Bag-of-features for category classification

Cordelia Schmid
Category recognition

- Image classification: assigning a class label to the image

- Car: present
- Cow: present
- Bike: not present
- Horse: not present
...
Category recognition

- Image classification: assigning a class label to the image
  - Car: present
  - Cow: present
  - Bike: not present
  - Horse: not present
  - ... 

- Object localization: define the location and the category

  ![Image with objects categorized and localized]
Difficulties: within object variations

Variability: Camera position, Illumination, Internal parameters

Within-object variations
Difficulties: within-class variations
Category recognition

- Robust image description
  - Appropriate descriptors for categories

- Statistical modeling and machine learning for vision
  - Use and validation of appropriate techniques
Why machine learning?

- Early approaches: simple features + handcrafted models
- Can handle only few images, simples tasks

Why machine learning?

- Early approaches: manual programming of rules
- Tedious, limited and does not take into account the data

Figure 3. A system developed in 1978 by Ohta, Kanade and Sakai [33, 32] for knowledge-based interpretation of outdoor natural scenes. The system is able to label an image (c) into semantic classes: S-sky, T-tree, R-road, B-building, U-unknown.

Why machine learning?

• Today lots of data, complex tasks

• Instead of trying to encode rules directly, learn them from examples of inputs and desired outputs
Types of learning problems

- Supervised
  - Classification
  - Regression
- Unsupervised
- Semi-supervised
- Active learning
- ....
Supervised learning

• Given training examples of inputs and corresponding outputs, produce the “correct” outputs for new inputs

• Two main scenarios:
  
  – **Classification**: outputs are discrete variables (category labels). Learn a decision boundary that separates one class from the other

  – **Regression**: also known as “curve fitting” or “function approximation.” Learn a continuous input-output mapping from examples (possibly noisy)
Unsupervised Learning

• Given only *unlabeled* data as input, learn some sort of structure

• The objective is often more vague or subjective than in supervised learning. This is more an exploratory/descriptive data analysis
Unsupervised Learning

- **Clustering**
  - Discover groups of “similar” data points
Unsupervised Learning

- **Quantization**
  - Map a continuous input to a discrete (more compact) output
Unsupervised Learning

- Dimensionality reduction, manifold learning
  - Discover a lower-dimensional surface on which the data lives
Other types of learning

- **Semi-supervised learning**: lots of data is available, but only small portion is labeled (e.g. since labeling is expensive)
Other types of learning

- **Semi-supervised learning**: lots of data is available, but only small portion is labeled (e.g. since labeling is expensive)
  - Why is learning from labeled and unlabeled data better than learning from labeled data alone?
Other types of learning

- **Active learning**: the learning algorithm can choose its own training examples, or ask a “teacher” for an answer on selected inputs.
Image classification

- Given
  Positive training images containing an object class
  ![Motorcycle](image1.png)  ![Motorcycle](image2.png)  ![Motorcycle](image3.png)
  Negative training images that don’t
  ![Pattern](image4.png)  ![Airplane](image5.png)  ![Image](image6.png)

- Classify
  A test image as to whether it contains the object class or not
  ![Motorcycle](image7.png)
Bag-of-features for image classification

- Origin: texture recognition
  - Texture is characterized by the repetition of basic elements or *textons*

Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001
Texture recognition

Texture recognition

Texture recognition

Texture recognition

Texture recognition
Bag-of-features – Origin: bag-of-words (text)

- Orderless document representation: frequencies of words from a dictionary
- Classification to determine document categories

<table>
<thead>
<tr>
<th>Bag-of-words</th>
<th>d1</th>
<th>d2</th>
<th>d3</th>
<th>d4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>People</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Sculpture</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Bag-of-features for image classification

Extract regions  
Compute descriptors  
Find clusters and frequencies  
Compute distance matrix  
Classification

[Nowak, Jurie & Triggs, ECCV’06], [Zhang, Marszalek, Lazebnik & Schmid, IJCV’07]
Bag-of-features for image classification

Step 1
- Extract regions
- Compute descriptors

Step 2
- Find clusters and frequencies

Step 3
- Compute distance matrix
- Classification

[Nowak, Jurie & Triggs, ECCV’06], [Zhang, Marszalek, Lazebnik & Schmid, IJCV’07]
Step 1: feature extraction

- Scale-invariant image regions + SIFT (see lecture 2)
  - Affine invariant regions give “too” much invariance
  - Rotation invariance for many realistic collections “too” much invariance

