 ± 0.5</td>
<td>70.4 ± 0.8</td>
<td>73.3 ± 0.3</td>
</tr>
<tr>
<td>k-means</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVM</td>
<td>63.9 ± 0.4</td>
<td>68.0 ± 0.5</td>
<td>71.6 ± 0.7</td>
<td>74.7 ± 0.4</td>
</tr>
<tr>
<td>info-loss</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Scene-15 data set; Courtesy of Lazebnik and Raginsky, TPAMI09
Compact Codebooks

- Aims to generate **small-sized** codebooks by maximally maintain **discriminative** power
- Hierarchically merge a large-sized initial codebook subject to various criteria
  - Conditional probability of class label
  - Mutual information loss
  - Class separability measure
- Discriminative codebooks
  - Incorporates **class-label information**
Codebook generation

Compact Codebooks

Illustration of the effect of merging two visual words

Images Courtesy of Winn et al., ICCV05
Compact Codebooks
(Conditional probability of class label)

\[ h = (h_1, h_2, \cdots, h_k)^\top \] is the histogram of words,
\( \hat{h}_{rs} \) is the histogram obtained after merging bins \( r \) and \( s \).

\{\hat{h}_{rs}\} is the set of the new histograms of training images.
\( c \) is the set of the class labels of training images.

Conditional probability of class label (UVD, Winn et al. ICCV05)

\[
P(c|\{\hat{h}_{rs}\}) = \frac{P(\{\hat{h}_{rs}\}|c)}{P(\{\hat{h}_{rs}\}|c) + P(\{\hat{h}_{rs}\}|c_{\text{same}})}
\]
Compact Codebooks
(Mutual information loss)

\[ \mathbf{h} = (h_1, h_2, \cdots, h_k) \]
\[ \hat{\mathbf{h}}_{rs} \] is the histogram obtained after merging bins \( r \) and \( s \).
\[ \{\hat{\mathbf{h}}_{rs}\} \] is the set of the new histograms of training images.
\( \mathbf{c} \) is the set of the class labels of training images.

Loss of mutual information (AIB, Fulkerson et al. ECCV08)

\[ \delta I_{rs} = I(\{\mathbf{h}\}, \mathbf{c}) - I(\{\hat{\mathbf{h}}_{rs}\}, \mathbf{c}) \]
Compact Codebooks
(Class separability measure)

\( \mathbf{h} = (h_1, h_2, \ldots, h_k)^	op \) is the histogram of words,
\( \hat{\mathbf{h}}_{rs} \) is the histogram obtained after merging bins \( r \) and \( s \).

\( \hat{\mathbf{S}}_b \) is the between-class scatter matrix by using \( \{\hat{\mathbf{h}}_{rs}\} \).
\( \hat{\mathbf{S}}_w \) is the within-class scatter matrix by using \( \{\hat{\mathbf{h}}_{rs}\} \).

Class separability measure (CSM, Wang et al. ECCV08)

\[
J_{rs} = \frac{\text{trace}(\hat{\mathbf{S}}_b)}{\text{trace}(\hat{\mathbf{S}}_w)}
\]
Compact Codebooks
(Class separability measure)

\[ J_{rs} = \frac{\text{trace}(\hat{S}_b)}{\text{trace}(\hat{S}_w)} = \frac{\text{trace}(S_b) + f_{rs}}{\text{trace}(S_w) + g_{rs}} = \frac{f_{rs} - (-\text{trace}(S_b))}{g_{rs} - (-\text{trace}(S_w))} \]

Images Courtesy of Wang et al., ECCV08

The slope between two 2D points
\( A(-\text{trace}(S_w), -\text{trace}(S_b)) \) and \( B(g_{rs}, f_{rs}) \)
Compact Codebooks

Comparison of time cost

Comparison of memory cost

Images Courtesy of Wang et al., ECCV08
Compact Codebooks

Images Courtesy of Wang et al., ECCV08
Compact Codebooks

Images Courtesy of Wang et al., ECCV08
Compact Codebooks

A Generalized Probabilistic Framework for Compact Codebook Creation

Images Courtesy of Liu et al., CVPR11
Compact Codebooks

MME (Multinomial distribution + Max-margin parameter distribution)

\[
\begin{align*}
\min_{c_p,c_q} & \sum_{j=1}^{t} \left( \log \frac{P(v_j|c_p)}{P(v_j|c_q)} \right)^2 + \left( \log \frac{P(c_p)}{P(c_q)} \right)^2 + \lambda \sum_{i,c} \xi_{i,c} \\
\text{s.t.} & \sum_{j=1}^{t} h_{ij} \left( \log \frac{P(v_j|c_i)}{P(v_j|c)} \right) + \left( \log \frac{P(c_i)}{P(c)} \right) \geq 1 - \xi_{i,c}, \\
\xi_{i,c} & \geq 0, \quad \forall \; c \neq c_i; \quad \forall \; i = 1, 2, \cdots, n; \\
\forall \; p \neq q, \; p, q, c = 1, 2, \cdots, C;
\end{align*}
\]
Compact Codebooks

