Segmentation with Machine Learning / Introduction to ML methods

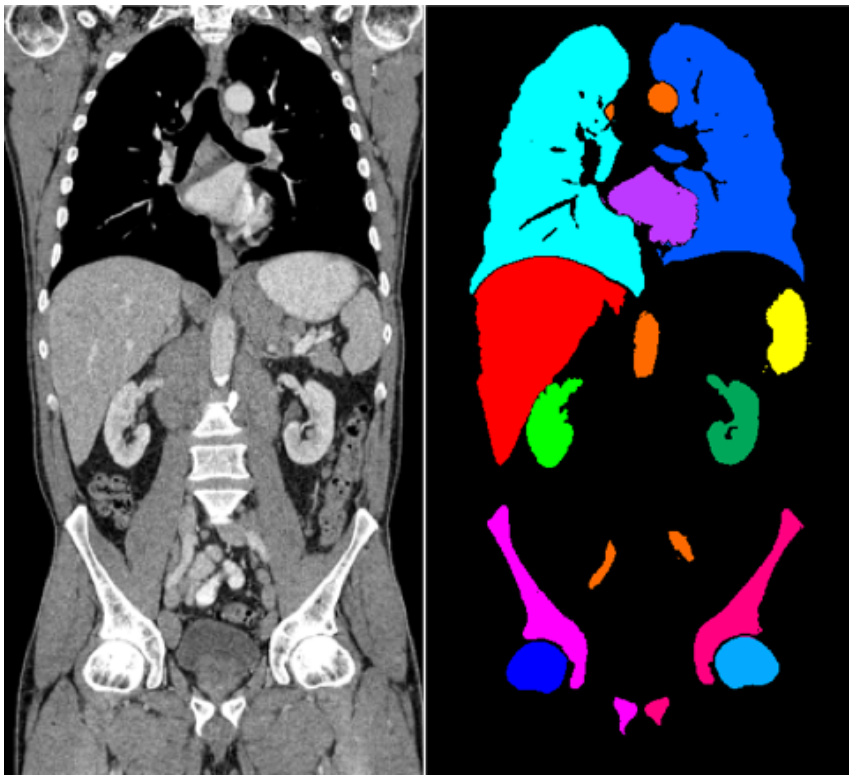
Ender Konukoglu

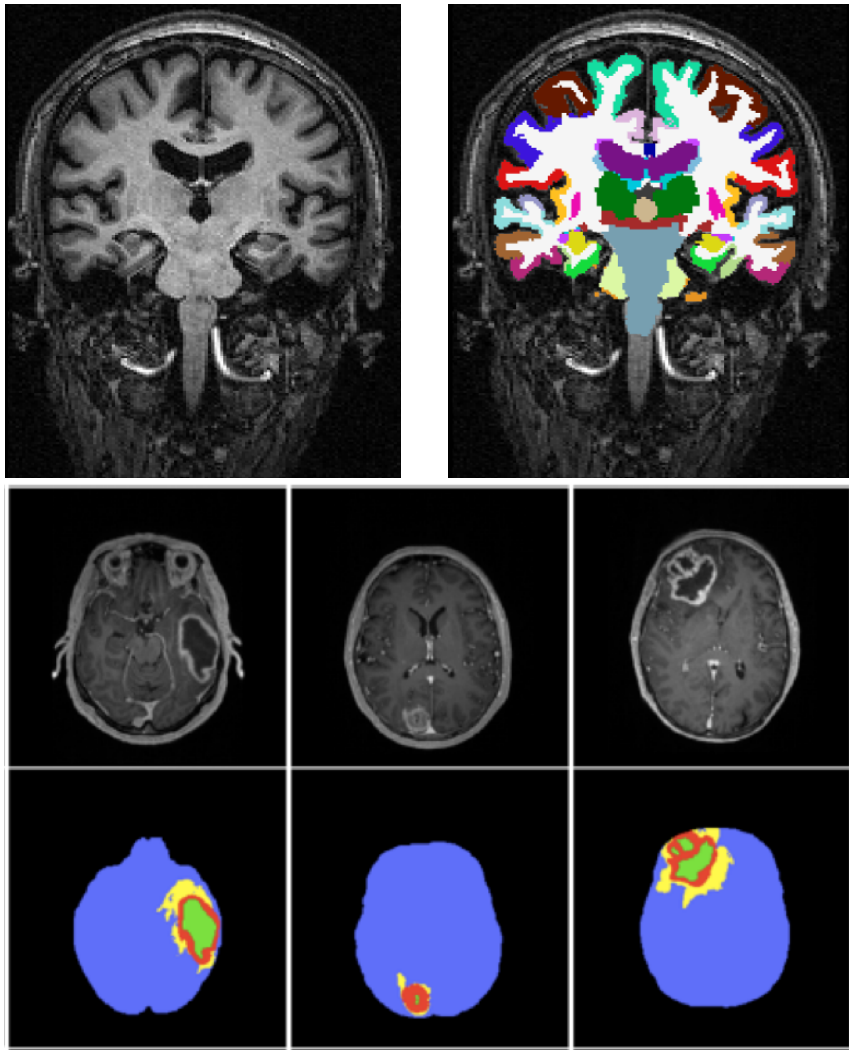
ETH Zürich

April 8, 2020

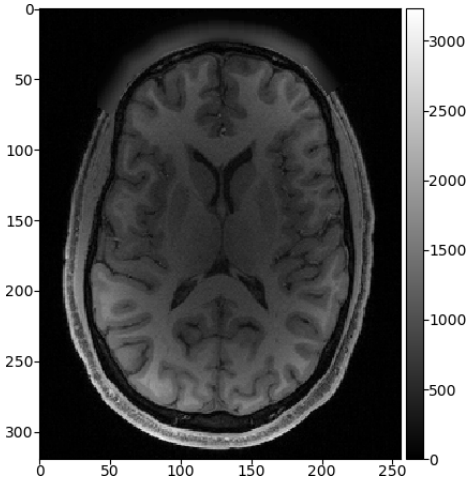
Segmentation

From image intensities to anatomical structures and semantic information

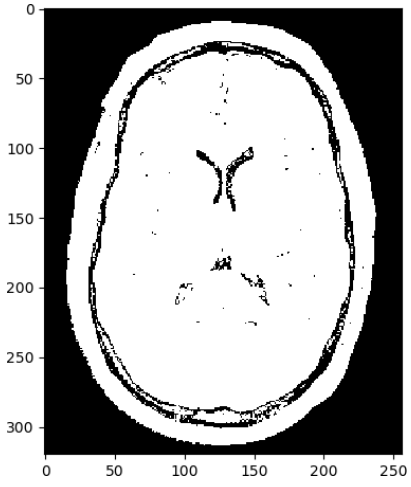




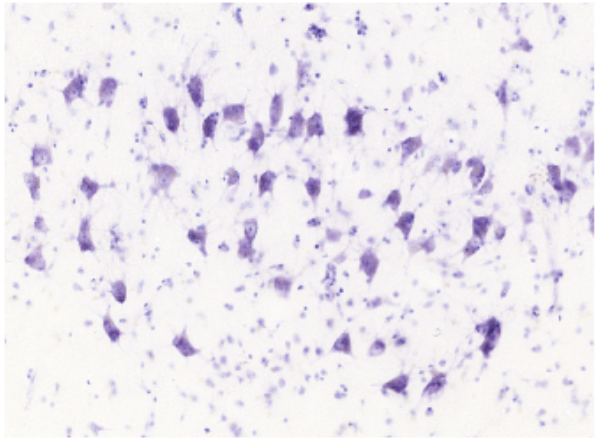
Where were we? - Thresholding



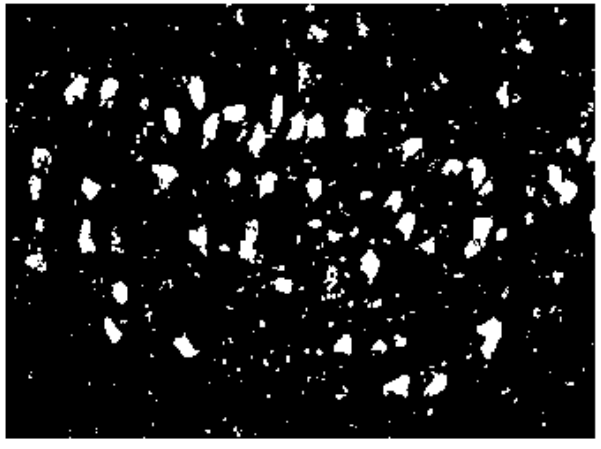
Image