- Dense descriptors
  - Improve results in the context of categories (for most categories)
  - Interest points do not necessarily capture “all” features

- Color-based descriptors

- Shape-based descriptors
Dense features

- Multi-scale dense grid: extraction of small overlapping patches at multiple scales
- Computation of the SIFT descriptor for each grid cell
- Exp.: Horizontal/vertical step size 6 pixel, scaling factor of 1.2 per level
Bag-of-features for image classification

Step 1: Extract regions

Step 2: Compute descriptors, find clusters and frequencies

Step 3: Compute distance matrix, classification

SVM
Step 2: Quantization
Step 2: Quantization

Clustering
Step 2: Quantization

Visual vocabulary

Clustering
Examples for visual words

<table>
<thead>
<tr>
<th>Category</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Airplanes</td>
<td><img src="image1.png" alt="Airplanes Examples" /></td>
</tr>
<tr>
<td>Motorbikes</td>
<td><img src="image2.png" alt="Motorbikes Examples" /></td>
</tr>
<tr>
<td>Faces</td>
<td><img src="image3.png" alt="Faces Examples" /></td>
</tr>
<tr>
<td>Wild Cats</td>
<td><img src="image4.png" alt="Wild Cats Examples" /></td>
</tr>
<tr>
<td>Leaves</td>
<td><img src="image5.png" alt="Leaves Examples" /></td>
</tr>
<tr>
<td>People</td>
<td><img src="image6.png" alt="People Examples" /></td>
</tr>
<tr>
<td>Bikes</td>
<td><img src="image7.png" alt="Bikes Examples" /></td>
</tr>
</tbody>
</table>
Step 2: Quantization

- Cluster descriptors
  - K-means
  - Gaussian mixture model

- Assign each visual word to a cluster
  - Hard or soft assignment

- Build frequency histogram
Gaussian mixture model (GMM)

- Mixture of Gaussians: weighted sum of Gaussians

\[ p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) \]

where \( \mathcal{N}(x; \mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp \left( -\frac{1}{2}(x - \mu)^\top \Sigma^{-1}(x - \mu) \right) \)
Hard or soft assignment

- **K-means** → hard assignment
  - Assign to the closest cluster center
  - Count number of descriptors assigned to a center

- **Gaussian mixture model** → soft assignment
  - Estimate distance to all centers
  - Sum over number of descriptors

- Represent image by a frequency histogram
Each image is represented by a vector, typically 1000-4000 dimension, normalization with L1 norm.

- fine grained – represent model instances
- coarse grained – represent object categories
Bag-of-features for image classification

Step 1
Extract regions

Step 2
Compute descriptors
Find clusters and frequencies

Step 3
Compute distance matrix
Classification

SVM
Step 3: Classification

- Learn a decision rule (classifier) assigning bag-of-features representations of images to different classes.
Training data

Vectors are histograms, one from each training image

Train classifier, e.g. SVM
Classification

- Assign input vector to one of two or more classes
- Any decision rule divides input space into *decision regions* separated by *decision boundaries*
Nearest Neighbor Classifier

- Assign label of nearest training data point to each test data point

Voronoi partitioning of feature space for 2-category 2-D and 3-D data

from Duda et al.
• For a new point, find the k closest points from training data
• Labels of the k points "vote" to classify
• Works well provided there is lots of data and the distance function is good

$k = 5$
Linear classifiers

- Find linear function (*hyperplane*) to separate positive and negative examples

\[ \mathbf{x}_i \text{ positive: } \mathbf{x}_i \cdot \mathbf{w} + b \geq 0 \]

\[ \mathbf{x}_i \text{ negative: } \mathbf{x}_i \cdot \mathbf{w} + b < 0 \]

Which hyperplane is best?
Linear classifiers - margin

- Generalization is not good in this case:

- Better if a margin is introduced:
Support vector machines

- Find hyperplane that maximizes the *margin* between the positive and negative examples

\[
\begin{align*}
\mathbf{x}_i \text{ positive } (y_i = 1): & \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1 \\
\mathbf{x}_i \text{ negative } (y_i = -1): & \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1 \\
\end{align*}
\]

For support, vectors, \( \mathbf{x}_i \cdot \mathbf{w} + b = \pm 1 \)

The margin is \( 2 / \|\mathbf{w}\| \)
Nonlinear SVMs

• Datasets that are linearly separable work out great:

But what if the dataset is just too hard?