Comparison of five compact codebook creation algorithms on Graz02 data set

Images Courtesy of Liu et al., CVPR11
Compact Codebooks

Comparison of five compact codebook creation algorithms on 15-Scenes data set

Images Courtesy of Liu et al., CVPR11
Comparison of five compact codebook creation algorithms on KTH data set

Images Courtesy of Liu et al., CVPR11
Codebook Generation

- $k$-means clustering
- Vocabulary tree
- Randomized clustering forests
- Information loss minimization
- Compact codebooks
A Bag-of-Word Image Categorization System

1. Image database
2. Feature extraction
3. Feature pooling
4. Codebook generation
5. Feature coding
6. Classification
Feature Coding

- Represent a descriptor with visual codebook
- Hard assignment, Soft assignment
- Kernel codebook
- Fisher kernels, Super-vector coding
- Sparse coding, Locality-constrained coding
- Localized soft assignment

Courtesy of Wang et al., CVPR2010
How to reduce the quantization error?

How to consider the underlying manifold structure?
Hard or Soft assignment

• How to reduce the quantization error?
  – Kernel codebook
  – Fisher kernels
  – Sparse coding, Laplacian Sparse coding

• How to consider the manifold structure?
  – Locality-constrained coding
  – Localized soft assignment
Kernel Codebook

• Address two drawbacks in hard assignment
  – Codeword uncertainty
    • Rigidly assigning a descriptor to one codeword even if there are multiple relevant codewords
  – Codeword plausibility
    • Rigidly assigning a descriptor to one codeword even if there is no a suitable candidate

• Use kernel density estimation to allow a degree of ambiguity in assigning descriptors
Feature Coding

Kernel Codebook

A point with codeword uncertainty

A good point

A point with codeword plausibility

Codewords

Images Courtesy of Gemert et al., ECCV08
Kernel Codebook

Kernel codebook scheme:

\[ \omega_i = K_\sigma (D(x, w_i)) \]

Codebook uncertainty scheme:

\[ \omega_i = \frac{K_\sigma (D(x, w_i))}{\sum_{j=1}^{m} K_\sigma (D(x, w_j))} \]

Codebook plausibility scheme:

\[ \omega_i = \begin{cases} 
K_\sigma (D(x, w_i)) , & \text{if } w_i \text{ is closest to } x; \\
0 , & \text{otherwise}
\end{cases} \]
Kernel Codebook

Images Courtesy of Gemert et al., ECCV08
Kernel Codebook

Scene-15 data set; Courtesy of Gemert et al., ECCV08
Fisher Kernel

• Uses high-order information to code a descriptor vector
  – Small codebook, large coding vector

• Fisher kernel
  – Compare samples (descriptor vectors) induced by a generative probabilistic model (visual codebook)

• The gradient space of a generative model
  – Reflect how an observed sample affects the estimation of model parameters
Fisher Kernel

A generative model \( p(x|\theta) \)

\[
\mathcal{L}(\theta) = \log p(x|\theta)
\]

Fisher score:

\[
u_x = \nabla_{\theta} \log p(x|\theta)
\]

Fisher kernel:

\[
k(x_i, x_j) = u_{x_i}^\top F^{-1} u_{x_j}
\]
Fisher Kernel

Visual codebook $\rightarrow$ Gaussian mixture model

$$\hat{p}(x|\theta) = \sum_{i=1}^{k} w_i p_i(x|\{\mu_i, \sigma_i\})$$

Fisher score:

\[
\begin{align*}
\mathbf{u}_x(\gamma_i) &= \frac{\gamma_i(x)}{w_i} - \frac{\gamma_1(x)}{w_1} \\
\mathbf{u}_x(\mu_i,j) &= \gamma_i(x) \left( \frac{x_j - \mu_{i,d}}{\sigma_{i,j}^2} \right) \\
\mathbf{u}_x(\sigma_i,j) &= \gamma_i(x) \left( \frac{(x_j - \mu_{i,j})^2}{\sigma_{i,j}^3} - \frac{1}{\sigma_{i,j}} \right)
\end{align*}
\]

\[\text{dim} = 2dk + (k-1)\]
Comparison of the contribution of different parameters (128 words only, In-house data set)

<table>
<thead>
<tr>
<th>Gradient</th>
<th>$\text{max } F_1 \text{ (in %)}$</th>
<th>Gradient Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w$</td>
<td>58.1</td>
<td>127</td>
</tr>
<tr>
<td>$\mu$</td>
<td>69.4</td>
<td>6,400</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>70.4</td>
<td>6,400</td>
</tr>
<tr>
<td>$\mu \sigma$</td>
<td>74.1</td>
<td>12,800</td>
</tr>
<tr>
<td>$w \mu \sigma$</td>
<td>74.1</td>
<td>12,927</td>
</tr>
</tbody>
</table>

