Lecture: Visual Bag of Words

Juan Carlos Niebles and Ranjay Krishna
Stanford Vision and Learning Lab
What we will learn today

• Visual bag of words (BoW)
• Spatial Pyramid Matching
• Naive Bayes
What we will learn today

• Visual bag of words (BoW)
• Spatial Pyramid Matching
• Naïve Bayes
Bag of Words Models

Adapted from slides by Rob Fergus and Svetlana Lazebnik
Object → Bag of ‘words’
Origin 1: Texture Recognition

Example textures (from Wikipedia)
Origin 1: Texture recognition

- Texture is characterized by the repetition of basic elements or *textons*

Origin 1: Texture recognition

Universal texton dictionary

Stanford University
Origin 2: Bag-of-words models

Origin 2: Bag-of-words models

- Orderless document representation: frequencies of words from a dictionary  
  Salton & McGill (1983)

US Presidential Speeches Tag Cloud
http://chir.ag/phernalia/preztags/
Origin 2: Bag-of-words models


US Presidential Speeches Tag Cloud
http://chir.ag/phernalia/preztags/
Origin 2: Bag-of-words models

- Orderless document representation: frequencies of words from a dictionary  
  Salton & McGill (1983)

US Presidential Speeches Tag Cloud
http://chir.ag/phernalia/preztags/
Bags of features for object recognition

- Works pretty well for image-level classification and for recognizing object *instances*
# Bags of features for object recognition

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>airplanes</td>
<td>98.8</td>
<td>97.1</td>
<td>90.2</td>
</tr>
<tr>
<td>cars (rear)</td>
<td>98.3</td>
<td>98.6</td>
<td>90.3</td>
</tr>
<tr>
<td>cars (side)</td>
<td>95.0</td>
<td>87.3</td>
<td>88.5</td>
</tr>
<tr>
<td>faces</td>
<td>100</td>
<td>99.3</td>
<td>96.4</td>
</tr>
<tr>
<td>motorbikes</td>
<td>98.5</td>
<td>98.0</td>
<td>92.5</td>
</tr>
<tr>
<td>spotted cats</td>
<td>97.0</td>
<td>—</td>
<td>90.0</td>
</tr>
</tbody>
</table>
Bag of features

• First, take a bunch of images, extract features, and build up a “dictionary” or “visual vocabulary” – a list of common features

• Given a new image, extract features and build a histogram – for each feature, find the closest visual word in the dictionary
Bag of features: outline

1. Extract features
Bag of features: outline

1. Extract features
2. Learn “visual vocabulary”
Bag of features: outline

1. Extract features
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary
Bag of features: outline

1. Extract features
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary
4. Represent images by frequencies of “visual words”
1. Feature extraction

- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005
1. Feature extraction

- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005

- Interest point detector
  - Csurka et al. 2004
  - Fei-Fei & Perona, 2005
  - Sivic et al. 2005
1. Feature extraction

- Regular grid
  - Vogel & Schiele, 2003
  - Fei-Fei & Perona, 2005

- Interest point detector
  - Csurka et al. 2004
  - Fei-Fei & Perona, 2005
  - Sivic et al. 2005

- Other methods
  - Random sampling (Vidal-Naquet & Ullman, 2002)
  - Segmentation-based patches (Barnard et al. 2003)
2. Learning the visual vocabulary
2. Learning the visual vocabulary

Clustering

Slide credit: Josef Sivic
2. Learning the visual vocabulary

Visual vocabulary

Clustering
K-means clustering recap

- Want to minimize sum of squared Euclidean distances between points $x_i$ and their nearest cluster centers $m_k$

\[
D(X, M) = \sum_{\text{cluster } k} \sum_{\text{point } i \text{ in cluster } k} (x_i - m_k)^2
\]

- Algorithm:
  - Randomly initialize K cluster centers
  - Iterate until convergence:
    - Assign each data point to the nearest center
    - Recompute each cluster center as the mean of all points assigned to it
From clustering to vector quantization

• Clustering is a common method for learning a visual vocabulary or codebook
  – Unsupervised learning process
  – Each cluster center produced by k-means becomes a codevector
  – Codebook can be learned on separate training set
  – Provided the training set is sufficiently representative, the codebook will be “universal”

• The codebook is used for quantizing features
  – A vector quantizer takes a feature vector and maps it to the index of the nearest codevector in a codebook
  – Codebook = visual vocabulary
  – Codevector = visual word
Example visual vocabulary
Image patch examples of visual words
Visual vocabularies: Issues

• How to choose vocabulary size?
  – Too small: visual words not representative of all patches
  – Too large: quantization artifacts, overfitting

• Computational efficiency
  – Vocabulary trees
    (Nister & Stewenius, 2006)
3. Image representation
Image classification

- Given the bag-of-features representations of images from different classes, how do we learn a model for distinguishing them?
Uses of BoW representation

• Treat as feature vector for standard classifier
  – e.g k-nearest neighbors, support vector machine

• Cluster BoW vectors over image collection
  – Discover visual themes
Large-scale image matching

- Bag-of-words models have been useful in matching an image to a large database of object instances.