We can map it to a higher-dimensional space:
Nonlinear SVMs

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:
Nonlinear SVMs

- *The kernel trick*: instead of explicitly computing the lifting transformation \( \phi(x) \), define a kernel function \( K \) such that

\[
K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)
\]

- This gives a nonlinear decision boundary in the original feature space:

\[
\sum_i \alpha_i y_i K(x_i, x) + b
\]
Kernels for bags of features

- Hellinger kernel
  \[
  K(h_1, h_2) = \sum_{i=1}^{N} \sqrt{h_1(i)h_2(i)}
  \]

- Histogram intersection kernel
  \[
  I(h_1, h_2) = \sum_{i=1}^{N} \min(h_1(i), h_2(i))
  \]

- Generalized Gaussian kernel
  \[
  K(h_1, h_2) = \exp\left(-\frac{1}{A}D(h_1, h_2)^2\right)
  \]

- \(D\) can be Euclidean distance, \(\chi^2\) distance etc.

\[
D_{\chi^2}(h_1, h_2) = \sum_{i=1}^{N} \frac{(h_1(i) - h_2(i))^2}{h_1(i) + h_2(i)}
\]
Combining features

• SVM with multi-channel chi-square kernel

\[ K(H_i, H_j) = \exp \left( - \sum_{c \in C} \frac{1}{A_c} D_c(H_i, H_j) \right) \]

  - Channel \( c \) is a combination of detector, descriptor
  - \( D_c(H_i, H_j) \) is the chi-square distance between histograms

\[ D_c(H_1, H_2) = \frac{1}{2} \sum_{i=1}^{m} \left[ \frac{(h_{1i} - h_{2i})^2}{(h_{1i} + h_{2i})} \right] \]

  - \( A_c \) is the mean value of the distances between all training sample

  - Extension: learning of the weights, for example with Multiple Kernel Learning (MKL)

Multi-class SVMs

• Various direct formulations exist, but they are not widely used in practice. It is more common to obtain multi-class SVMs by combining two-class SVMs in various ways.

• One versus all:
  – Training: learn an SVM for each class versus the others
  – Testing: apply each SVM to test example and assign to it the class of the SVM that returns the highest decision value

• One versus one:
  – Training: learn an SVM for each pair of classes
  – Testing: each learned SVM “votes” for a class to assign to the test example
Why does SVM learning work?

• Learns foreground and background visual words

  foreground words – high weight

  background words – low weight
Localization according to visual word probability

Illustration

foreground word more probable
background word more probable
A linear SVM trained from positive and negative window descriptors

A few of the highest weighed descriptor vector dimensions (= 'PAS + tile')

+ lie on object boundary (= local shape structures common to many training exemplars)
Bag-of-features for image classification

• Excellent results in the presence of background clutter

bikes  books  building  cars  people  phones  trees
Examples for misclassified images

Books- misclassified into faces, faces, buildings

Buildings- misclassified into faces, trees, trees

Cars- misclassified into buildings, phones, phones
Bag of visual words summary

• Advantages:
  – largely unaffected by position and orientation of object in image
  – fixed length vector irrespective of number of detections
  – very successful in classifying images according to the objects they contain

• Disadvantages:
  – no explicit use of configuration of visual word positions
  – poor at localizing objects within an image
Evaluation of image classification

- PASCAL VOC [05-10] datasets

- PASCAL VOC 2007
  - Training and test dataset available
  - Used to report state-of-the-art results
  - Collected January 2007 from Flickr
  - 500,000 images downloaded and random subset selected
  - 20 classes
  - Class labels per image + bounding boxes
  - 5011 training images, 4952 test images

- Evaluation measure: average precision
PASCAL 2007 dataset

Aeroplane  Bicycle  Bird  Boat  Bottle

Bus  Car  Cat  Chair  Cow
PASCAL 2007 dataset

Dining Table  Dog  Horse  Motorbike  Person

Potted Plant  Sheep  Sofa  Train  TV/Monitor
Evaluation

- **Average Precision [TREC]** averages precision over the entire range of recall
  - Curve interpolated to reduce influence of “outliers”

- A good score requires both high recall and high precision
- Application-independent
- Penalizes methods giving high precision but low recall
Results for PASCAL 2007

• Winner of PASCAL 2007 [Marszalek et al.] : mAP 59.4
  – Combination of several different channels (dense + interest points, SIFT + color descriptors, spatial grids)
  – Non-linear SVM with Gaussian kernel

• Multiple kernel learning [Yang et al. 2009] : mAP 62.2
  – Combination of several features
  – Group-based MKL approach