thresholded

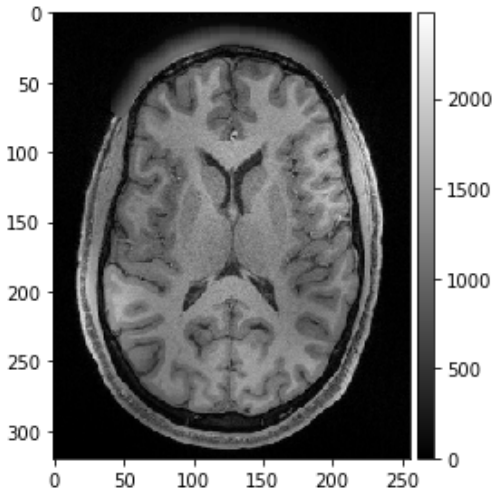


Image

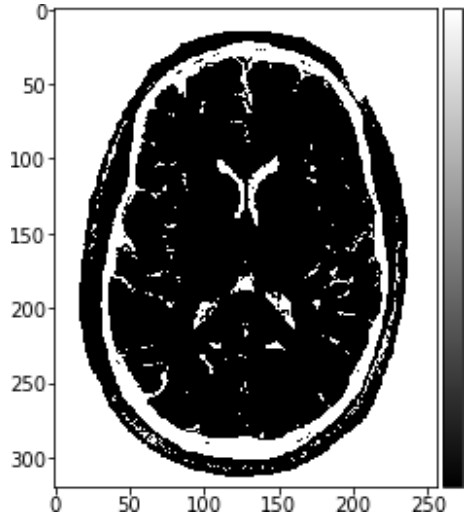


thresholded

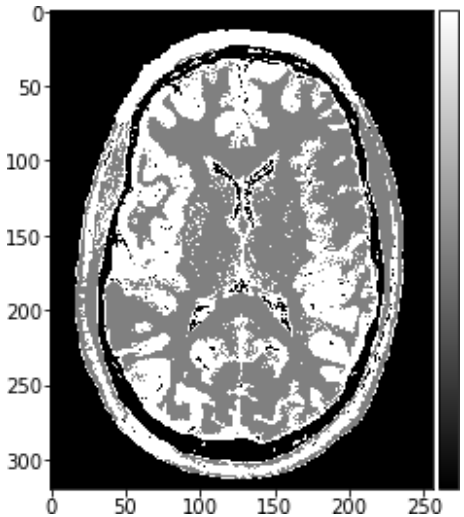
Where were we? - K-Means



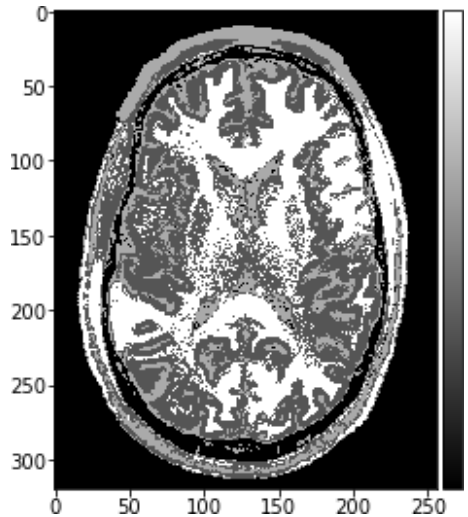
Image



2 clusters

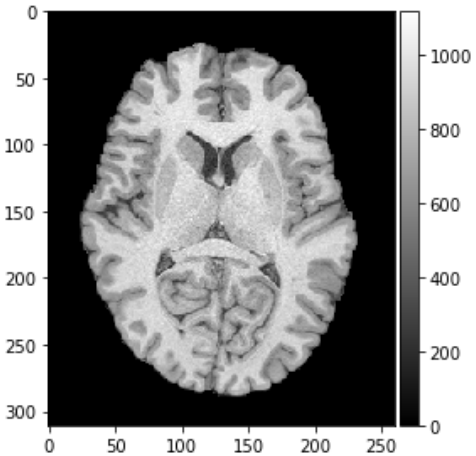


3 clusters

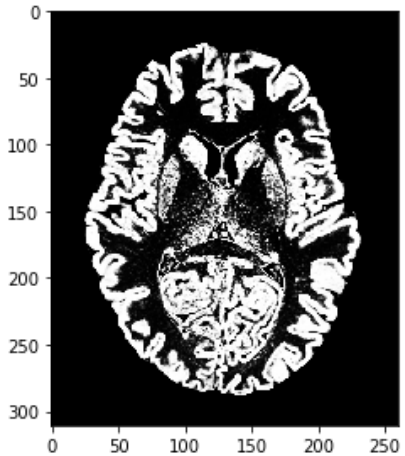


4 clusters

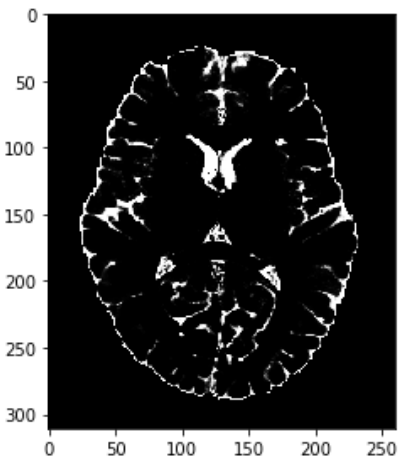
Where were we? - EM segmentation



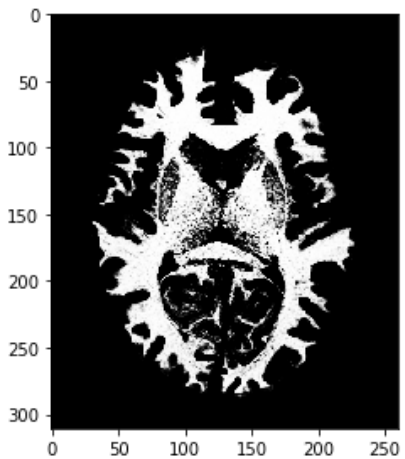
Image



$p(c = 1|I)$



$p(c = 2|I)$



$p(c = 3|I)$

Where were we? - Improvements

- Atlas-based segmentation
  - Registration
  - Anatomical priors
  - Contextual information
- Multi-atlas segmentation
  - Addressing the issues with registration
  - Multiple examples bring robustness
  - Aggregation through voting, STAPLE,...
- Patch-based methods for segmentation
  - Speeding up the multi-atlas approach by bypassing registration
  - Patch-matching and aggregation
  - Extensions with dictionary learning

