Computer Vision I - Algorithms and Applications: *Semantic Segmentation*

Carsten Rother
Roadmap this lecture (chapter 14.4.3, 5.5 in book)

- Interactive Image Segmentation
  - From Generative models to
    - Discriminative models to
      - Discriminative function

- Image Segmentation using GrabCut

- Semantic Segmentation
Roadmap this lecture (chapter 14.4.3, 5.5 in book)

- Interactive Image Segmentation
  - From Generative models to
    - Discriminative models to
    - Discriminative function

- Image Segmentation using GrabCut

- Semantic Segmentation
Probabilities - Reminder

• Discrete probability distribution: \( P(x) \) satisfies
\[
\sum_x P(x) = 1 \quad \text{where} \quad x \in \{0, \ldots, K\}
\]

• Joint distribution of two variables: \( P(x, z) \)

• Conditional distribution: \( P(x|z) \)

• Sum rule: \( P(z) = \sum_x P(x, z) \)

• Product rule: \( P(x, z) = P(z|x)P(x) \)

• Bayes’ rule: \( P(x|z) = \frac{P(z|x)P(x)}{P(z)} \)
Modelling a problem:
• The data is $z$ and the desired output $x$

We can identify three different approaches:
[see details in Bishop, page 42ff]:

• Generative (probabilistic) models: $P(x, z)$
• Discriminative (probabilistic) models: $P(x|z)$
• Discriminative functions: $f(x, z)$
Generative Model

Models explicitly (or implicitly) the distribution of the input $z$ and output $x$

Joint Probability $P(x, z) = P(z|x) \cdot P(x)$

\[ \text{likelihood prior} \]

Comment:
1. The joint distribution does not necessarily have to be decomposed into likelihood and prior, but in practice it (nearly) always is
2. Generative Models are used successfully when input $z$ and output $x$ are very related, e.g. image denoising.

Pros:
1. Possible to sample both: $x$ and $z$
2. Can be quite easily used for many applications (since prior and likelihood are modeled separately)
3. In some applications, e.g. biology, people want to model likelihood and Prior explicitly, since the want to understand the model as much possible
4. Probability can be used in bigger systems

Cons:
1. might not always be possible to write down the full distribution (involves a distribution over images $z$).
Generative Model – Example De-noising

Joint Probability \( P(z, x) = P(z|x) \cdot P(x) \)

Pixel-wise likelihood: \( P(z|x) = \prod_i N(x_i; z_i, \sigma) \sim \prod_i \exp\left\{ \frac{1}{2\sigma^2} (z_i - x_i)^2 \right\} \)

Data \( z \) (pixel independent Gaussian noise)

Label \( x \)

\( N(x_i; z_i, \sigma) \) (sketched)
Generative Model – Example De-noising

Joint Probability $P(z, x) = P(z|x) P(x)$

Pixel-wise likelihood: $P(z|x) = \prod_i N(x_i; z_i, \sigma) \sim \prod_i \exp\left\{ \frac{1}{2\sigma^2} (z_i - x_i)^2 \right\}$

Prior: $P(x) = \frac{1}{f} \exp\left\{ -\sum_{ij \in N_4} |x_i - x_j| \right\}$

Robust Prior:

$P(x) = \frac{1}{f} \exp\left\{ -\sum_{ij \in N_4} \min(-|x_i - x_j|, \tau) \right\}$

Follows the statistic of gradients in natural images

"sketched"
Result of more advanced prior models

(a) original

(b) noisy input

(c) pairwise MRF

(d) 3rd-order MRF

[Komodiakis et al. CVPR 2009]
Result of more advanced prior models

Figure 7: Denoising with a Field of Experts: Full image (top) and detail (bottom). (a) Original noiseless image. (b) Image with additive Gaussian noise ($\sigma = 25$); PSNR = 20.29dB. (c) Denoised image using a Field of Experts; PSNR = 28.72dB. (d) Denoised image using the approach of Portilla et al. (2003); PSNR = 28.90dB. (e) Denoised image using non-local means (Buades et al., 2004); PSNR = 28.21dB. (f) Denoised image using standard non-linear diffusion; PSNR = 27.18dB.