Images Courtesy of Perronnin and Dance, CVPR07
### Improved Fisher Kernel

<table>
<thead>
<tr>
<th>Method</th>
<th>AP (in %)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard FK (SIFT) [22]</td>
<td>47.9</td>
</tr>
<tr>
<td>Best of VOC07 [8]</td>
<td>59.4</td>
</tr>
<tr>
<td>Context (SIFT) [28]</td>
<td>59.4</td>
</tr>
<tr>
<td>Kernel Codebook [3]</td>
<td>60.5</td>
</tr>
<tr>
<td>MKL [6]</td>
<td>62.2</td>
</tr>
<tr>
<td>Cls + Loc [29]</td>
<td>63.5</td>
</tr>
<tr>
<td>IFK (SIFT)</td>
<td>58.3</td>
</tr>
<tr>
<td>IFK (SIFT+Color)</td>
<td>60.3</td>
</tr>
</tbody>
</table>

(PASCAL VOC 2007 256 words only)

<table>
<thead>
<tr>
<th>Method</th>
<th>ntrain=15</th>
<th>ntrain=30</th>
<th>ntrain=45</th>
<th>ntrain=60</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kernel Codebook [3]</td>
<td>-</td>
<td>27.2 (0.4)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EMK (SIFT) [20]</td>
<td>23.2 (0.6)</td>
<td>30.5 (0.4)</td>
<td>34.4 (0.4)</td>
<td>37.6 (0.5)</td>
</tr>
<tr>
<td>Standard FK (SIFT) [22]</td>
<td>25.6 (0.6)</td>
<td>29.0 (0.5)</td>
<td>34.9 (0.2)</td>
<td>38.5 (0.5)</td>
</tr>
<tr>
<td>Sparse Coding (SIFT) [18]</td>
<td>27.7 (0.5)</td>
<td>34.0 (0.4)</td>
<td>37.5 (0.6)</td>
<td>40.1 (0.9)</td>
</tr>
<tr>
<td>Baseline (SIFT) [26]</td>
<td>-</td>
<td>34.1 (0.2)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>NN (SIFT) [19]</td>
<td>-</td>
<td>38.0 (-)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>NN [19]</td>
<td>-</td>
<td>42.7 (-)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>IFK (SIFT)</td>
<td>34.7 (0.2)</td>
<td>40.8 (0.1)</td>
<td>45.0 (0.2)</td>
<td>47.9 (0.4)</td>
</tr>
</tbody>
</table>

(Caltech-256, 256 words only)

Images Courtesy of Perronnin et al., ECCV10
Sparse Coding

- Minimizes the reconstruction error in coding a descriptor vector
- Jointly learns the coding coefficient and visual codebook
- Uses a small set of visual words to represent a descriptor vector —— Sparsity
- Works well with linear SVMs, significantly reduce training and test time
Sparse Coding

\( \mathbf{x} \in \mathbb{R}^d; \quad \mathbf{V}_{d \times k} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k); \quad \mathbf{U}_{k \times n} = (\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n) \)

\( k \)-means clustering

\[
\min_{\mathbf{U}, \mathbf{V}} \sum_{i=1}^{n} \left\| \mathbf{x}_i - \mathbf{V} \mathbf{u}_i \right\|^2 \\
\text{s.t.} \quad \|\mathbf{u}_i\|_0 = 1, \quad \|\mathbf{u}_i\|_1 = 1, \quad \mathbf{u}_i \geq 0, \quad \forall i
\]

Sparse coding

\[
\min_{\mathbf{U}, \mathbf{V}} \sum_{i=1}^{n} \left( \left\| \mathbf{x}_i - \mathbf{V} \mathbf{u}_i \right\|^2 + \lambda \|\mathbf{u}_i\|_1 \right) \\
\text{s.t.} \quad \|\mathbf{v}_j\| \cdot 1, \quad \forall j = 1, 2, \cdots, k
\]
Sparse Coding

\( \mathbf{x} \in \mathbb{R}^d; \quad \mathbf{V}_{d \times k} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k); \quad \mathbf{U}_{k \times n} = (\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n) \)

When \( V \) is fixed, for each descriptor (Lasso)

\[
\begin{align*}
\min_{\mathbf{u}_i} & \quad \| \mathbf{x}_i - \mathbf{V} \mathbf{u}_i \|^2 + \lambda \| \mathbf{u}_i \|_1 \\
\end{align*}
\]

When \( U \) is fixed (Least-squares problem + quadratic constraints)

\[
\begin{align*}
\min_{\mathbf{V}} & \quad \sum_{i=1}^{n} \| \mathbf{x}_i - \mathbf{V} \mathbf{u}_i \|^2 \\
\text{s.t.} & \quad \| \mathbf{v}_j \| \leq 1, \quad \forall j = 1, 2, \cdots, k \\
\end{align*}
\]
Sparse Coding

