Outline

- D1 Starting toolbox for statistical recognition

- D2 Structured prediction
  - Hidden Markov model, Markov random field (MRF)
  - Inference problems, EM with MAR
  - Support Vector Machine

- D3 Learning for structured prediction
  - Structured output SVM, advanced examples
  - Cutting Plane methods
Introduction
Classification

- SVM era
  
  ![Salmon](image1) ![Sea Bass](image2)

  \[
  \{0, 1\} \rightarrow \{1, \ldots, K\}
  \]

- Deep NN era
Classification

- SVM era

\[ \{0, 1\} \quad \rightarrow \quad \{1, \ldots, K\} \]

- Design measurements, represent them as a feature vector

- Deep NN era
Classification

- SVM era

\[ \{0, 1\} \rightarrow \{1, \ldots, K\} \]

- Design measurements, represent them as a feature vector
- Learn the best discriminant function

- Deep NN era
Classification

- SVM era
  - Design measurements, represent them as a feature vector
  - Learn the best discriminant function

- Deep NN era
  - Learn some feature vector
Classification

- SVM era
  - Design measurements, represent them as a feature vector
  - Learn the best discriminant function

- Deep NN era
  - Learn some feature vector
  - Apply SVM
Structured Prediction

- Text Recognition
- Optical Structure Recognition
- Body Parts Segmentation
- Image Segmentation
- Facial Landmarks Detection

{space of text sentences}
Markov Models
Outline

- Statistical models over large (structured) state spaces
  - Conditional independences, p.d.f. factorization
- Hidden Markov model (chain)
  - Connection: recurrent NNs
- Markov random field, conditional random field
  - Connection: CNN, deep Bolzman machine
The Modeling Problem

- Consider a collection of random variables describing hidden state
  \[ x_1, x_2, \ldots, x_n, \quad x_i \in D \]

- Wish to have a statistical model:
  \[ p(x) = p(x_1, x_2, \ldots, x_n) \]

- But how to represent complicated function of many variables?
- Trivial observation for independent variables:
  - The distribution factors:
    \[ p(x) = p(x_1)p(x_2) \ldots p(x_n) \]
    - it is easy to evaluate maximize or integrate
  - Something in between?
Conditional Independence

- Conditional Independence
  - Example: smoke, fire, alarm
  - all 3 correlated, but
  - smoke $\Rightarrow$ fire and alarm are independent

- Conveniently represented in a graph diagram

$$\begin{align*}
  p(x_2, x_3 | x_1) &= p(x_2 | x_1)p(x_3 | x_1) \\
p(x_2 | x_1, x_3) &= p(x_2 | x_1)
\end{align*}$$

- Factorization: $p(x_1, x_2, x_3) = p(x_2 | x_1)p(x_3 | x_1)p(x_1)$

- A directed graphical model (Bayes Network)
A region is independent of the rest given some neighborhood.
Random Field

- Collection of random variables

\[ x_1, x_2, \ldots, x_n, \quad x_i \in D \]

**Definition**

\[ p: D^n \rightarrow \mathbb{R} \text{ is a random field if } p(x) > 0 \ \forall x, \sum_x p(x) = 1. \]

- Non-negativity is important for existence of conditional probabilities and other good reasons. Practically not a limitation.

**Definition**

\[ p \text{ is a Markov random field if it satisfies one or more conditional independence (Markov) properties.} \]

Undirected Graphical Model

- **Undirected Graphical Model**
  - Graph $G = (V, E)$
  - Set of nodes $V$: random variables $x_i, i \in V$
  - Set of edges $E$

- **Local Markov Property w.r.t. $G$**:
  - Given neighbors of $x_i$, it is independent of the rest:
    $$p(x_i \mid x_{\mathcal{N}(i)}) = p(x_i \mid x_{\mathcal{V}\setminus(i)}), \forall i \in V$$

- **Pairwise Markov Property w.r.t. $G$**:
  - Absent edge $(i, j)$ in $G$ iff $x_i$ and $x_j$ are conditionally independent given the rest of variables.