Leraned Prior on $5 \times 5$ patch

[Field of Expert, Roth et al IJCV 2008]
Figure 10: **Inpainting with a Field of Experts.** (a) Original image with overlaid text. (b) Inpainting result from diffusion algorithm using the FoE prior. (c) Close-up comparison between a (left), b (middle), and the results of Bertalmío et al. (2000) (right).
Joint Probability $P(z, x) = P(z|x) P(x)$

Pixel-wise likelihood: $P(z|x) = \prod_i N(x_i; z_i, \sigma) \sim \prod_i \exp\left\{ \frac{1}{2\sigma^2} (z_i - x_i)^2 \right\}$

Pixel-wise likelihood: $P(z|x) = \text{const}$ (for red text)

$P(z|x) = \delta(x_i = z_i)$ (otherwise)
Generative Model for image segmentation

Interactive Segmentation

\[ z = (R, G, B)^n \]

**Goal**

\[ x = \{0, 1\}^n \]

(user-specified pixels are not optimized for)

Given \( z \); derive binary \( x \):

Statistical model \( P(x, z) \) for both images \( z \) and data \( x \)

Optimal solution: \( x^* = \arg\max_x P(x, z) \) for a fixed \( z \)

(we then later come to \( P(x|z) \) and \( f(x, z) \))
Generative Model for image segmentation - likelihood

Joint Probability \( P(z, x) = P(z|x) \cdot P(x) \)

- **likelihood**
- **prior**

The red brush strokes give training data for foreground pixels
The blue brush strokes give training data for background pixels
Gaussian Mixture Model (GMM)

• Mixture Model: \( p(z) = \sum_{k=1}^{K} p(k) \ p(z|k) \)

• “\( k \)” is a latent variable we are not interested in

• \( k \in \{1, \ldots, K\} \) represents the \( K \) mixtures.

• Each mixture \( k \) is a 3D Gaussian distribution \( N_k(z; \mu_k, \Sigma_k) \) where \( \mu_k \), is a 3d vector and \( \Sigma_k \) a \( 3 \times 3 \) matrix (positive-semidefinite) called covariance matrices:

\[
N(z, \mu, \Sigma) = \frac{1}{(2\pi)^{p/2} |\Sigma|^{1/2}} \exp\left\{ -\frac{1}{2} (z - \mu)^T \Sigma^{-1} (z - \mu) \right\}
\]

• \( p(z) = \sum_{k=1}^{K} \pi_k \ N_k(z; \mu_k, \Sigma_k) \)

Mixture coefficient
Gaussian Mixture Model (GMM)

- GMM probability \( p(z) = \sum_{k=1}^{K} \pi_k \mathcal{N}_k(z; \mu_k, \Sigma_k) \)
- Unknown parameters: \( \Theta = (\pi_1, \ldots, \pi_K, \mu_1, \ldots, \mu_K, \Sigma_1, \ldots, \Sigma_K) \)

Example:

\[
\begin{align*}
\pi_1 &= \frac{4}{5} \\
\pi_2 &= \frac{1}{5} \\
\mathcal{N}_1 &= \text{GMM: } p(x) = \frac{4}{5} \mathcal{N}_1(x_1, \mu_1, \Sigma_1) + \frac{1}{5} \mathcal{N}_2(x, \mu_2, \Sigma_2)
\end{align*}
\]

- How to learn \( \Theta \) given data \( z \):
  - Maximum Likelihood estimation using EM (see machine learning lecture ML 1)
- Next: simpler version using to learn GMMs (close to k-means)
Let us introduce an assignment variable for each data point (pixel) to which Gaussian it belongs to: \( k_1, \ldots k_n \) where \( k_i \in \{1, \ldots K\} \)

\[
k_i = \arg\max_k N_k(z; \mu_k, \Sigma_k)
\]
Extensions of K-means

• Choose $K$ automatically

• Go to probabilistic version using Expectation Maximization (EM). Now $k_i$ are probabilistic assignments to all Gaussian (not only one)

• Faster versions:
  • Fit GMM to all data points and then only change the mixture coefficients
  • Use Histograms instead of GMMs
Illustration EM

Soft assignment: $p(a_i)$

[Bishop page 437]
Some comments on clustering

• Clustering without spatial constraints: (K-means, mean-shift, etc)

• Clustering with spatial constraints: (super-pixels, normalized cut, etc)

• Gestalt Theory:

• More in CV2
Joint Probability - Likelihood

Joint Probability \( P(z, x) = P(z|x) \ P(x) \)

Likelihood:

\[
P(z|x) = \prod_i P(z_i|x_i) = \prod_i \left( \sum_{k=1}^{K} \pi_k^{x_i} N_k^{x_i}(z_i, \mu_k^{x_i}, \Sigma_k^{x_i}) \right)
\]