• Combining object localization and classification [Harzallah et al.’09] : mAP 63.5
  – Use detection results to improve classification
Comparison interest point - dense

**Image classification** results on PASCAL’07 train/val set

<table>
<thead>
<tr>
<th>Method</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>((SHarris + Lap) \times SIFT)</td>
<td>0.452</td>
</tr>
<tr>
<td>MSDense \times SIFT</td>
<td>0.489</td>
</tr>
<tr>
<td>((SHarris + Lap + MSDense) \times SIFT)</td>
<td>0.515</td>
</tr>
</tbody>
</table>

Method: bag-of-features + SVM classifier
Comparison interest point - dense

**Image classification** results on PASCAL’07 train/val set

<table>
<thead>
<tr>
<th>Method</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>(SHarris + Lap) x SIFT</td>
<td>0.452</td>
</tr>
<tr>
<td>MSDense x SIFT</td>
<td>0.489</td>
</tr>
<tr>
<td>(SHarris + Lap + MSDense) x SIFT</td>
<td>0.515</td>
</tr>
</tbody>
</table>

Dense is on average a bit better!
IP and dense are complementary, combination improves results.
Comparison interest point - dense

**Image classification** results on PASCAL’07 train/val set for individual categories

<table>
<thead>
<tr>
<th></th>
<th>(SHarris + Lap) x SIFT</th>
<th>MSDense x SIFT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicycle</td>
<td>0.534</td>
<td>0.443</td>
</tr>
<tr>
<td>PottedPlant</td>
<td>0.234</td>
<td>0.167</td>
</tr>
<tr>
<td>Bird</td>
<td>0.342</td>
<td>0.497</td>
</tr>
<tr>
<td>Boat</td>
<td>0.482</td>
<td>0.622</td>
</tr>
</tbody>
</table>

Results are category dependent!
Evaluation BoF – spatial

**Image classification** results on PASCAL’07 train/val set

<table>
<thead>
<tr>
<th>Spatial layout</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.53</td>
</tr>
<tr>
<td>2x2</td>
<td>0.52</td>
</tr>
<tr>
<td>3x1</td>
<td>0.52</td>
</tr>
<tr>
<td>1,2x2,3x1</td>
<td>0.54</td>
</tr>
</tbody>
</table>

Spatial layout not dominant for PASCAL’07 dataset
Combination improves average results, i.e., it is appropriate for some classes
**Evaluation BoF - spatial**

Image classification results on PASCAL’07 train/val set for individual categories

<table>
<thead>
<tr>
<th>Category</th>
<th>1</th>
<th>3x1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sheep</td>
<td>0.339</td>
<td>0.256</td>
</tr>
<tr>
<td>Bird</td>
<td>0.539</td>
<td>0.484</td>
</tr>
<tr>
<td>DiningTable</td>
<td>0.455</td>
<td>0.502</td>
</tr>
<tr>
<td>Train</td>
<td>0.724</td>
<td>0.745</td>
</tr>
</tbody>
</table>

Results are category dependent! ➔ Combination helps somewhat
Spatial pyramid matching

• Add spatial information to the bag-of-features

• Perform matching in 2D image space

[Lazebnik, Schmid & Ponce, CVPR 2006]
Related work

Similar approaches:
- Subblock description [Szummer & Picard, 1997]
- SIFT [Lowe, 1999]
- GIST [Torralba et al., 2003]
Spatial pyramid representation

Locally orderless representation at several levels of spatial resolution
Spatial pyramid representation

Locally orderless representation at several levels of spatial resolution
Spatial pyramid representation

Locally orderless representation at several levels of spatial resolution
Pyramid match kernel

- Weighted sum of histogram intersections at multiple resolutions (linear in the number of features instead of cubic)

\[ \bigcap \approx \text{optimal partial matching between sets of features} \]
Spatial pyramid matching

- Combination of spatial levels with pyramid match kernel [Grauman & Darell'05]
- Intersect histograms, more weight to finer grids
## Scene dataset [Labzenik et al.’06]