---

**Theorem (Lauritzen 96)**

*Local and Pairwise Markov Properties are equivalent.*

**Definition**

MRF w.r.t. graph $G$ is a random field satisfying Markov property w.r.t. $G$.
MRF factorization

- Conditional independencies help to structure and simplify the distribution

**Theorem (Hammersley-Clifford, 1971)**

\[ p \text{ MRF w.r.t. graph } G \text{ factors over cliques of } G: \ p(x) = \prod_{c \in C} f_c(x_c), \]

- \( C \) is the set of cliques – maximal fully connected subgraphs

**Definition**

\( p \) is a *Gibbs Random field* if it factors as \( p(x) = \prod_{c \subset S} f_c(x_c), \)

- A generalization, \( S \subset 2^V \)
- Here we do not need \( c \) to be a clique in some graph
Maximum a posteriori

- Given the model $p(x) = \prod_{c \in S} f_c(x_c)$ find the most probable state:
  $$\max_x p(x)$$

- Joint maximization in all variables
- Take negative logarithm:
  $$\min_x \sum_{c \in S} -\log f_c(x_c) = \min_x E(x)$$

- Partially separable minimization problem (Energy minimization)
- Converted to optimization domain (ILP, maximum cut, submodular function minimization, relaxations)
- Gibbs distribution: $p(x) = \exp(-E(x))$ – physics origins
Conditional Random Field

- $x_i, \ i \in V$ - hidden random variables (segmentation)
- $y_j, \ j \in V'$ - observed random variables (Image)

Definition (Lafferty et al. 01)

$p(x \mid y)$ is a conditional random field if it satisfies Markov properties w.r.t. $x$ given $y$.

MRF $p(x,y)$

CRF $p(x \mid y)$

Generative: $p(y) = \sum_x p(x, y)$
can be learned unsupervised

Recognition is the same: $\text{argmin}_x p(x, y) = \text{argmin}_x p(x \mid y)$

Discriminative, no model of $p(y)$
more flexible for recognition
Energy Minimization
Energy Minimization

\[
\min_x \sum_{i \in V} f_i(x_i) + \sum_{ij \in E} f_{ij}(x_i, x_j)
\]

\((V, E)\) - graph

\(V\) - set of nodes

\(E\) - set of edges

\(x = (x_i \mid i \in V)\) - labeling

- NP-hard (includes MAX-CUT, vertex packing, etc.)
- exp-APX-complete (approximation-preserving reduction from WSAT)
- Two large groups of methods:
  - minimum cut (graph cuts)
  - LP relaxation / message passing
- There are much more
Example: Potts Model for Object Class Segmentation

- $\mathcal{V}$ - set of pixels; $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ neighboring pixels;
- $\mathcal{X}_s = \{1, \ldots, K\}$ - class label;
- $E_f(x) = \sum_{s \in \mathcal{V}} f_s(x_s) + \sum_{st \in \mathcal{E}} \lambda_{st}[x_s \neq x_t]$.

Image

Ground Truth

(MSRC object class segmentation)
Minimum Cut

- Minimum s-t cut problem

  Capacitated network
  \[ G = (V, E, c), \]
  \[ c(u, v) \geq 0 - \text{arc capacities} \]

  ![Graph illustration](image)

  Cut cost: \[ \sum_{(u,v) \in E} c(u, v) \] \[ \rightarrow \min_{S} \] \[ \frac{s \in S}{s \in S} \] \[ t \notin S \]

  Source set \( S \)
  Cut \( (S, V \setminus S) \)

  Sink set \( T = V \setminus S \)

- Problem history: 30+ years
- Active research for better algorithms:
  - theoretical (Orlin’12: \( O(mn) \) algorithm), parallel algorithms
  - practical, esp. in computer vision
Let \( x_i \in \{0, 1\} \)

Energy minimization: \( \min_x \sum_{i \in V} f_i(x_i) + \sum_{ij \in E} f_{ij}(x_i, x_j) \)