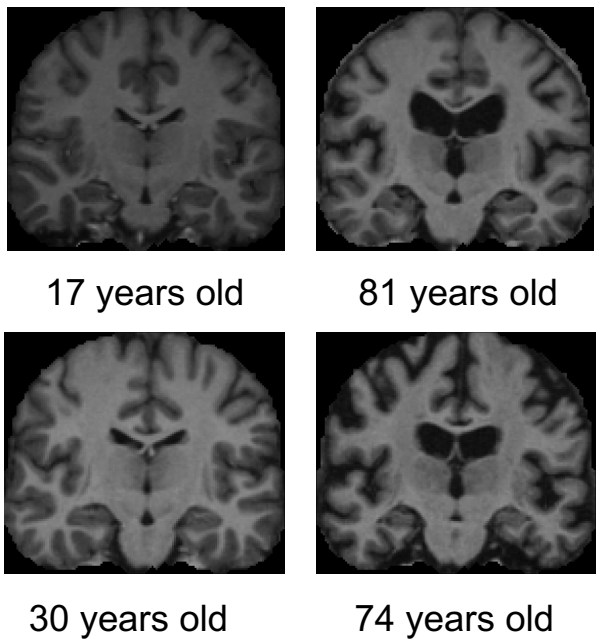
Outline

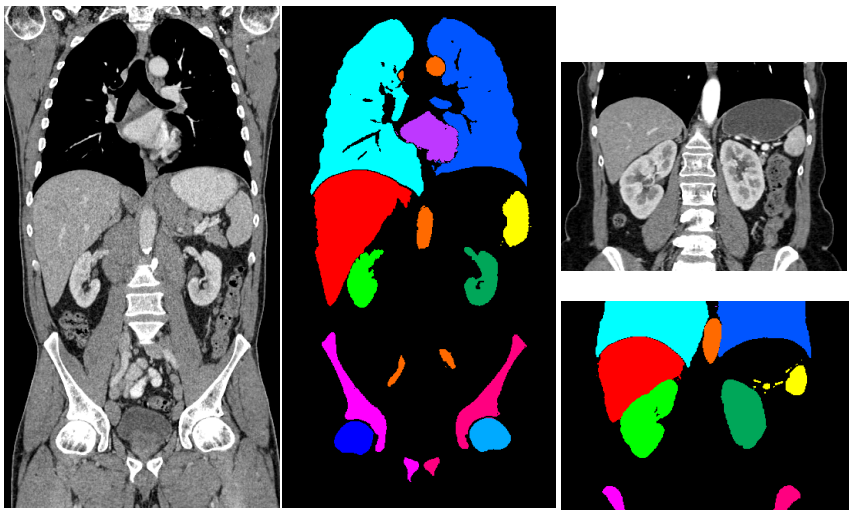
- Machine learning basics
- Segmentation with Random Decision Forests
- Post-processing

Machine Learning Basics

Patterns

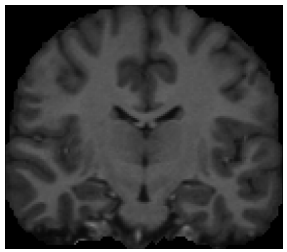
- Patterns may exist in the data (correlations, relationships, dependencies,...)
- Within images: e.g. liver is always below the lungs
- Between images and other information: e.g. brains age "similarly"
- Patterns arise due to nature or some causal relationships



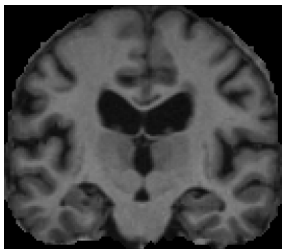


Elementary types of learning: supervised learning

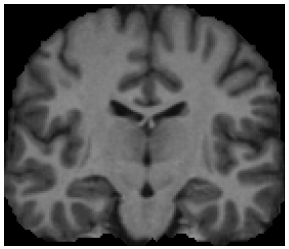
- Patterns between two types of data
- Goal: predict one from the other
- Learn the mapping from examples
- Examples have both types of data
- At prediction only one exists



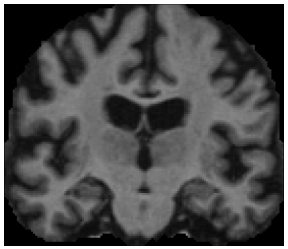
17 years old



81 years old



30 years old

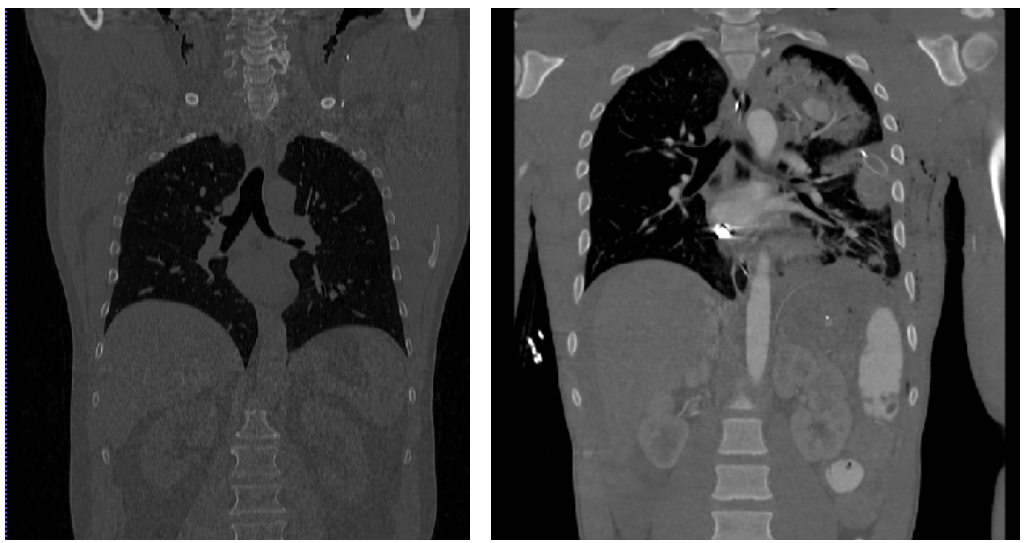


74 years old

30 years old

Elementary types of learning: unsupervised learning

- Patterns within the data
- Goal: describe variability
- Learning a high dimensional distribution
- Examples have only the features
- Unsupervised representation learning



