Homework 1 grade posted:
• you should have received an email.
• questions? email me and I will resolve it with the graders.

Today’s office hours are cancelled
• Instead having it tomorrow, i.e., Th 3:45 - 4:45 PM
• DLS speaker Richard Sutton (go to his talk instead)
Texture-related tasks

- **Shape from texture**
  - Estimate surface orientation or shape from image texture

- **Segmentation/classification** from texture cues
  - Analyze, represent texture
  - Group image regions with consistent texture

- **Synthesis**
  - Generate new texture patches/images given some examples
Recap: Filter bank
We can form a feature vector from the list of responses at each pixel.

\([r1, r2, \ldots, r38]\)
Given a set of observations \((x_1, x_2, \ldots, x_n)\), where each observation is a \(d\)-dimensional real vector, k-means clustering aims to partition the \(n\) observations into \(k\) (\(\leq n\)) sets \(S = \{S_1, S_2, \ldots, S_k\}\) so as to minimize the within-cluster sum of squares (WCSS). In other words, its objective is to find:

\[
\arg\min_S \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2
\]

where \(\mu_i\) is the mean of points in \(S_i\).

Easy to compute \(\mu\) given \(S\) and vice versa.

Lloyd’s algorithm for k-means

- Initialize k centers by picking k-points randomly
- Repeat till convergence (or max iterations)
  - Assign each point to the nearest center (assignment step)
  - Estimate the mean of each group (update step)

**MATLAB**  
```matlab  
[idx, c] = kmeans(X, k)  
```

- Simple, fast and works well in practice
- But can be unstable
  - Run multiple times and the best solution (one with the smallest WCSS)
  - Better initializations are possible (e.g. kmeans++)
Textons in images

image

convolution with f.b.

square

clustering into k=64 centers

cluster (k-means)

aggregate
Classifying materials, “stuff”

Global texton histogram is a good representation
Texture-related tasks

- **Shape from texture**
  - Estimate surface orientation or shape from image texture

- **Segmentation/classification** from texture cues
  - Analyze, represent texture
  - Group image regions with consistent texture

- **Synthesis**
  - Generate new texture patches/images given some examples
Texture synthesis

- **Goal**: create new samples of a given texture
- Many applications: virtual environments, hole-filling, texturing surfaces
The challenge

- Need to model the whole spectrum: from repeated to stochastic texture

Markov chains

Markov chain

- A sequence of random variables $x_1, x_2, \ldots, x_n$
- $x_t$ is the state of the model at time $t$

> $\xrightarrow{}$  

- Markov assumption: each state is dependent only on the previous one - dependency given by a conditional probability:

$$p(x_t|x_{t-1})$$

- The above is actually a first-order Markov chain
- An $N$th-order Markov chain:

$$p(x_t|x_{t-1}, \ldots, x_{t-N})$$
"A dog is a man’s best friend. It’s a dog eat dog world out there."

Markov Chain Example: Text

\[ p(x_t|x_{t-1}) \]

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>dog</th>
<th>is</th>
<th>man’s</th>
<th>best</th>
<th>friend</th>
<th>it’s</th>
<th>eat</th>
<th>world</th>
<th>out</th>
<th>there</th>
<th>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td></td>
<td>2/3</td>
<td></td>
<td>1/3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>dog</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>is</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>man’s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>best</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>friend</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>it’s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>eat</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>world</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>out</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>there</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

Source: S. Seitz
Text synthesis

Create plausible looking poetry, love letters, term papers, etc.

Most basic algorithm

1. Build probability histogram
   - find all blocks of N consecutive words/letters in training documents
   - compute probability of occurrence $p(x_t|x_{t-1}, \ldots, x_{t-(n-1)})$

2. Given words $x_1, x_2, \ldots, x_{k-1}$
   - compute $x_k$ by sampling from $p(x_t|x_{t-1}, \ldots, x_{t-(n-1)})$

Source: S. Seitz
“As I've commented before, really relating to someone involves standing next to impossible.”

“One morning I shot an elephant in my arms and kissed him.”

“I spent an interesting evening recently with a grain of salt.”