Expand as polynomial:

\[
\begin{align*}
    f_i(x_i) &= f_i(1)x_i + f_i(0)(1 - x_i) = c_0 + c_i x_i; \\
    f_{ij}(x_i, x_j) &= \ldots = c'_0 + c'_i x_i + c''_j x_j + c_{ij} x_i (1 - x_j).
\end{align*}
\]

Minimum cut: \( \min_{S \subseteq V} \sum_{ij \in (S, V\setminus S)} c_{ij} \)

- Solvable in polynomial time if \( c_{uv} \geq 0 \)
Applications of min-cut

- Exemplar applications

Stereo
Boykov et al. 1998
Kolmogorov and Zabih 2001

Multiview Reconstruction
Lempitsky et al. 2006
Boykov and Lempitsky 2006

Surface Fitting
Lempitsky and Boykov 2007

3D Segmentation
Boykov and Joly 2001
Boykov and Funka-Lea 2006
Boykov and Kolmogorov 2003
Optimized Crossover

Current best solution $x$

Proposal solution $y$

Crossover (fusion problem) $x$

Local Search in some combinatorial locality
Expansion Move
Example: Stereo Reconstruction

- **Input**
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows

![Input Pair](image1.png)

![Disparity Map (GT)](image2.png)
**Example: Stereo Reconstruction**

- **Input**
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows

- **Problem**

![Input Pair](image_url)

![Disparity Map (GT)](image_url)
Example: Stereo Reconstruction

- **Input**
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows

- **Problem**
  - For each pixel in the left image find the corresponding pixel in the right image

Input Pair

Disparity Map (GT)
Example: Stereo Reconstruction

- **Input**
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows

- **Problem**
  - For each pixel in the left image find the corresponding pixel in the right image

Input Pair

Disparity Map (GT)
Example: Stereo Reconstruction

- **Input**
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows

- **Problem**
  - For each pixel in the left image find the corresponding pixel in the right image

- **Output**

![Input Pair](image1)

![Disparity Map (GT)](image2)
Example: Stereo Reconstruction

- **Input**
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows

- **Problem**
  - For each pixel in the left image find the corresponding pixel in the right image

- **Output**
  - Dense depth (disparity) map
Example: Scan-line Stereo

$i$ - pixel

$x_i$ - chosen disparity label

$x = (x_i \mid i \in V)$ - labeling

$f_i(x_i)$ - matching cost

$f_{ij}(x_i, x_j)$ - smoothness cost

$$\min_x \sum_{i \in V} f_i(x_i) + \sum_{ij \in E} f_{ij}(x_i, x_j)$$
Example: Scan-line Stereo

- **Trellis graph**

  ![Trellis graph diagram]

  - $i$ - pixel
  - $x_i$ - chosen disparity label
  - $x = (x_i | i \in V)$ - labeling
  - $f_i(x_i)$ - matching cost
  - $f_{ij}(x_i, x_j)$ - smoothness cost

- **Energy minimization**

  $$\min_{x} \sum_{i \in V} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j)$$
Hidden Markov Model

\[ p(x, y) = p(x_1) \prod_{i=2}^{n} p(x_i \mid x_{i-1}) \prod_{i=1}^{n} p(y_i \mid x_i) \]

- Conditionally independent model: given \( x_i \), \( y_i \) is independent of everything else.
- Recognition (MAP):

\[
\arg\max_x p(x, y) = \arg\min_x \sum_i f_i(x_i) + \sum_{i=2}^{n} f_{i-1,i}(x_{i-1}, x_i) \\

f_i(x_i) = - \log p(y_i \mid x_i)
\]
Problem:

\[
\min_x \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j)
\]

Use distributivity:

\[
\min(a + c, b + c) = \min(a, b) + c
\]

\[
\min_{x_n} \left( \cdots + \min_{x_2} \left( f_{2,3}(x_2, x_3) + f_2(x_2) + \min_{x_1} \left( f_{1,2}(x_1, x_2) + f_1(x_1) \right) \right) \right)
\]