All parameters with superscript 0 belong to background and all with superscript 1 belong to foreground:

\[
\Theta = (\pi_1^0, \ldots, \pi_K^0, \mu_1^0, \ldots, \mu_K^0, \Sigma_1^0, \ldots, \Sigma_K^0, \pi_1^1, \ldots, \pi_K^1, \mu_1^1, \ldots, \mu_K^1, \Sigma_1^1, \ldots, \Sigma_K^1)
\]
Joint Probability - Likelihood

Maximum likelihood estimation

\[ x^* = \arg\max_x P(z|x) = \prod_i P(z_i|x_i) \]
Joint Probability - Prior

Joint Probability $P(z, x) = P(z|x) \cdot P(x)$

$$P(x) = \frac{1}{f} \prod_{i,j \in N_4} \Theta_{i,j}(x_i, x_j)$$

$$f = \sum_x \prod_{i,j \in N_4} \Theta_{i,j}(x_i, x_j)$$  \text{Partition function, sum over all possible results } x

$$\Theta_{i,j}(x_i, x_j) = \exp\{-|x_i - x_j|\}$$  \text{called “Ising prior”}

(exp{-1}=0.36; \exp{0}=1)
Joint Probability – Prior (4x4 Grid example)

Pure Prior model: \( P(x) = \frac{1}{f} \prod_{i,j \in N_4} \exp\{-|x_i - x_j|\} \)

Best Solutions sorted by probability

Worst Solutions sorted by probability

“Smoothness prior needs the likelihood”
Joint Probability – Prior (4x4 Grid example)

Pure Prior model: \( P(x) = \frac{1}{f} \prod_{i,j \in N_4} \exp\{-|x_i - x_j|\} \)

Distribution

2\(^{16}\) configurations

Samples
Joint Probability – Result

Joint Probability $P(z, x) = P(z|x) \cdot P(x) = \prod_{i} \left( \sum_{k=1}^{K} \prod_{i}^{x_i} N_{Z_i} \left( z_i, \mu_{k_i}, \Sigma_{k_i} \right) \right) \frac{1}{f} \prod_{i,j \in N_4} \exp\{-|x_i - x_j|\}$

Global optimum
$x^* = \text{argmax}_x P(z, x)$

ML solution:
$x^* = \text{argmax}_x P(z|x)$

Hard constraint:

- $P(x_i = 0) = 0; P(x_i = 1) = 1$;
- $P(x_i = 0) = 1; P(x_i = 1) = 0$.
Sample from the model

\[ P(z, x) = P(z|x) \, P(x) \]

**Samples:**

- True image:
- Most likely:

**VLD**

28/01/2014  Computer Vision I: Semantic Segmentation
Why does it still work?

• We only evaluate $x$ for a given $z$

Global optimum

other likely solutions will look similar (sketched)
Best Prior Models for Images

Simple model for segmentations:

\[ P(x) \]

Looks good on texture level but not on global level (e.g. scene layout)

Remind denoising: \[ P(z, x) = P(z|x) \cdot P(x) \]

Pixel-wise likelihood: \[ P(z|x) = \prod_i N(x_i; z_i, \sigma) \sim \prod_i \text{exp} \left\{ \frac{1}{2\sigma^2} (z_i - x_i)^2 \right\} \]

Best prior models for images \[ P(x) \] can give such results:
Is it the best we can do?

4-connected segmentation

zoom zoom

Zoom-in on image
Reminder: Going to 8-connectivity

Larger connectivity can model true Euclidean length (also other metric possible)

Length of the paths:

<table>
<thead>
<tr>
<th></th>
<th>Eucl.</th>
<th>4-con.</th>
<th>8-con.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5.65</td>
<td>6.28</td>
<td>5.08</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>6.28</td>
<td>6.75</td>
</tr>
</tbody>
</table>

[Boykov et al. ‘03; ‘05]
Going to 8-connectivity

4-connected Euclidean

8-connected Euclidean (MRF)

Zoom-in image

[Boykov et al. ‘03; ‘05]
Modelling edges

- A transition is likely when two neighboring pixels have different color.