### Classification rate (%) comparison on Caltech-101.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>15 training</th>
<th>30 training</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zhang et al. [28]</td>
<td>59.10 ± 0.60</td>
<td>66.20 ± 0.50</td>
</tr>
<tr>
<td>KSPM [12]</td>
<td>56.40</td>
<td>64.40 ± 0.80</td>
</tr>
<tr>
<td>NBNN [1]</td>
<td>65.00 ± 1.14</td>
<td>70.40</td>
</tr>
<tr>
<td>ML+CORR [9]</td>
<td>61.00</td>
<td>69.60</td>
</tr>
<tr>
<td>KC [25]</td>
<td>–</td>
<td>64.14 ± 1.18</td>
</tr>
<tr>
<td>KSPM</td>
<td>56.44 ± 0.78</td>
<td>63.99 ± 0.88</td>
</tr>
<tr>
<td>LSPM</td>
<td>53.23 ± 0.65</td>
<td>58.81 ± 1.51</td>
</tr>
<tr>
<td>ScSPM</td>
<td>67.0 ± 0.45</td>
<td>73.2 ± 0.54</td>
</tr>
</tbody>
</table>

### Classification rate (%) comparison on 15 scenes.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Classification Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>KSPM [12]</td>
<td>81.40 ± 0.50</td>
</tr>
<tr>
<td>KC [25]</td>
<td>76.67 ± 0.39</td>
</tr>
<tr>
<td>KSPM</td>
<td>76.73 ± 0.65</td>
</tr>
<tr>
<td>LSPM</td>
<td>65.32 ± 1.02</td>
</tr>
<tr>
<td>ScSPM</td>
<td>80.28 ± 0.93</td>
</tr>
</tbody>
</table>

### Classification rate (%) comparison on Caltech-256 dataset.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>15 train</th>
<th>30 train</th>
<th>45 train</th>
<th>60 train</th>
</tr>
</thead>
<tbody>
<tr>
<td>KSPM [8]</td>
<td>–</td>
<td>34.10</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>KC [25]</td>
<td>–</td>
<td>27.17 ± 0.46</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>KSPM</td>
<td>23.34 ± 0.42</td>
<td>29.51 ± 0.52</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>LSPM</td>
<td>13.20 ± 0.62</td>
<td>15.45 ± 0.37</td>
<td>16.37 ± 0.47</td>
<td>16.57 ± 1.01</td>
</tr>
<tr>
<td>ScSPM</td>
<td>27.73 ± 0.51</td>
<td>34.02 ± 0.35</td>
<td>37.46 ± 0.55</td>
<td>40.14 ± 0.91</td>
</tr>
</tbody>
</table>

Courtesy of Raina et al., ICML07

Courtesy of Yang et al., CVPR09

Copyright © 2007, 2009 Stanford University
Feature Coding

Laplacian Sparse Coding

\[ \mathbf{x} \in \mathbb{R}^d; \quad \mathbf{V}_{d \times k} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k); \quad \mathbf{U}_{k \times n} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n) \]

Sparse coding

\[
\min_{\mathbf{U}, \mathbf{V}} \sum_{i=1}^{n} \left( \| \mathbf{x}_i - \mathbf{Vu}_i \|^2 + \lambda \| \mathbf{u}_i \|_1 \right) \\
\text{s.t.} \quad \| \mathbf{v}_j \| \cdot 1, \quad \forall j = 1, 2, \ldots, k
\]

Laplacian sparse coding

\[
\min_{\mathbf{U}, \mathbf{V}} \sum_{i=1}^{n} \left( \| \mathbf{x}_i - \mathbf{Vu}_i \|^2 + \lambda \| \mathbf{u}_i \|_1 \right) + \gamma \text{trace}(\mathbf{ULU}^\top) \\
\text{s.t.} \quad \| \mathbf{v}_j \| \cdot 1, \quad \forall j = 1, 2, \ldots, k
\]
Feature Coding

Laplacian Sparse Coding

Images Courtesy of Gao et al., CVPR10

<table>
<thead>
<tr>
<th>Method</th>
<th>Average Classification Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>KSPM[8]</td>
<td>81.40±0.50</td>
</tr>
<tr>
<td>ScSPM[25]</td>
<td>80.28±0.93</td>
</tr>
<tr>
<td>HIK+OCSVM[22]</td>
<td>84.00±0.46</td>
</tr>
<tr>
<td>LScSPM</td>
<td>89.75±0.50</td>
</tr>
</tbody>
</table>

**Scene-15 data set**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>NA</td>
<td>NA</td>
<td>27.73±0.51</td>
<td>30.00±0.14</td>
</tr>
<tr>
<td>30</td>
<td>34.10</td>
<td>27.17±0.46</td>
<td>34.02±0.35</td>
<td>35.74±0.10</td>
</tr>
<tr>
<td>45</td>
<td>NA</td>
<td>NA</td>
<td>37.46±0.55</td>
<td>38.54±0.36</td>
</tr>
<tr>
<td>60</td>
<td>NA</td>
<td>NA</td>
<td>40.14±0.91</td>
<td>40.43±0.38</td>
</tr>
</tbody>
</table>

**Caltech-256 data set**
Locality-constrained Coding

- Minimizes the reconstruction error in coding a descriptor vector
- Jointly learns the coding coefficient and visual codebook
- Locality is more essential than sparsity, and locality leads to sparsity
- Works well with linear SVMs, significantly reduce training and test time
Locality-constrained Coding

\( \mathbf{x} \in \mathbb{R}^d; \quad \mathbf{V}_{d \times k} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k); \quad \mathbf{U}_{k \times n} = (\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n) \)