Recurrent update:

\[
\varphi_j(x_j) = \min_{x_i} \left( f_{ij}(x_i, x_j) + f_i(x_i) + \varphi_i(x_i) \right)
\]

Shortest path from the left

Core of all message passing algorithms
Distance Transforms

- Recurrent update (message passing):

\[ \varphi_j(x_j) = \min_{x_i} (\varphi_i(x_i) + f_i(x_i) + f_{ij}(x_i, x_j)) \]

- Lower envelope (distance transform)

\[ f_{ij}(x_i, x_j) = w_{ij} \rho(x_i - x_j) \]

- \( O(nL^2) \) - naive approach, \( n \) variables, \( L \) labels

- \( O(nL) \) - efficient sequential algorithms [Hirata’96, Meijster’02] [Felzenszwalb&H.’06]

- \( O(n \log L) \) - efficient parallel algorithms, using \( L \) processors [Goodrich’86, Chen’02]

- Extends to trees
Here, the data term formulate the stereo matching task as finding a dis-
ing is addressed in section 2.6.

two different types of trees. Finally, occlusion han-
all pixels (as shown in figure

G. FACCIOLO, C. DE FRANCHIS, E. MEINHARDT: MORE GLOBAL MATCHING

2 More Global Matching

Let us consider the left-to-right direction. The image is traversed in raster order (left-to-

2.2 SGM and streaking artifacts

defines the complexity of the resulting optimization

individual scanline is then determined using DP. Such

aggregation is then performed by summing up the in-

smoothness terms. They enforce smoothness only

classical DP approaches (Bobick and Intille, 1999;

Tree-based DP proposed by (Veksler, 2005). (d) Approach

Hirschmüller-05 (SGM)
+ own tree for each pixel
+ reuse messages in DP

Facciolo et al.-15 (MGM)
+ combine more messages

Yang-15
+ with cost volume modification

Bleyer & Gelautz-VISSAP-08
+ own tree for each pixel
+ larger coverage

Veksler-05
Psota et al. ICCV-15
+ connect similar colors first
+ learned potentials

× VISSAP-08

2008 - International Conference on Computer Vision Theory and Applications
Max-Product BP, Tree-Reweighted

- Can Run Message passing in parallel
  
  $O(n)$ time, $O(n)$ processors

- Can apply on graphs with loops (loopy BP)
  
  - Over-counting
  - May oscillate
  - May diverge (unbounded)

- Tree-Reweighted [Wainwright’05]
  
  - Decomposition into trees
  - Connection to LP relaxation and its dual
  - Parallel algorithm may still oscillate

\[
d(i, j) := \min_k (d(i, k) + d(k, j))\]

(c.f. all shortest paths in a graph (Floyd–Warshall alg.))
\[ f(x) = \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j) \]

**Sum of chain subproblems:** \( f = f^1 + f^2 \)

\[
\min_x (f^1(x) + f^2(x)) = \min_{x^1 = x^2} (f^1(x^1) + f^2(x^2)) \\
= \min_{x^1, x^2} \max_\varphi \langle \varphi, x^1 - x^2 \rangle + f^1(x^1) + f^2(x^2) \\
\geq \max_\varphi [\min_{x^1} (f^1 + \varphi)(x^1) + \min_{x^2} (f^2 - \varphi)(x^2)]
\]

- **Convex**
- **Dual to the Schlesinger’s LP relaxation\(^1\)**

---

\(^1\) Shlezinger, (1976) “Syntactic analysis of two-dimensional visual signals in the presence of noise,” Cybernetics and Systems Analysis
Basic idea of duality

strong duality

weak duality
Equivalent Transforms

Minimum Cut / Maximum Flow

Linear Programming

\[
\min c^T x - d \\
Ax = b \\
x \geq 0
\]

\[
\begin{bmatrix}
c_1 & c_2 & c_3 & d \\
a_{11} & a_{12} & a_{13} & b_1 \\
a_{21} & a_{22} & a_{23} & b_2 \\
0 & 0 & \tilde{c}_3 & d \\
1 & 0 & \tilde{a}_{13} & \tilde{b}_1 \\
0 & 1 & \tilde{a}_{23} & \tilde{b}_2 \\
\end{bmatrix}
\]
Equivalent Transforms