- How to put this into our model?
  - $P(x)$ cannot depend on data!
  - $P(z|x) = \prod_{i,j \in N_4} P(z_{ij}|x_{ij})$ must be extended to model all possible pairwise transitions from training data (e.g. with 6D Gaussian). But:
    - This is difficult for the user to label
    - Hard to get from other images
  - There is a much simpler way: model only $P(x|z)$
Half way slide

3 Minutes break
Roadmap this lecture (chapter 14.4.3, 5.5 in book)

- Interactive Image Segmentation
  - From Generative models to
  - Discriminative models to
  - Discriminative function

- Image Segmentation using GrabCut

- Semantic Segmentation
Models that model the Posterior directly are discriminative models. In Computer Vision we use mostly the Gibbs distribution with an Energy $E$:

$$P(x|z) = \frac{1}{f} \exp\{-E(x, z)\} \text{ where } f = \sum_x \exp\{-E(x, z)\}$$

These are also called: “Conditional random field”

**Pros:**
1. Simpler to write down than generative model
   (no need to model $z$)
   and goes directly for the desired output $x$
2. More flexible since energy is arbitrary
3. Probability can be used in bigger systems

**Cons:**
we can no longer sample images $z$
Discriminative model

• Relation: Posterior and Joint: \( P(x|z) = \frac{1}{P(z)} P(x, z) \)

• \( P(x, z), P(x|z) \) and \( E(x, z) \) all have the same optimal solution \( x^* \) given \( z \):
  • \( x^* = \arg\max_x P(x, z) \) given \( z \)
  • \( x^* = \arg\max_x P(x|z) \) given \( z \) (since \( P(x|z) = \frac{1}{P(z)} P(x, z) \))
  • \( x^* = \arg\min_x E(x, z) \) (since \( -\log P(x|z) = \log f + E(x, z) \))
How does $E$ looks like for our segmentation example?

- So that $P(x|z), P(x,z)$ have the same optimal solution $x^*$ we need:

$$P(x,z) \sim P(x|z) = \frac{1}{f} \exp \{-E(x,z)\}$$

$$-\log P(x,z) = E(x,z) + \text{constant} = \sum_i \Theta_i(x_i,z) + \sum_{i,j \in N_4} \Theta_{ij}(x_i,x_j,z) + \text{constant}$$

$$-\log P(x,z) = -\log \left( \prod_i P(z_i|x_i) \right) - \log \left( \frac{1}{f} \sum_{i,j \in N_4} \exp \left[ \Theta(x_i,x_j,z) \right] \right)$$

$$= \sum_i -\log P(z_i|x_i) + \log f + \sum_{i,j \in N_4} |x_i - x_j|$$

$$= \sum_i \left( \sum_{k=0}^K \pi_k N_k^0(z_i,N_k^0,\Sigma_k^0) \right) (1-x_i) + \sum_{i,j \in N_4} |x_i - x_j| + \text{constant}$$

with $P(z_i|x_i=0) = \sum_{k=1}^K \frac{\pi_k}{\Sigma_k} N_k^0(z_i,N_k^0,\Sigma_k^0)$

$P(z_i|x_i=1) = \sum_{k=1}^K \frac{\pi_k}{\Sigma_k} N_k^1(z_i,N_k^1,\Sigma_k^1)$
One may also write the joint distribution $P(x, z)$ as a Gibbs distribution:

$$ P(x, z) = \frac{1}{f} \exp\{-E(x, z)\} \quad \text{where} \quad f = \sum_{x,z} \exp\{-E(x, z)\} $$

But ... it lost the meaning of a “generative” model, since we don’t have a likelihood which says how the data was “generated”.

If likelihood and prior are no longer modelled separately:

- sampling $x, z$ gets very difficult
- We can no longer learn prior and likelihood separately (as in de-noising)
- We train $P(x, z) = \frac{1}{f} \exp\{-E(x, z)\}$, $P(x|z) = \frac{1}{f} \exp\{-E(x, z)\}$ in a similar way.

(see CV 2 lectures)

The advantage of a generative model over a discriminative model are gone
Adding a contrast term

\[ E(x, z) = \sum_i \Theta_i(x_i, z) + \sum_{i,j \in N_4} \Theta_{ij}(x_i, x_j, z) \]

\[ \Theta_{ij}(x_i, x_j, z) = |x_i - x_j| \left( \exp \left\{ -\beta (z_i - z_j)^2 \right\} \right) \]

\[ \beta = \frac{2}{|N_4|} \left( \sum_{ij \in N_4} \|z_i - z_j\|_2^2 \right)^{-1} \]
Roadmap this lecture (chapter 14.4.3, 5.5 in book)

- Interactive Image Segmentation
  - From Generative models to
  - Discriminative models to
  - Discriminative function