Inspired by “Local Coordinate Coding” (Yu et al., NIPS09)

\[
\left| f(\mathbf{x}) - \sum_{i=1}^{k} u_i f(\mathbf{v}_i) \right| \cdot \alpha \| \mathbf{x} - \mathbf{V}_k \mathbf{u} \| + \beta \sum_{i=1}^{k} |u_i| \| \mathbf{v}_i - \mathbf{V}_k \mathbf{u} \|^{1+p}
\]

Reconstruction error
prefer coding with closer bases
Feature Coding

**Locality-constrained Coding**

\[ \mathbf{x} \in \mathbb{R}^d; \quad \mathbf{V}_{d \times k} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k); \quad \mathbf{U}_{k \times n} = (\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n) \]

**Sparse coding**

\[
\begin{aligned}
\min_{\mathbf{U}, \mathbf{V}} & \quad \sum_{i=1}^{n} (\| \mathbf{x}_i - \mathbf{V}\mathbf{u}_i \|^2 + \lambda \| \mathbf{u}_i \|_1) \\
\text{s.t.} & \quad \| \mathbf{v}_j \| \cdot 1, \quad \forall j = 1, 2, \cdots, k
\end{aligned}
\]

**Local Coordinate Coding**

\[
\begin{aligned}
\min_{\mathbf{U}, \mathbf{V}} & \quad \sum_{i=1}^{n} \left( \| \mathbf{x}_i - \mathbf{V}\mathbf{u}_i \|^2 + \lambda \sum_{j=1}^{k} |u_{ij}| \| \mathbf{x}_i - \mathbf{v}_j \|^2 \right) \\
\text{s.t.} & \quad \mathbf{1}^\top \mathbf{u}_i = 1, \quad \| \mathbf{v}_j \| \cdot 1, \quad \forall j = 1, 2, \cdots, k
\end{aligned}
\]
Feature Coding

Locality-constrained Coding

\[ \mathbf{x} \in \mathbb{R}^d; \quad \mathbf{V}_{d \times k} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k); \quad \mathbf{U}_{k \times n} = (\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n) \]

Locality-constrained Coding

\[
\min_{\mathbf{U}, \mathbf{V}} \sum_{i=1}^{n} \left( \| \mathbf{x}_i - \mathbf{V} \mathbf{u}_i \|^2 + \lambda \| \mathbf{d}_i \|^2 \mathbf{u}_i \|^2 \right)
\]

s.t. \quad 1^T \mathbf{u}_i = 1, \quad \| \mathbf{v}_j \| \cdot 1, \quad \forall j = 1, 2, \cdots, k

Images Courtesy of Wang et al., CVPR10
Feature Coding

Locality-constrained Coding

<table>
<thead>
<tr>
<th>training images</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
</tr>
</thead>
<tbody>
<tr>
<td>50%</td>
<td>5.6</td>
<td>5.8</td>
<td>6.5</td>
<td>7.0</td>
<td>7.5</td>
<td>8.0</td>
</tr>
<tr>
<td>60%</td>
<td>4.6</td>
<td>5.5</td>
<td>6.2</td>
<td>6.5</td>
<td>7.0</td>
<td>7.5</td>
</tr>
<tr>
<td>70%</td>
<td>3.6</td>
<td>4.5</td>
<td>5.2</td>
<td>5.5</td>
<td>6.0</td>
<td>6.5</td>
</tr>
<tr>
<td>80%</td>
<td>2.6</td>
<td>3.5</td>
<td>4.2</td>
<td>4.5</td>
<td>5.0</td>
<td>5.5</td>
</tr>
<tr>
<td>90%</td>
<td>1.6</td>
<td>2.5</td>
<td>3.2</td>
<td>3.5</td>
<td>4.0</td>
<td>4.5</td>
</tr>
<tr>
<td>100%</td>
<td>0.6</td>
<td>1.5</td>
<td>2.2</td>
<td>2.5</td>
<td>3.0</td>
<td>3.5</td>
</tr>
</tbody>
</table>

Image classification results on Caltech-101 dataset

<table>
<thead>
<tr>
<th>training images</th>
<th>15</th>
<th>30</th>
<th>45</th>
<th>60</th>
</tr>
</thead>
<tbody>
<tr>
<td>50%</td>
<td>5.6</td>
<td>6.0</td>
<td>7.0</td>
<td>7.5</td>
</tr>
<tr>
<td>60%</td>
<td>4.6</td>
<td>5.0</td>
<td>6.0</td>
<td>6.5</td>
</tr>
<tr>
<td>70%</td>
<td>3.6</td>
<td>4.0</td>
<td>5.0</td>
<td>5.5</td>
</tr>
<tr>
<td>80%</td>
<td>2.6</td>
<td>3.0</td>
<td>4.0</td>
<td>4.5</td>
</tr>
<tr>
<td>90%</td>
<td>1.6</td>
<td>2.0</td>
<td>3.0</td>
<td>3.5</td>
</tr>
<tr>
<td>100%</td>
<td>0.6</td>
<td>1.0</td>
<td>2.0</td>
<td>2.5</td>
</tr>
</tbody>
</table>