Elementary Equivalent transform:

\[
\begin{align*}
x_1 & \rightarrow x_1' = x_1 + 1
\end{align*}
\]

Want to achieve the state:

\[
\begin{align*}
x_1 & \rightarrow x_1' = x_1 + 1
\end{align*}
\]

Equivalent Transformation Method, Primal-Dual
Dual Decomposition: Primal Solutions

\[
\varphi \left[ \min_{x^1} (f^1 + \varphi)(x^1) + \min_{x^2} (f^2 - \varphi)(x^2) \right]
\]

\(\varphi\) - Lagrange multiplier to \(x^1 = x^2\)

**Figure 1:** Grid structures of previous approaches. Nodes represent image pixels.

<table>
<thead>
<tr>
<th>Iterations</th>
<th>Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><img src="image1" alt="Image 1" /></td>
</tr>
<tr>
<td>2</td>
<td><img src="image2" alt="Image 2" /></td>
</tr>
<tr>
<td>5</td>
<td><img src="image5" alt="Image 5" /></td>
</tr>
<tr>
<td>500</td>
<td><img src="image500" alt="Image 500" /></td>
</tr>
</tbody>
</table>

\(x^1\) and \(x^2\) are independent but connected through the disparity cost function.

The algorithm proposed in this paper performs a separate disparity computation for each individual pixel.
### Table: Runtime analysis of the individual components of our stereo matching method (640×480, 128 labels).

<table>
<thead>
<tr>
<th></th>
<th>Cost Vol.</th>
<th>Discrete</th>
<th>Cont. Ref.</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>27 ms</td>
<td>73 ms (4 iterations)</td>
<td>39 ms</td>
<td>139 ms</td>
</tr>
</tbody>
</table>

**Figure:** Influence of continuous refinement on the reconstruction quality of KinectFusion.
Stereo Matching
Stereo Matching: Real-time fusion

Live reconstruction of a desktop scene
Stereo Matching: Real-time fusion

Live reconstruction of a desktop scene
GrabCut: Joint Segmentation and Parameter Estimation

Based on the work by Rother, Kolmogorov, Blake:
“GrabCut” — Interactive Foreground Extraction using Iterated Graph Cuts
Task 1: Joint Segmentation and Parameter Estimation

• Input:
  Image
  FG / BG brush

• Output:
  • Complete segmentation
Task 1, model

- Markov random field (generative) model:
- Segmentation $x: \Omega \rightarrow \{0, 1\}$
  - Model: $p(x)$ - neighboring pixels are more likely to take the same segment
- Color clusters: $k: \Omega \rightarrow \{1, \ldots, K\}$
  - Model: $p(k|x)$ - conditionally independent for all pixels
- Image: $I: \Omega \rightarrow \mathbb{R}^3$ - color drawn from a color cluster
  - Model: $p(I|k)$ - conditionally independent for all pixels
Task 1, model: segmentation

- **Segmentation** $x: \Omega \rightarrow \{0, 1\}$

  $$p(x) = \exp(-J(x)), \quad J(x) = \sum_{ij \in \mathcal{E}} \lambda |x_i - x_j|$$
Task 1, model: mixture components

- **Color clusters** \( k : \Omega \rightarrow \{1, \ldots K\} \)
  - Conditional independent model
    \[
    p(k \mid x) = \prod_{i \in \Omega} p(k_i \mid x_i)
    \]
  - \( p(k_i=\kappa \mid x_i=s) = \pi(\kappa \mid s) \) - mixture coefficients

Colors: \( p(l \mid k) \)

Mixture Components: \( p(k \mid x) \)

Segmentation: \( p(x) \)