- Image Segmentation using GrabCut

- Semantic Segmentation
Discriminative functions

Models that model the classification problem via a function

\[ E(\mathbf{x}, \mathbf{z}) : K^n \rightarrow R \quad x^* = \text{argmin}_x E(\mathbf{x}, \mathbf{z}) \]

Examples:
- Energy
- Support vector machines
- Nearest neighbour classifier

Pros: most direct approach to model the problem

Cons: no probabilities

This is the most used approach in computer vision!
Modelling a problem:
- The input data is $z$ and the desired output $x$

We can identify three different approaches:
[see details in Bishop, page 42ff]:

- Generative (probabilistic) models: $P(x, z)$
- Discriminative (probabilistic) models: $P(x|z)$
- Discriminative functions: $f(x, z)$

The key difference are:
- Probabilistic or none-probabilistic model
- Generative models model also the data $z$
- Differences in Training (see CV 2)
Simple example: Learning Discriminative functions

\[
E(x, z) = \sum_{i} \Theta_i(x_i, z) + \omega \sum_{i,j \in N_4} \Theta_{ij}(x_i, x_j, z)
\]
**Training phase:** infer $\omega$ given a set of training images

$$\{x^t, z^t\} \rightarrow \omega$$

where $t$ denotes all training images (here around 50 images)

**Testing phase:**

$$Z, \omega \rightarrow x$$

$t = 1$

$t = 2$
A simple procedure: Learning Discriminative functions

1. Iterate $\omega = 0, \ldots, 500$
2. Compute $x^{*,t}$ for all training images $\{x^t, z^t\}$
3. Compute average error $Error = \frac{1}{T} \sum_t C(x^t, x^{*,t})$
   with loss/cost function: $C(x, x') = \sum_i |x_i - x_i'|$ (called Hamming Error)
4. Take $\omega$ with smallest $Error$

**Questions:**
- Is it the best and only way?
- Can we over-fit to training data?
Probabilistic Learning (for generative, discriminative models):

1. **Training**: Fit distribution $P(x, z)$, or $P(x|z)$ to set training images (use e.g. maximum likelihood learning)
2. **Test**: Make a decision according to some cost (loss) function $\Delta$ (depending on the cost function one computes optimal solution or marginal, etc)

Loss-based Learning (for discriminative functions)

1. **Training**: Fit $f(x, z)$ given a certain cost (loss) function (see above)
2. **Test**: Compute optimal value $x^*$ of function wrt test image

These are only high-level comments, we dive into that in CV 2!
Roadmap this lecture (chapter 14.4.3, 5.5 in book)

• Interactive Image Segmentation
  • From Generative models to
    • Discriminative models to
    • Discriminative function

• Image Segmentation using GrabCut

• Semantic Segmentation
Are we done?

[GrabCut, Rother et al. Siggraph 2004]
GrabCut Segmentation

\[ E(x) = \sum_i \Theta_i(x_i) + \omega \sum_{i,j \in N_4} w_{ij} |x_i - x_j| \]

Hard constraint:

\[ \Theta_i(x_i = 0) = \infty; \Theta_i(x_i = 1) = 0 \]

\[ \Theta_i(x_i = 0) = 0; \Theta_i(x_i = 1) = \infty \]

How to prevent the trivial solution?

[Boykov & Jolly ICCV ‘01]
What is a good segmentation?

Objects (foreground and background) are self-similar with respect to appearance.

**Option 1**
- Foreground
- Background

\[ E_{\text{unary}} = 460000 \]

\[ \Theta^F \]

\[ \Theta^B \]

**Option 2**
- Foreground
- Background

\[ E_{\text{unary}} = 482000 \]

\[ \Theta^F \]

\[ \Theta^B \]

**Option 3**
- Foreground
- Background

\[ E_{\text{unary}} = 483000 \]

\[ \Theta^F \]

\[ \Theta^B \]

**Equation**

\[
E_{\text{unary}}(x, \Theta^F, \Theta^B) = -\log P(z|x, \Theta^F, \Theta^B) = \sum_i -\log P(z_i|\Theta^F, x_i = 1)x_i - \log P(z_i|\Theta^B, x_i = 0)(1 - x_i)
\]
Gibbs distribution with energy:

\[
E(x, \Theta^F, \Theta^B) = \sum_i - \log P(z_i | \Theta^F, x_i = 1) x_i - \log P(z_i | \Theta^B, x_i = 0) (1 - x_i) + \sum_{ij} (-\exp\{-\beta ||z_i - z_j||\}) |x_i - x_j| \]

\[
P(z_i | x_i = 0, \Theta^B) = \sum_{k=1}^{K} \pi_k^B N_k^B (z_i, \mu_k^B, \Sigma_k^B)
\]

\[
P(z_i | x_i = 1, \Theta^F) = \sum_{k=1}^{K} \pi_k^F N_k^F (z_i, \mu_k^F, \Sigma_k^F)
\]