<table>
<thead>
<tr>
<th>Category</th>
<th>Example Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coast</td>
<td><img src="image1" alt="Coast Image" /></td>
</tr>
<tr>
<td>Forest</td>
<td><img src="image2" alt="Forest Image" /></td>
</tr>
<tr>
<td>Mountain</td>
<td><img src="image3" alt="Mountain Image" /></td>
</tr>
<tr>
<td>Open country</td>
<td><img src="image4" alt="Open Country Image" /></td>
</tr>
<tr>
<td>Highway</td>
<td><img src="image5" alt="Highway Image" /></td>
</tr>
<tr>
<td>Inside city</td>
<td><img src="image6" alt="Inside City Image" /></td>
</tr>
<tr>
<td>Tall building</td>
<td><img src="image7" alt="Tall Building Image" /></td>
</tr>
<tr>
<td>Street</td>
<td><img src="image8" alt="Street Image" /></td>
</tr>
<tr>
<td>Suburb</td>
<td><img src="image9" alt="Suburb Image" /></td>
</tr>
<tr>
<td>Bedroom</td>
<td><img src="image10" alt="Bedroom Image" /></td>
</tr>
<tr>
<td>Kitchen</td>
<td><img src="image11" alt="Kitchen Image" /></td>
</tr>
<tr>
<td>Living room</td>
<td><img src="image12" alt="Living Room Image" /></td>
</tr>
<tr>
<td>Office</td>
<td><img src="image13" alt="Office Image" /></td>
</tr>
<tr>
<td>Store</td>
<td><img src="image14" alt="Store Image" /></td>
</tr>
<tr>
<td>Industrial</td>
<td><img src="image15" alt="Industrial Image" /></td>
</tr>
</tbody>
</table>

- **4385 images**
- **15 categories**
Scene classification

<table>
<thead>
<tr>
<th>L</th>
<th>Single-level</th>
<th>Pyramid</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(1x1)</td>
<td>72.2±0.6</td>
<td></td>
</tr>
<tr>
<td>1(2x2)</td>
<td>77.9±0.6</td>
<td>79.0 ±0.5</td>
</tr>
<tr>
<td>2(4x4)</td>
<td>79.4±0.3</td>
<td>81.1 ±0.3</td>
</tr>
<tr>
<td>3(8x8)</td>
<td>77.2±0.4</td>
<td>80.7 ±0.3</td>
</tr>
<tr>
<td>Examples</td>
<td>Image</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>(a) kitchen</td>
<td><img src="image1.png" alt="Image of a kitchen" /></td>
<td></td>
</tr>
<tr>
<td>(b) kitchen</td>
<td><img src="image2.png" alt="Image of a kitchen" /></td>
<td></td>
</tr>
<tr>
<td>(c) store</td>
<td><img src="image3.png" alt="Image of a store" /></td>
<td></td>
</tr>
<tr>
<td>(d) tall bldg</td>
<td><img src="image4.png" alt="Image of a tall building" /></td>
<td></td>
</tr>
<tr>
<td>(e) tall bldg</td>
<td><img src="image5.png" alt="Image of a tall building" /></td>
<td></td>
</tr>
<tr>
<td>(f) inside city</td>
<td><img src="image6.png" alt="Image of inside city" /></td>
<td></td>
</tr>
</tbody>
</table>

Retrieval examples
Category classification – CalTech101

<table>
<thead>
<tr>
<th>L</th>
<th>Single-level</th>
<th>Pyramid</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(1x1)</td>
<td>41.2±1.2</td>
<td></td>
</tr>
<tr>
<td>1(2x2)</td>
<td>55.9±0.9</td>
<td>57.0 ±0.8</td>
</tr>
<tr>
<td>2(4x4)</td>
<td>63.6±0.9</td>
<td>64.6 ±0.8</td>
</tr>
<tr>
<td>3(8x8)</td>
<td>60.3±0.9</td>
<td>64.6 ±0.7</td>
</tr>
</tbody>
</table>

Bag-of-features approach by Zhang et al.’07: 54 %
CalTech101

Easiest and hardest classes

- Sources of difficulty:
  - Lack of texture
  - Camouflage
  - Thin, articulated limbs
  - Highly deformable shape
Discussion

• Summary
  – Spatial pyramid representation: appearance of local image patches + coarse global position information
  – Substantial improvement over bag of features
  – Depends on the similarity of image layout

• Extensions
  – Flexible, object-centered grid
Motivation

- Evaluating the influence of background features [J. Zhang et al., IJCV’07]
  - Train and test on different combinations of foreground and background by separating features based on bounding boxes

![ROC curve](attachment:image.png)

*Training*: original training set

*Testing*: different combinations foreground + background features

Best results when testing with foreground features only
Approach

• Better to train on a “harder” dataset with background clutter and test on an easier one without background clutter

• Spatial weighting for bag-of-features [Marszalek & Schmid, CVPR’06]
  – weight features by the likelihood of belonging to the object
  – determine likelihood based on shape masks
Masks for spatial weighting