Basic notation

- Data representation:

$$\{f_1, \ldots, f_N, y_1, \ldots, y_M\} = \{\mathbf{f}, \mathbf{y}\} = \{\text{features, labels}\}$$

Basic notation

- Data representation:

$$\{f_1, \ldots, f_N, y_1, \ldots, y_M\} = \{\mathbf{f}, \mathbf{y}\} = \{\text{features, labels}\}$$

- Continuous, discrete, categorical

Basic notation

- Data representation:

$$\{f_1, \ldots, f_N, y_1, \ldots, y_M\} = \{\mathbf{f}, \mathbf{y}\} = \{\text{features, labels}\}$$

- Continuous, discrete, categorical
- Supervised learning - mapping between features and labels

$$\mathbf{y} = l(\mathbf{f}; \theta), \ \theta : \text{model parameters}$$

Basic notation

- Data representation:

$$\{f_1, \ldots, f_N, y_1, \ldots, y_M\} = \{\mathbf{f}, \mathbf{y}\} = \{\text{features, labels}\}$$

- Continuous, discrete, categorical
- Supervised learning - mapping between features and labels

$$\mathbf{y} = l\left(\mathbf{f}; \theta\right), \ \theta : \text{model parameters}$$

- Unsupervised learning - learning distribution of features

$$p(\mathbf{f}; \theta), \ \theta : \text{model parameters}$$

Prediction

Given the set of parameters prediction is simply evaluating the functions

$$\hat{\mathbf{y}} = I(\mathbf{f}; \theta^*) \text{ or } p(\mathbf{f}; \theta^*)$$

with θ^* being the "best" parameter set.

Prediction

Given the set of parameters prediction is simply evaluating the functions

$$\hat{\mathbf{y}} = I(\mathbf{f}; \theta^*) \text{ or } p(\mathbf{f}; \theta^*)$$

with θ^* being the "best" parameter set.
For segmentation, $\hat{\mathbf{y}}(x)$ is the prediction at point x and $\mathbf{f}(x)$ are the feature representation of the point.

$$\hat{y}(x) = I(\mathbf{f}(x); \theta^*)$$

Prediction

Given the set of parameters prediction is simply evaluating the functions

$$\hat{\mathbf{y}} = l(\mathbf{f}; \theta^*) \text{ or } p(\mathbf{f}; \theta^*)$$

with θ^* being the "best" parameter set.
For segmentation, $\hat{\mathbf{y}}(x)$ is the prediction at point x and $\mathbf{f}(x)$ are the feature representation of the point.

$$\hat{y}(x) = l(\mathbf{f}(x); \theta^*)$$

What is the link with the probabilistic models we have seen before?

Prediction

Given the set of parameters prediction is simply evaluating the functions

$$\hat{\mathbf{y}} = I(\mathbf{f}; \theta^*) \text{ or } p(\mathbf{f}; \theta^*)$$

with θ^* being the "best" parameter set.
For segmentation, $\hat{\mathbf{y}}(x)$ is the prediction at point x and $\mathbf{f}(x)$ are the feature representation of the point.

$$\hat{y}(x) = I(\mathbf{f}(x); \theta^*)$$

What is the link with the probabilistic models we have seen before?

$$\hat{y}(x) = I(\mathbf{f}(x); \theta^*) \longleftrightarrow p(y(x)|\mathbf{f}(x); \theta^*))$$

Prediction

Given the set of parameters prediction is simply evaluating the functions

$$\hat{\mathbf{y}} = I(\mathbf{f}; \theta^*) \text{ or } p(\mathbf{f}; \theta^*)$$

with θ^* being the "best" parameter set.
For segmentation, $\hat{\mathbf{y}}(x)$ is the prediction at point x and $\mathbf{f}(x)$ are the feature representation of the point.

$$\hat{y}(x) = I(\mathbf{f}(x); \theta^*)$$

What is the link with the probabilistic models we have seen before?

$$\hat{y}(x) = I(\mathbf{f}(x); \theta^*) \longleftrightarrow p(y(x)|\mathbf{f}(x); \theta^*))$$

$$\underbrace{p\left(y(x)|\mathbf{f}(x); \theta^*\right))}_{\text{Discriminative model}}, \quad \underbrace{p(\mathbf{f}(x)|y(x); \phi_{\text{likelihood}})p(\mathbf{y}; \phi_{\text{prior}})}_{\text{Generative model}}$$

Learning - Training

- Determining the parameters based on examples - **training**

$$\{\mathbf{f}_s, \mathbf{y}_s\}_{s=1,\ldots,S}$$

Learning - Training

- Determining the parameters based on examples - **training**

$$\{\mathbf{f}_s, \mathbf{y}_s\}_{s=1,\ldots,S}$$

- Best parameters are those that minimize a distance

$$\theta^* = \arg_\theta \min \sum_{s=1}^{S} d\left(I(\mathbf{f}_s; \theta), \mathbf{y}_s\right)$$