Image classification results using Caltech-256 dataset

<table>
<thead>
<tr>
<th>training images</th>
<th>15</th>
<th>30</th>
<th>45</th>
<th>60</th>
</tr>
</thead>
<tbody>
<tr>
<td>50%</td>
<td>5.6</td>
<td>6.0</td>
<td>7.0</td>
<td>7.5</td>
</tr>
<tr>
<td>60%</td>
<td>4.6</td>
<td>5.0</td>
<td>6.0</td>
<td>6.5</td>
</tr>
<tr>
<td>70%</td>
<td>3.6</td>
<td>4.0</td>
<td>5.0</td>
<td>5.5</td>
</tr>
<tr>
<td>80%</td>
<td>2.6</td>
<td>3.0</td>
<td>4.0</td>
<td>4.5</td>
</tr>
<tr>
<td>90%</td>
<td>1.6</td>
<td>2.0</td>
<td>3.0</td>
<td>3.5</td>
</tr>
<tr>
<td>100%</td>
<td>0.6</td>
<td>1.0</td>
<td>2.0</td>
<td>2.5</td>
</tr>
</tbody>
</table>

Images Courtesy of Wang et al., CVPR10
# Locality-constrained Coding

<table>
<thead>
<tr>
<th>object class</th>
<th>aero</th>
<th>bicy</th>
<th>bird</th>
<th>boat</th>
<th>bottle</th>
<th>bus</th>
<th>car</th>
<th>cat</th>
<th>chair</th>
<th>cow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obj.+Contex [20]</td>
<td>80.2</td>
<td>61.0</td>
<td>49.8</td>
<td>69.6</td>
<td>21.0</td>
<td>66.8</td>
<td>80.7</td>
<td>51.1</td>
<td>51.4</td>
<td>35.9</td>
</tr>
<tr>
<td>Best PASCAL'07 [6]</td>
<td>77.5</td>
<td>63.6</td>
<td>56.1</td>
<td>71.9</td>
<td>33.1</td>
<td>60.6</td>
<td>78.0</td>
<td>58.8</td>
<td>53.5</td>
<td>42.6</td>
</tr>
<tr>
<td>Ours</td>
<td>74.8</td>
<td><strong>65.2</strong></td>
<td>50.7</td>
<td>70.9</td>
<td>28.7</td>
<td><strong>68.8</strong></td>
<td>78.5</td>
<td><strong>61.7</strong></td>
<td><strong>54.3</strong></td>
<td><strong>48.6</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>object class</th>
<th>table</th>
<th>dog</th>
<th>horse</th>
<th>mbike</th>
<th>person</th>
<th>plant</th>
<th>sheep</th>
<th>sofa</th>
<th>train</th>
<th>tv</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obj.+Contex [20]</td>
<td>62.0</td>
<td>38.6</td>
<td>69.0</td>
<td>61.4</td>
<td>84.6</td>
<td>28.7</td>
<td><strong>53.5</strong></td>
<td><strong>61.9</strong></td>
<td><strong>81.7</strong></td>
<td><strong>59.5</strong></td>
<td><strong>58.4</strong></td>
</tr>
<tr>
<td>Best of PASCAL'07 [6]</td>
<td>54.9</td>
<td><strong>45.8</strong></td>
<td><strong>77.5</strong></td>
<td><strong>64.0</strong></td>
<td><strong>85.9</strong></td>
<td><strong>36.3</strong></td>
<td>44.7</td>
<td>50.9</td>
<td>79.2</td>
<td>53.2</td>
<td><strong>59.4</strong></td>
</tr>
<tr>
<td>Ours</td>
<td>51.8</td>
<td>44.1</td>
<td>76.6</td>
<td><strong>66.9</strong></td>
<td>83.5</td>
<td>30.8</td>
<td>44.6</td>
<td>53.4</td>
<td>78.2</td>
<td>53.5</td>
<td>59.3</td>
</tr>
</tbody>
</table>

Images Courtesy of Wang et al., CVPR10
Localized Soft Assignment

• Soft-assignment coding
  – computational **efficiency** and conceptual **simplicity**
  – However, it is inferior to sparse or localized coding

• Localized Soft Assignment
  – Identify the cause of inferiority of soft assignment
  – Integrate **locality** into soft-assignment coding

• Works well with **linear SVMs**
  – Significantly reduce coding time
  – Performance comparable to existing coding schemes
Localized Soft Assignment

Kernel codebook:

\[ \omega_i = \frac{K_\sigma(D(x, v_i))}{\sum_{j=1}^{m} K_\sigma(D(x, v_j))} \]

Interpreted as posteriori probability

\[ p(v_i | x) = \frac{\exp(-\beta \| x - v_i \|^2)}{\sum_{j=1}^{k} \exp(-\beta \| x - v_j \|^2)} \]

Shorter Euclidean distance is reliable

Longer Euclidean distance is unreliable

A data manifold

Images Courtesy of Liu et al., ICCV11
Localized Soft Assignment

- Use smaller $\beta$ value
- Only consider the $r$ nearest visual words

$$p(v_i | x) = \frac{\exp(-\beta d(x, v_i))}{\sum_{j=1}^{k} \exp(-\beta d(x, v_j))}$$

$$d(x, v_i) = \begin{cases} 
\|x - v_i\|^2; & \text{if } v_i \in \mathcal{N}_r(x) \\
+\infty; & \text{otherwise.}
\end{cases}$$