11,400 images of game covers (Caltech games dataset)

how do I find this image in the database?
Large-scale image search

Build the database:

- Extract features from the database images
- Learn a vocabulary using k-means (typical $k$: 100,000)
- Compute *weights* for each word
- Create an inverted file mapping words $\mapsto$ images
Weighting the words

• Just as with text, some visual words are more discriminative than others

  *the, and, or*  vs.  *cow, AT&T, Cher*

• the bigger fraction of the documents a word appears in, the less useful it is for matching
  – e.g., a word that appears in *all* documents is not helping us
TF-IDF weighting

• Instead of computing a regular histogram distance, we’ll weight each word by it’s *inverse document frequency*

• inverse document frequency (IDF) of word $j =$

$$\log \frac{\text{number of documents}}{\text{number of documents in which } j \text{ appears}}$$
TF-IDF weighting

- To compute the value of bin $j$ in image $l$:

$$\text{term frequency of } j \text{ in } l \times \text{inverse document frequency of } j$$
Inverted file

• Each image has ~1,000 features
• We have ~100,000 visual words
  → each histogram is extremely sparse (mostly zeros)

• Inverted file
  – mapping from words to documents

```
"a": {2}
"banana": {2}
"is": {0, 1, 2}
"it": {0, 1, 2}
"what": {0, 1}
```
Inverted file

• Can quickly use the inverted file to compute similarity between a new image and all the images in the database
  – Only consider database images whose bins overlap the query image
Large-scale image search

query image

• Cons:
  – performance degrades as the database grows

top 6 results
Large-scale image search

• Pros:
  – Works well for CD covers, movie posters
  – Real-time performance possible

real-time retrieval from a database of 40,000 CD covers

Nister & Stewenius, *Scalable Recognition with a Vocabulary Tree*
Example bag-of-words matches
Example bag-of-words matches
## Matching Statistics

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Size</th>
<th>Matches possible</th>
<th>Matches Tried</th>
<th>Matches Found</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dubrovnik</td>
<td>58K</td>
<td>1.6 Billion</td>
<td>2.6M</td>
<td>0.5M</td>
<td>5 hrs</td>
</tr>
<tr>
<td>Rome</td>
<td>150K</td>
<td>11.2 Billion</td>
<td>8.8M</td>
<td>2.7M</td>
<td>13 hrs</td>
</tr>
<tr>
<td>Venice</td>
<td>250K</td>
<td>31.2 Billion</td>
<td>35.5M</td>
<td>6.2M</td>
<td>27 hrs</td>
</tr>
</tbody>
</table>
What about spatial info?
What we will learn today

• Visual bag of words (BoW)
• Spatial Pyramid Matching
• Naïve Bayes
Pyramids

- Very useful for representing images.
- Pyramid is built by using multiple copies of image.
- Each level in the pyramid is 1/4 of the size of previous level.
- The lowest level is of the highest resolution.
- The highest level is of the lowest resolution.
Bag of words + pyramids

Locally orderless representation at several levels of spatial resolution
Bag of words + pyramids

Locally orderless representation at several levels of spatial resolution
Bag of words + pyramids

Locally orderless representation at several levels of spatial resolution

level 0

level 1

level 2
What we will learn today

• Visual bag of words (BoW)
• Spatial Pyramid Matching
• Naïve Bayes
Naïve Bayes

• Classify image using histograms of occurrences on visual words:

\[ x_i \in \{0, 1\} \]

• if only present/absence of a word is taken into account:

• Naïve Bayes classifier assumes that visual words are conditionally independent given object class

Csurka Bray, Dance & Fan, 2004
Naïve Bayes - prior

• Model for each object class:

\[ P(x \mid c) = \prod_{i=1}^{m} P(x_i \mid c) \]

• Class priors \( P(c) \) encode how likely we are to see one class versus others.