Learning - Training

- Determining the parameters based on examples - **training**

$$\{\mathbf{f}_s, \mathbf{y}_s\}_{s=1,\ldots,S}$$

- Best parameters are those that minimize a distance

$$\theta^* = \arg_\theta \min \sum_{s=1}^{S} d\left(I(\mathbf{f}_s; \theta), \mathbf{y}_s\right)$$

- $d(\cdot, \cdot)$ is an appropriate distance function

Learning - Training

- Determining the parameters based on examples - **training**

$$\{\mathbf{f}_s, \mathbf{y}_s\}_{s=1,\ldots,S}$$

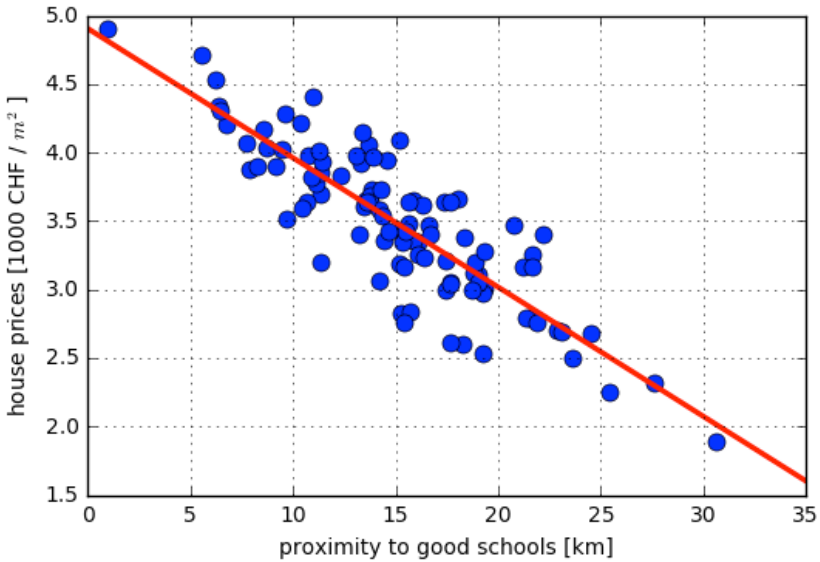
- Best parameters are those that minimize a distance

$$\theta^* = \arg_\theta \min \sum_{s=1}^{S} d\left(I(\mathbf{f}_s; \theta), \mathbf{y}_s\right)$$

- $d(\cdot, \cdot)$ is an appropriate distance function
- Unsupervised learning - maximizing the likelihood function

$$\theta^* = \arg_\theta \max \prod_{s=1}^{S} p(\mathbf{f}_s; \theta)$$

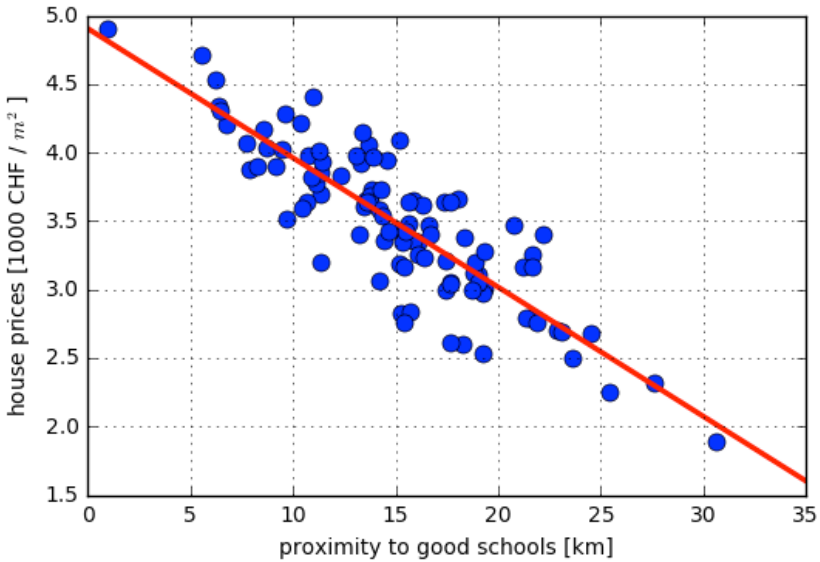
Regression example



Linear regression model

Red line is the $\hat{y} = l(f)$ line.