Images Courtesy of Liu et al., ICCV11
Localized Soft Assignment

### Comparison on Caltech-101 data set

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Classification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localized soft-assignment coding</td>
<td>74.21 ± 0.81</td>
</tr>
<tr>
<td>Soft-assignment coding</td>
<td>72.56 ± 0.65</td>
</tr>
<tr>
<td>LLC</td>
<td>71.25 ± 0.98</td>
</tr>
<tr>
<td>Hard-assignment coding [8]</td>
<td>64.6 ± 0.80</td>
</tr>
<tr>
<td>Hard-assignment coding [2]</td>
<td>64.3 ± 0.9</td>
</tr>
<tr>
<td>Soft-assignment coding [6]</td>
<td>64.1 ± 1.2</td>
</tr>
<tr>
<td>Soft-assignment coding [2]</td>
<td>69.0 ± 0.8</td>
</tr>
<tr>
<td>Sparse coding [16]</td>
<td>73.2 ± 0.55</td>
</tr>
<tr>
<td>Sparse coding [2]</td>
<td>71.5 ± 1.1</td>
</tr>
<tr>
<td>LLC [15]</td>
<td>73.44 ± -</td>
</tr>
</tbody>
</table>

### Comparison on Scene-15 data set

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Classification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Localized soft-assignment coding</td>
<td>82.70 ± 0.39</td>
</tr>
<tr>
<td>Soft-assignment coding</td>
<td>81.09 ± 0.43</td>
</tr>
<tr>
<td>LLC</td>
<td>81.53 ± 0.65</td>
</tr>
<tr>
<td>Hard-assignment coding [8]</td>
<td>81.4 ± 0.50</td>
</tr>
<tr>
<td>Hard-assignment coding [2]</td>
<td>80.1 ± 0.6</td>
</tr>
<tr>
<td>Soft-assignment coding [6]</td>
<td>76.67 ± 0.39</td>
</tr>
<tr>
<td>Soft-assignment coding [2]</td>
<td>81.4 ± 0.6</td>
</tr>
<tr>
<td>Sparse coding [16]</td>
<td>80.28 ± 0.93</td>
</tr>
<tr>
<td>Sparse coding [2]</td>
<td>83.1 ± 0.6</td>
</tr>
</tbody>
</table>

### Comparison on UIUC 8-Sport data set

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Classification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local Soft Assignment</td>
<td>82.29 ± 1.84</td>
</tr>
<tr>
<td>Soft-assignment coding</td>
<td>82.04 ± 2.37</td>
</tr>
<tr>
<td>LLC</td>
<td>81.41 ± 1.84</td>
</tr>
<tr>
<td>Sparse Coding [5]</td>
<td>82.74 ± 1.46</td>
</tr>
</tbody>
</table>

Images Courtesy of Liu et al., ICCV11
Feature Coding

- Represent a descriptor with visual codebook
- Hard assignment, Soft assignment
- Kernel codebook
- Fisher kernel
- Sparse coding, Locality-constrained coding
- Localized soft assignment

Courtesy of Wang et al., CVPR2010
A Bag-of-Word Image Categorization System

1. Image database
2. Feature extraction
3. Codebook generation
4. Classification
5. Feature pooling
6. Feature coding
Feature pooling

- Sum pooling, average pooling
- Max pooling
- Mix-order pooling

\[(\omega_1, \omega_{12}, \ldots, \omega_{1k}) \]
\[(\omega_2, \omega_{22}, \ldots, \omega_{2k}) \]
\[(\omega_3, \omega_{32}, \ldots, \omega_{3k}) \]
\[(\omega_4, \omega_{42}, \ldots, \omega_{4k}) \]
\[\vdots \]

\[\in \mathbb{R}^k\]
Feature pooling

Feature Pooling

\[
\begin{align*}
\sum & \sum & \sum \\
(\omega_{11}, \omega_{12}, \cdots, \omega_{1k}) & (\omega_{21}, \omega_{22}, \cdots, \omega_{2k}) & (\omega_{31}, \omega_{32}, \cdots, \omega_{3k}) \\
(\omega_{41}, \omega_{42}, \cdots, \omega_{4k}) & & (\omega_{11}, \omega_{12}, \cdots, \omega_{1k}) \\
(\omega_{21}, \omega_{22}, \cdots, \omega_{2k}) & (\omega_{31}, \omega_{32}, \cdots, \omega_{3k}) & (\omega_{41}, \omega_{42}, \cdots, \omega_{4k}) \\
(\omega_{11}, \omega_{12}, \cdots, \omega_{1k}) & (\omega_{21}, \omega_{22}, \cdots, \omega_{2k}) & (\omega_{31}, \omega_{32}, \cdots, \omega_{3k}) \\
(\omega_{41}, \omega_{42}, \cdots, \omega_{4k}) & & (\omega_{11}, \omega_{12}, \cdots, \omega_{1k})
\end{align*}
\]

\[y = \sum_i \alpha_i y_i^k(x, x_i) + b\]

\[y = w^\top x + b\]
Feature pooling

Understand Max Pooling

\( p(v_i|x) \): *posteriori* probability of a descriptor to a visual word

\[
p(v_i|x) = \frac{\exp(-\beta \|x - v_i\|^2)}{\sum_{j=1}^{k} \exp(-\beta \|x - v_j\|^2)}
\]