BG appearance: \( \pi(\kappa \mid 0) \)
Task 1, model: colors

- **Colors**: $l : \Omega \rightarrow \mathbb{R}^3$
  - Conditional independent model
    $$p(l \mid x, k) = \prod_{i \in \Omega} p(l_i \mid k_i)$$
  - $p(l_i \mid k_i = \kappa) = \mathcal{N}(l_i; \mu_{\kappa}, \Sigma_{\kappa})$
- Parameters $\mu_{\kappa}, \Sigma_{\kappa}$ can be learned or pre-estimated for efficiency
Task 1, Gaussian Mixture View

- **Mixture Components:** \( p(k \mid x) \)

- **Segmentation:** \( p(x) \)

- **Colors:** \( p(l \mid k) \)

- **BG appearance:** \( \pi(\kappa \mid 0) \)

- **Efficient Color model:**

\[
p(l_i \mid x_i) = \sum_{k_i} p(l_i, k_i \mid x_i) = \sum_{k_i} p(l_i \mid k_i)p(k_i \mid x_i) \\
= \sum_{\kappa} p_N(l_i; \mu_{\kappa}, \Sigma_{\kappa})\pi(\kappa \mid x_i)
\]

- **Gaussian mixture**
Task 1, Derivation of EM Algorithm

- **Maximum Likelihood:**
  - Find segmentation $x$
  - estimate color models $\pi(\kappa | s)$
  - marginalize over hidden color clusters $k$

Colors: $p(l | k)$

Mixture Components: $p(k | x)$

Segmentation: $p(x)$

BG appearance: $\pi(\kappa | 0)$
Task 1, Derivation of EM Algorithm

- **Maximum log Likelihood** (for one image):

\[
\max_{x, \pi} \log \prod_{i \in \Omega} \sum_{\kappa} p_N(l_i; \mu, \Sigma) \pi(\kappa | x_i) p(x)
\]

- Of the form \(\max \prod \sum\), apply EM lower bound:

\[
\geq \sum_{i \in \Omega} \sum_{\kappa} \alpha(\kappa | x_i) \left( \log \left( p_N(\mu, \Sigma) + \log \pi(\kappa | x_i) \right) - \log \alpha(\kappa | x_i) \right) + \log p(x)
\]

- **E step:**

\[
\alpha(\kappa | x_i) \propto p_N(l_i; \mu, \Sigma) \pi(\kappa | x_i) \quad (1)
\]

- **M step:**

\[
x \in \arg\min_x J(x) - \sum_{i} \sum_{\kappa} \alpha(\kappa | x_i) \log \left( p(l_i \mid \kappa) \right) \pi(\kappa | x_i) \quad (2)
\]

\[
\pi(\kappa \mid s) \propto \sum_{i \mid x_i = s} \alpha(\kappa \mid s) \quad (3)
\]
Task 1, Overall Algorithm

- **EM: Iteratively reestimate**
  - Segmentation \( x: \Omega \rightarrow \{0, 1\} \) having appearance model \( \pi \) and probabilities of hidden components \( \alpha \)
  - Appearance models, \( \pi(\kappa \mid s) \)
  - Soft cluster assignment for each pixel, \( \alpha(\kappa \mid x_i) \)

Notice the difference to the following "ad-hoc" algorithm:

- **Ad-hock: Iteratively reestimate**
  - Segmentation \( x \) for current appearance model \( \pi \)
  - Appearance models \( \pi \) from current segmentation

The later method may be not converging and can get stuck more easily, similarly to K-means.
GrabCut TV version
Generative model (bottom-up in the picture)

- Segmentation $u : \Omega \rightarrow \{0, 1\}$
- Assignment of pixels to color clusters $k : \Omega \rightarrow \{1, \ldots, K\}$
- Image $I : \Omega \rightarrow \mathbb{R}^3$ – color drawn from Gaussian cluster $k$
Segmentation:

- Assume TV prior (neighboring pixels are more likely to be in the same segment)

\[ p(u) = \exp(-J(u)), \quad J(u) = \lambda \sum \| \nabla u(x) \|_2. \]
Graphical Model

Color cluster:
- Assume conditional independence

\[
p(k | u) = \prod_{x \in \Omega} p(k(x) | u(x));
\]

\[
p(k(x) = \kappa, u(x) = s) = \pi(\kappa | s) - \text{unknown appearance}
\]
**Image colors:**