Regression example



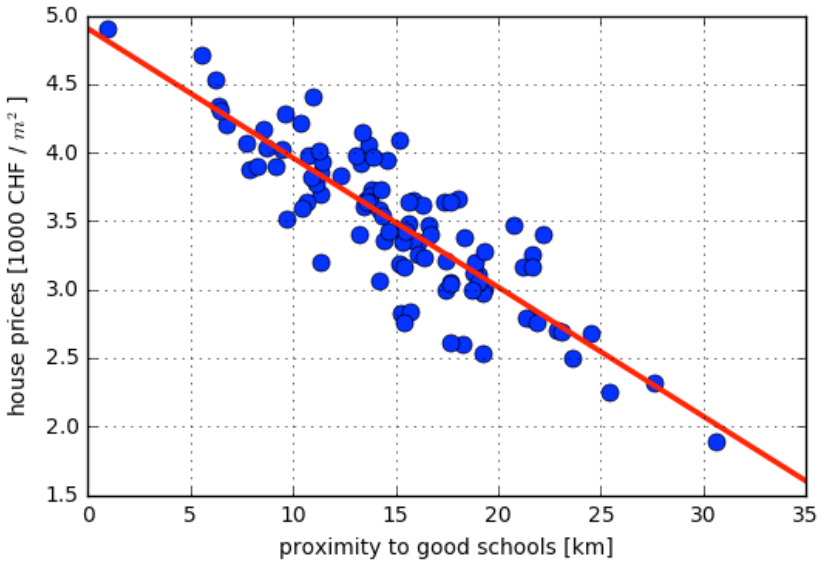
Red line is the $\hat{y} = l(f)$ line.

Linear regression model

f: proximity, y: house price

$\{f_s, y_s\}_{s=1,\ldots,S}$

Regression example



Red line is the $\hat{y} = l(f)$ line.

Linear regression model

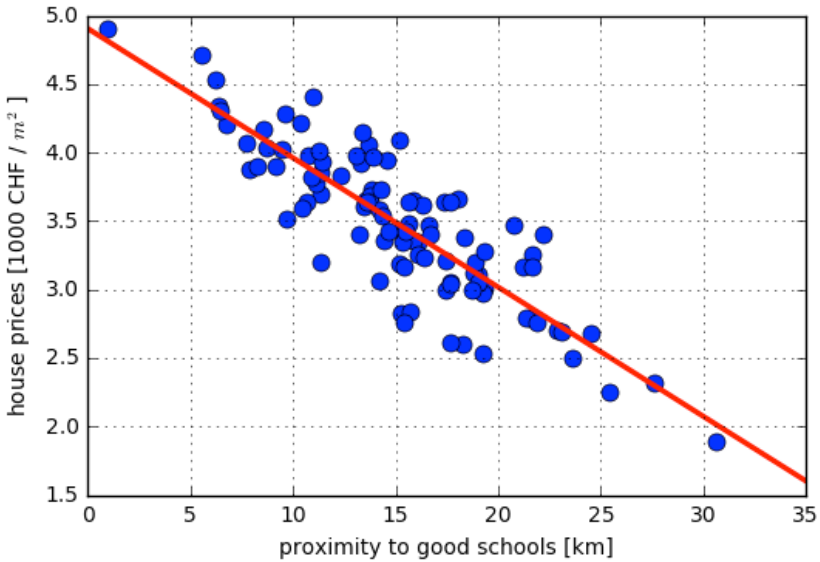
f: proximity, y: house price

$\{f_s, y_s\}_{s=1,\ldots,S}$

$y = l(f) = af + b$

$\theta = \{a, b\}$

Regression example



Red line is the $\hat{y} = l(f)$ line.