An image: \( \mathcal{I} = \{x_1, x_2, \cdots, x_n\} \)

\( p(v_i|\mathcal{I}) \): probability of the presence of word \( v_i \) in image \( \mathcal{I} \)

\[
p(v_i|\mathcal{I}) = 1 - \prod_{j=1}^{n} (1 - p(v_i|x_j)) \geq \max_{j=1,\cdots,n} p(v_i|x_j)
\]

This is just Max pooling!
Mix-order Max Pooling

An image: \( \mathcal{I} = \{x_1, x_2, \cdots, x_n\} \)

\[ p(\#v_i \geq r \mid \mathcal{I}) : \text{probability "word } v_i \text{ is present in image } \mathcal{I} \text{ no less than } r \text{ times"} \]

\[ p(v_i \mid \mathcal{I}) : \text{a special case of } p(\#v_i \geq r \mid \mathcal{I}) \text{ when } r = 1 \]

Mix-order Max Pooling:

\[ p(\#v_i \geq r \mid \mathcal{I}) \geq \left( \max_{j=1,\cdots,n} p(v_i \mid x_j) \right)^r \]

This is just the \( r \)-th Max pooling!
Feature pooling

Mix-order Max Pooling

Verify \( p(v_i|I) \geq \max_{j=1,\ldots,n} p(v_i|x_j) \)

Comparison of “mix-order max-pooling” with “max-pooling”

<table>
<thead>
<tr>
<th>Data set</th>
<th>Max-pooling</th>
<th>Mix-order max-pooling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caltech-101</td>
<td>90.87 ± 1.46 %</td>
<td>90.47 ± 0.46 %</td>
</tr>
<tr>
<td>Scene-15</td>
<td>82.70 ± 0.39 %</td>
<td>83.76 ± 0.59 %</td>
</tr>
<tr>
<td>UIUC 8-Sport</td>
<td>82.29 ± 1.84 %</td>
<td>84.56 ± 1.5 %</td>
</tr>
</tbody>
</table>

Images Courtesy of Liu et al., ICCV11
Conclusion

A Bag-of-Word Image Categorization System

1. Image database
2. Feature extraction
3. Codebook generation
4. Feature pooling
5. Feature coding
6. Classification
## Conclusion


### Caltech 101

<table>
<thead>
<tr>
<th>Year</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>2004</td>
<td>About 19%</td>
</tr>
<tr>
<td>2005</td>
<td>29%</td>
</tr>
<tr>
<td>2005</td>
<td>Pyramid Matching Kernel</td>
</tr>
<tr>
<td>2005</td>
<td>Bio-inspired</td>
</tr>
<tr>
<td>2006</td>
<td>SVM-KNN</td>
</tr>
<tr>
<td>2006</td>
<td>SPM</td>
</tr>
<tr>
<td>2006</td>
<td>Multi-Layer Filterban</td>
</tr>
<tr>
<td>2006</td>
<td>63%</td>
</tr>
<tr>
<td>2006</td>
<td>77.8%</td>
</tr>
<tr>
<td>2007</td>
<td>Random forest based MKL with detection</td>
</tr>
<tr>
<td>2008</td>
<td>In defense of nearest neighbor …</td>
</tr>
<tr>
<td>2008</td>
<td>Kernel codebook</td>
</tr>
<tr>
<td>2008</td>
<td>66.5%</td>
</tr>
<tr>
<td>2008</td>
<td>69.6%</td>
</tr>
<tr>
<td>2009</td>
<td>HIK codebook</td>
</tr>
<tr>
<td>2009</td>
<td>Sparse coding regions</td>
</tr>
<tr>
<td>2009</td>
<td>LP-beta MKL</td>
</tr>
<tr>
<td>2009</td>
<td>EMK</td>
</tr>
<tr>
<td>2010</td>
<td>LCC</td>
</tr>
<tr>
<td>2010</td>
<td>EMK-descriptor (multiple features)</td>
</tr>
<tr>
<td>2010</td>
<td>Pre-segmentation</td>
</tr>
<tr>
<td>2011</td>
<td>Early cut, single feature</td>
</tr>
<tr>
<td>2011</td>
<td>Fisher kernel with much denser sampling and 7 scales</td>
</tr>
</tbody>
</table>

Courtesy of Liu, 2011
Conclusion

K-means clustering
Hard assignment
Sum pooling
Kernel SVMs
...

Dictionary learning
Localized coding
Max pooling
Linear SVMs
...

Larger-scale visual recognition?
Higher-level features?
Higher-order information?
...

Thank You

Images Courtesy of Google Image