• Note that:

\[ \prod_{i=1}^{m} P(c) = 1 \]

Csurka Bray, Dance & Fan, 2004
Naïve Bayes - posterior

• With the equations from the previous slides, we can now calculate the probability that an image represented by \( x \) belongs to class category \( c \).

\[
P(c \mid x) = \frac{P(c) P(x \mid c)}{\sum_{c'} P(c') P(x \mid c)}
\]

Bayes Theorem
Naïve Bayes – posterior

• With the equations from the previous slides, we can now calculate the probability that an image represented by $x$ belongs to class category $c$.

$$P(c \mid x) = \frac{P(c) \ P(x \mid c)}{\sum_{c'} P(c') \ P(x \mid c')}$$

$$P(c \mid x) = \frac{P(c) \ \prod_{i=1}^{m} P(x_i \mid c)}{\sum_{c'} P(c') \ \prod_{i=1}^{m} P(x_i \mid c')}$$
Naïve Bayes - classification

• We can now classify that the image represented by $x$ is belongs class that has the highest probability:

$$c^* = \arg \max_c P(c \mid x)$$

$$c^* = \arg \max_c \log P(c \mid x)$$
Let’s break down the posterior

The probability that \( x \) belongs to class \( c_1 \):

\[
P(c_1 \mid x) = \frac{P(c_1) \prod_{i=1}^{m} P(x_i \mid c_1)}{\sum_{c'} P(c') \prod_{i=1}^{m} P(x_i \mid c')}
\]

And the probability that \( x \) belongs to class \( c_2 \):

\[
P(c_2 \mid x) = \frac{P(c_2) \prod_{i=1}^{m} P(x_i \mid c_2)}{\sum_{c'} P(c') \prod_{i=1}^{m} P(x_i \mid c')}
\]
Both their denominators are the same

The probability that $x$ belongs to class $c_1$:

$$P(c_1 \mid x) = \frac{P(c_1) \prod_{i=1}^{m} P(x_i \mid c_1)}{\sum_{c'} P(c') \prod_{i=1}^{m} P(x_i \mid c')}$$

And the probability that $x$ belongs to class $c_2$:

$$P(c_2 \mid x) = \frac{P(c_2) \prod_{i=1}^{m} P(x_i \mid c_2)}{\sum_{c'} P(c') \prod_{i=1}^{m} P(x_i \mid c')}$$
Both their denominators are the same

- Since we only want the max, we can ignore the denominator:

\[
P(c_1 \mid x) \propto P(c_1) \prod_{i=1}^{m} P(x_i \mid c_1)
\]

\[
P(c_2 \mid x) \propto P(c_2) \prod_{i=1}^{m} P(x_i \mid c_2)
\]
For the general class $c$, 

$$P(c \mid x) \propto P(c) \prod_{i=1}^{m} P(x_i \mid c)$$
For the general class $c$,

$$P(c \mid x) \propto P(c) \prod_{i=1}^{m} P(x_i \mid c)$$

We can take the log:

$$\log P(c \mid x) \propto \log P(c) + \sum_{i=1}^{m} \log P(x_i \mid c)$$
Naïve Bayes - classification

- So, the following classification becomes:

\[ c^* = \arg \max_c P(c \mid x) \]

\[ c^* = \arg \max_c \log P(c \mid x) \]

\[ c^* = \arg \max_c \log P(c) + \sum_{i=1}^{m} \log P(x_i \mid c) \]
Scene category dataset

Multi-class classification results
(100 training images per class)

<table>
<thead>
<tr>
<th>Level</th>
<th>Weak features (vocabulary size: 16)</th>
<th>Strong features (vocabulary size: 200)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Single-level</td>
<td>Pyramid</td>
</tr>
<tr>
<td>0 (1 × 1)</td>
<td>45.3 ±0.5</td>
<td>56.2 ±0.6</td>
</tr>
<tr>
<td>1 (2 × 2)</td>
<td>53.6 ±0.3</td>
<td>64.7 ±0.7</td>
</tr>
<tr>
<td>2 (4 × 4)</td>
<td>61.7 ±0.6</td>
<td>66.8 ±0.6</td>
</tr>
<tr>
<td>3 (8 × 8)</td>
<td>63.3 ±0.8</td>
<td>66.8 ±0.6</td>
</tr>
</tbody>
</table>
Caltech101 dataset


Multi-class classification results (30 training images per class)

<table>
<thead>
<tr>
<th>Level</th>
<th>Weak features (16)</th>
<th>Strong features (200)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Single-level</td>
<td>Pyramid</td>
</tr>
<tr>
<td>0</td>
<td>15.5 ± 0.9</td>
<td>32.8 ± 1.3</td>
</tr>
<tr>
<td>1</td>
<td>31.4 ± 1.2</td>
<td>49.3 ± 1.4</td>
</tr>
<tr>
<td>2</td>
<td>47.2 ± 1.1</td>
<td>54.0 ± 1.1</td>
</tr>
<tr>
<td>3</td>
<td>52.2 ± 0.8</td>
<td></td>
</tr>
</tbody>
</table>
Bags of features for action recognition

Space-time interest points

Bags of features for action recognition

What we have learned today

• Visual bag of words (BoW)
• Spatial Pyramid Matching
• Naïve Bayes