• What do we get if we extract the probabilities from a chapter on Linear Filters, and then synthesize new statements?

• This means we cannot obtain a separate copy of the best studied regions in the sum.
• All this activity will result in the primate visual system.
• The response is also Gaussian, and hence isn’t bandlimited.
• Instead, we need to know only its response to any data vector, we need to apply a low pass filter that strongly reduces the content of the Fourier transform of a very large standard deviation.
• It is clear how this integral exist (it is sufficient for all pixels within a $2k + 1 \times 2k + 1 \times 2k + 1 \times 2k + 1$ — required for the images separately.
A Markov random field (MRF)
• generalization of Markov chains to two or more dimensions.

First-order MRF:
• probability that pixel $X$ takes a certain value given the values of neighbors $A$, $B$, $C$, and $D$:

$$P(X|A, B, C, D)$$
Texture synthesis

Can apply 2D version of text synthesis

Texture corpus (sample)

Output

Efros & Leung, ICCV 99
Texture synthesis: intuition

• Before, we inserted the next word based on existing nearby words…

• Now we want to insert pixel intensities based on existing nearby pixel values.

Distribution of a value of a pixel is conditioned on its neighbors alone.
**Synthesizing one pixel**

- What is $P(x|\text{neighborhood of pixels around } x)$?
- Find all the windows in the image that match the neighborhood.
- To synthesize $x$
  - pick one matching window at random
  - assign $x$ to be the center pixel of that window
- An **exact** neighbourhood match might not be present, so find the **best** matches using **SSD error** and randomly choose between them, preferring better matches with higher probability.

Slide from Alyosha Efros, ICCV 1999
Neighborhood window

input
Varying window size

Increasing window size
Starting from the initial image, "grow" the texture one pixel at a time.
Synthesis results

french canvas

raffia weave

Slide from Alyosha Efros, ICCV 1999
Synthesis results

white bread

brick wall

Slide from Alyosha Efros, ICCV 1999
Failure Cases

Growing garbage

Verbatim copying
Extrapolation

Slide from Alyosha Efros, ICCV 1999
The Efros & Leung algorithm

- Simple
- Surprisingly good results
- Synthesis is easier than analysis!
- … but can be very slow
  - \([n \times m]\) image synthesis from \([p \times q]\) image requires \(n \times m \times p \times q\) patch comparisons
**Image Quilting [Efros & Freeman 2001]**

- **Observation:** neighbor pixels are highly correlated

**Idea:** unit of synthesis = block
- Exactly the same but now we want $P(B|N(B))$
- Much faster: synthesize all pixels in a block at once
Input texture

Random placement of blocks

Neighboring blocks constrained by overlap

Minimal error boundary cut
Minimal error boundary

overlapping blocks

vertical boundary

overlap error

min. error boundary

Slide from Alyosha Efros
Failures
(Chernobyl Harvest)
Texture transfer

• Take the texture from one object and “paint” it onto another object
  • This requires separating texture and shape
  • That’s hard, but we can cheat
  • Assume we can capture shape by boundary and rough shading

Then, just add another constraint when sampling: similarity to underlying image at that spot
parmesan + rice =

+ =

+ =
(Manual) texture synthesis in the media

Conclusion

• Texture is a useful property that is often indicative of materials, appearance cues

• **Texture representations** attempt to summarize repeating patterns of local structure

• **Filter banks** useful to measure redundant variety of structures in local neighborhood
  - Feature spaces can be multi-dimensional
  - Vector quantize to build histograms

• Neighborhood statistics can be exploited to “sample” or **synthesize** new texture regions
  - Example-based technique
Further thoughts and readings …

- Texture and human psychophysics
  - *N. Bhusan et al.*, The Texture Lexicon: Understanding the Categorization of Visual Texture Terms and Their Relationship to Texture Images [pdf]

- Texture representation
  - Are filter banks necessary? (Varma and Zisserman, CVPR 2003)
  - Local binary patterns (Ojala, Pietikainen, Maenpaa, PAMI 2002)

- State of the art in texture classification
  - Learning to detect describable attributes, e.g. lined, dotted, blotchy, striped, checkered, etc.