For each test feature:

- Select closest training features + corresponding masks
  (training requires segmented images or bounding boxes)

- Align mask based on local co-ordinates system
  (transformation between training and test co-ordinate systems)

Sum masks weighted by matching distance

three features agree on object localization,
the object has higher weights

Weight histogram features with the strength of the final mask
Example masks for spatial weighting
Classification for PASCAL dataset

<table>
<thead>
<tr>
<th></th>
<th>Zhang et al.</th>
<th>Spatial weighting</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>bikes</td>
<td>74.8</td>
<td>76.8</td>
<td>+2.0</td>
</tr>
<tr>
<td>cars</td>
<td>75.8</td>
<td>76.8</td>
<td>+1.0</td>
</tr>
<tr>
<td>motorbikes</td>
<td>78.8</td>
<td>79.3</td>
<td>+0.5</td>
</tr>
<tr>
<td>people</td>
<td>76.9</td>
<td>77.9</td>
<td>+1.0</td>
</tr>
</tbody>
</table>

Equal error rates for PASCAL test set 2
Discussion

• Including spatial information improves results

• Importance of flexible modeling of spatial information
  – coarse global position information
  – object based models
Recent extensions

• Linear Spatial Pyramid Matching Using Sparse Coding for Image Classification. J. Yang et al., CVPR’09.
  – Local coordinate coding, linear SVM, excellent results in 2009 PASCAL challenge

• Learning Mid-level features for recognition, Y. Boureau et al., CVPR’10.
  – Use of sparse coding techniques and max pooling
Recent extensions

• Efficient Additive Kernels via Explicit Feature Maps, A. Vedaldi and Zisserman, CVPR’10.
  – approximation by linear kernels

• Improving the Fisher Kernel for Large-Scale Image Classification, Perronnin et al., ECCV’10
  – More discriminative descriptor, power normalization, linear SVM
Fisher vector image representation

- Mixture of Gaussian/ k-means stores nr of points per cell

- Fisher vector adds 1st & 2nd order moments
  - More precise description of regions assigned to cluster
  - Fewer clusters needed for same accuracy
  - Per cluster also store: mean and variance of data in cell
Fisher vector image representation

\[ X = \{x_t, t = 1 \ldots T\} \] is the set of \( T \) i.i.d. D-dim local descriptors (e.g. SIFT) extracted from an image:

\[ u_\lambda(x) = \sum_{i=1}^{K} w_i u_i(x) \] is a Gaussian Mixture Model (GMM) with parameters \( \lambda = \{w_i, \mu_i, \Sigma_i, i = 1 \ldots N\} \) trained on a large set of local descriptors: a visual vocabulary

FV formulas:

\[
\begin{align*}
G^{X}_{\mu,i} &= \frac{1}{T \sqrt{w_i}} \sum_{t=1}^{T} \gamma_t(i) \left( \frac{x_t - \mu_i}{\sigma_i} \right) \\
G^{X}_{\sigma,i} &= \frac{1}{T \sqrt{2w_i}} \sum_{t=1}^{T} \gamma_t(i) \left[ \frac{(x_t - \mu_i)^2}{\sigma_i^2} - 1 \right]
\end{align*}
\]

\(\gamma_t(i)\) = soft-assignment of patch \(x_t\) to Gaussian \(i\)
Fisher vector image representation

- Fischer vector adds 1st & 2nd order moments
  - More precise description regions assigned to cluster
  - Fewer clusters needed for same accuracy
  - Representation 2D times larger, at same computational cost
  - High dimensional, robust representation
Relation to BOF

FV formulas:

\[ G^X_{\mu,i} = \frac{1}{T \sqrt{w_i}} \sum_{t=1}^{T} \gamma_t(i) \left( \frac{x_t - \mu_i}{\sigma_i} \right) \]

\[ G^X_{\sigma,i} = \frac{1}{T \sqrt{2w_i}} \sum_{i=1}^{T} \gamma_t(i) \left[ \frac{(x_t - \mu_i)^2}{\sigma_i^2} - 1 \right] \]

Soft BOV formula:

\[ \frac{1}{T} \sum_{t=1}^{T} \gamma_t(i) \]

Like the (original) BOV the FV is an average of local statistics.

The FV extends the BOV and includes higher-order statistics (up to 2nd order)

Results on VOC 2007: BOV = 43.6 % → FV = 57.7 % → √FV = 62.1 %