- assume conditional i.i.d. given cluster \( k \),

\[
p(I(x) | k(x) = \kappa) = G_{\Sigma_{\kappa}}(I(x) - \mu_{\kappa});
\]

parameters \( \Sigma_{\kappa}, \mu_{\kappa} \) could be learned or preestimated for efficiency
Image colors are drawn from a mixture:

\[ p(I(x) \mid u(x) = s) = \sum_{\kappa} \pi(\kappa \mid s) G_{\Sigma_{\kappa}}(I(x) - \mu_{\kappa}); \]
Graphical Model

Maximum Likelihood:

- find segmentation $u$
- estimate color models $\pi(\kappa \mid s)$
- marginalize over latent color clusters $k$
Apply EM Algorithm

- Maximum likelihood:

\[
\arg\max_{u, \pi} \sum_{k} \prod_{x \in \Omega} p(l(x) \mid k(x)) \pi(k(x) \mid u(x)) p(u)
\]

- derive (blackboard)

\[
= \arg\max_{u, \pi} \prod_{s \in \{0, 1\}} \prod_{x \in \Omega \mid u(x) = s} \sum_{\kappa} p(l(x) \mid \kappa) \pi(\kappa \mid s) p(u),
\]

- to allow for linearization, express log likelihood as as

\[
\sum_{s \in \{0, 1\}} \sum_{x \in \Omega} (1 + s - u(x)) \log \sum_{\kappa = 1}^{K} p(l(x) \mid \kappa) \pi(\kappa \mid s) + \log p(u).
\]
Apply EM Algorithm

- EM lower bound:

\[
\sum_{s \in \{0,1\}} \sum_{x \in \Omega} \log \sum_{\kappa=1}^{K} \left( p(I(x) \mid \kappa) \pi(\kappa \mid s) \right)^{(1+s-u(x))} + \log p(u)
\]

introduce numbers \( \alpha_x(\kappa \mid s) \geq 0 \) such that \( \sum_{\kappa} \alpha_x(\kappa \mid s) = 1 \),

\[
\geq \sum_{s \in \{0,1\}} \sum_{x} \sum_{\kappa} \left( \alpha_x(\kappa \mid s)(1 + s - u(x)) \log[p(I(x) \mid \kappa) \pi(\kappa \mid s)] - \log \alpha_x(k(x) \mid s) \right) + \log p(u).
\]

- Bound valid for \( u : \Omega \to [0, 1] \)!
Maximization Step

- Maximization step in $u$ (blackboard):

$$u := \arg\max_u \sum_x g(x)u(x) + J(u),$$

$$g(x) = \sum_{\kappa=1}^{K} \left( \alpha_x(\kappa \mid 1) - \alpha_x(\kappa \mid 0) \right) \log[p(I(x) \mid \kappa)\pi(\kappa \mid s)].$$

(log likelihood ratio of FG and BG models with soft assignment $\alpha$)

- Maximization step in $\pi$ (blackboard):

$$\pi(\kappa \mid s = 0) \propto \sum_x \alpha_x(\kappa \mid s = 0)(1 - u(x)),$$

$$\pi(\kappa \mid s = 1) \propto \sum_x \alpha_x(\kappa \mid s = 1)u(x).$$
Expectation Step

(Maximize (tighten) bound in $\alpha$)

(blackboard):

$$\alpha_x(\kappa \mid 1) \propto u(x)p(l(x) \mid \kappa)\pi(\kappa \mid 1)$$
$$\alpha_x(\kappa \mid 0) \propto (1 - u(x))p(l(x) \mid \kappa)\pi(\kappa \mid 0),$$
Overall Algorithm

Iteratively reestimate

- Soft (hard) segmentation, $u : \Omega \rightarrow [0, 1]$ (resp. $u : \Omega \rightarrow \{0, 1\}$)
- Appearance models, $\pi(\kappa | s)$
- Soft cluster assignment of each pixel, $\alpha_x(\kappa | s) \in [0, 1]$