Goal is to compute optimal solution (we could also marginalize over \(\Theta\)):

\[
x^* = \text{argmax}_x \left( \max_{\Theta^F, \Theta^B} E(x, \Theta^F, \Theta^B) \right)
\]

- So far, \(\Theta\) was determined from brush strokes (training data)
- Now we estimate \(\Theta\) from the segmentation \(x\)
Full GrabCut functional

Goal is to compute optimal solution:

\[ x^* = \arg\max_x \left( \max_{\theta^F, \theta^B} E(x, \theta^F, \theta^B) \right) \]

**Problem:** Joint optimization of \(x, \theta^F, \theta^B\) is NP-hard

Comment: Using histograms for Color models one can transform the problem to a higher-order Random Field Model which can be solved (sometimes) globally optimal for all unknowns: segmentation \(x\) and \(\theta\) with Dual-Decomposition, [Vicente, Kolmogorov, Rother, ICCV '09] (see CV 2)
GrabCut - optimization

Initial segmentation

\[ \min_{\theta^F, \theta^B} E(x, \theta^F, \theta^B) \]

GMM fitting to the current segmentation

\[ \min_{x} E(x, \theta^F, \theta^B) \]

Graph cut to infer segmentation

Image \( z \) and user input

Initial segmentation \( x \)
GrabCut - Optimization

Result

Energy after each Iteration

0 1 2 3 4
GrabCut - Optimization

At initialization

In the end
Comparison

<table>
<thead>
<tr>
<th>Magic Wand</th>
<th>Intelligent Scissors</th>
<th>Bayes Matte</th>
<th>Knockout 2</th>
<th>Graph cut</th>
<th>GrabCut</th>
</tr>
</thead>
</table>

*Input image*
Roadmap this lecture (chapter 14.4.3, 5.5 in book)

- Interactive Image Segmentation
  - From Generative models to
    - Discriminative models to
      - Discriminative function

- Image Segmentation using GrabCut

- Semantic Segmentation
The desired output

Label each pixel with one out of 21 classes

[TextonBoost; Shotton et al, ‘06]
Failure cases

[TextonBoost; Shotton et al, ‘06]
TextonBoost: How it is done

Define Energy: $x_i \in \{1, ..., K\}$

$$E(x, \Theta) = \sum_i \theta_i(x_i, z_i, \Theta) + \theta_i(x_i) + \theta_i(x_i, z) + \sum_{i,j} \theta_{i,j}(x_i, x_j)$$

- **(color model)**
- **(location prior)**
- **(edge aware smoothness prior)**

As in GrabCut, each object has an associate GMM

$$\theta_{ij}(x_i, x_j) = w_{ij}|x_i - x_j|$$

Location Prior:

- sky
- grass

**class information:**

Each pixel gets a distribution over 21-classes:

$$\theta_i(x_i = c, z) = P(x_i = c | z)$$

- Using boosting – explained next lecture
- Using Random Forest – explained next

[TextonBoost; Shotton et al, ‘06]
TextonBoost: Energy

(a) Class and location only
(b) 69.6%
(c) 70.3%
(d) 72.2%
+ edges
+ color model

<table>
<thead>
<tr>
<th>Object classes</th>
<th>Building</th>
<th>Grass</th>
<th>Tree</th>
<th>Cow</th>
<th>Sheep</th>
<th>Sky</th>
<th>Aeroplane</th>
<th>Water</th>
<th>Face</th>
<th>Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bike</td>
<td>Flower</td>
<td>Sign</td>
<td>Bird</td>
<td>Book</td>
<td>Chair</td>
<td>Road</td>
<td>Cat</td>
<td>Dog</td>
<td>Body</td>
<td>Boat</td>
</tr>
</tbody>
</table>
Roadmap this lecture (chapter 14.4.3, 5.5 in book)

- Interactive Image Segmentation
  - From Generative models to
  - Discriminative models to
  - Discriminative function

- Image Segmentation using GrabCut

- Semantic Segmentation