Linear regression model

f: proximity, y: house price

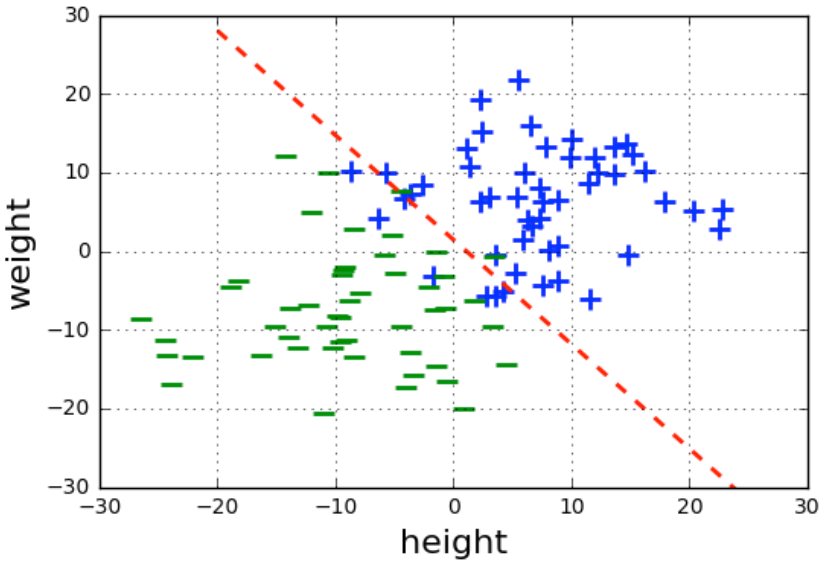
$\{f_s, y_s\}_{s=1,\ldots,S}$

$y = l(f) = af + b$

$\theta = \{a, b\}$

$d(l(f;\theta), y) = \|l(f;\theta) - y\|_2^2$

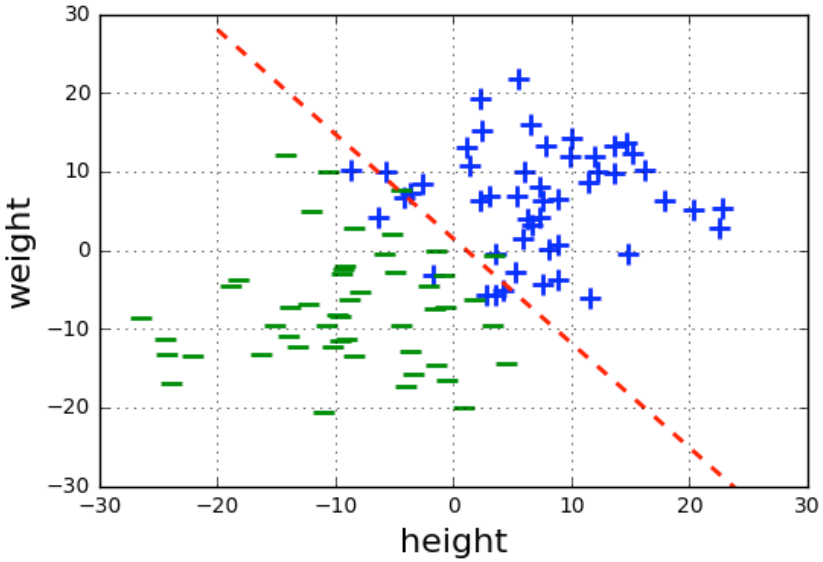
Classification example



Logistic regression model

The dashed line is at $l(\mathbf{f}) = 0.5$.

Classification example



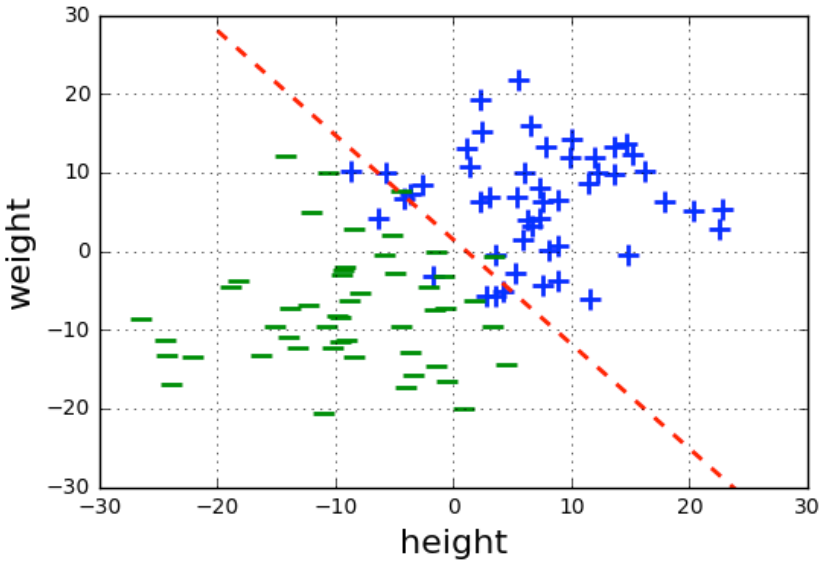
The dashed line is at $l(\mathbf{f}) = 0.5$.

Logistic regression model

f: [weight, height], y: gender

$\{\mathbf{f}_s, y_s\}_{s=1,\ldots,S}$

Classification example



The dashed line is at $l(\mathbf{f}) = 0.5$.

Logistic regression model

$\mathbf{f}$: [weight, height], y: gender

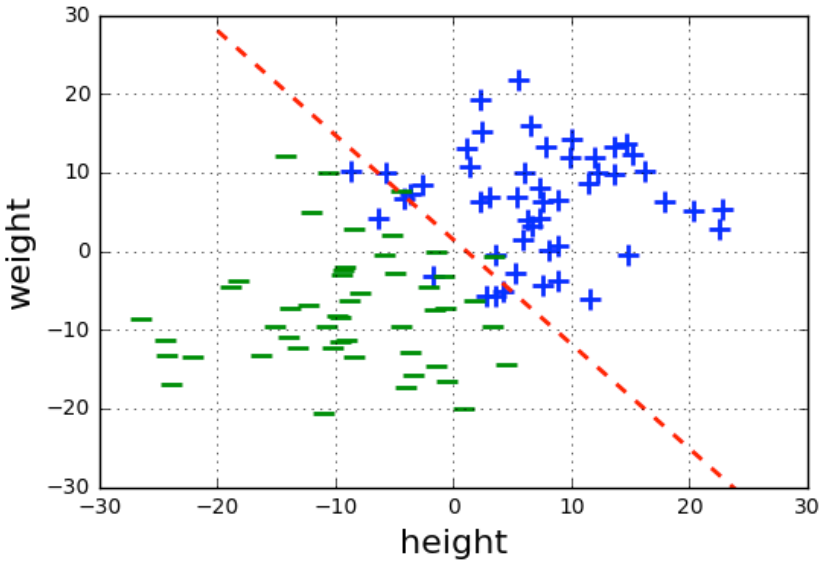
$\{\mathbf{f}_s, y_s\}_{s=1,\ldots,S}$

$l(\mathbf{f}) = \sigma(\mathbf{f}^T\mathbf{a}+b) = \sigma(a_1 f_1 + a_2 f_2 + b)$

$\sigma(z) = \frac{1}{1+e^{-z}}$

$\theta = \{a_1, a_2, b\}$

Classification example



The dashed line is at $l(\mathbf{f}) = 0.5$.

Logistic regression model

$\mathbf{f}$: [weight, height], y: gender

$\{\mathbf{f}_s, y_s\}_{s=1,\ldots,S}$

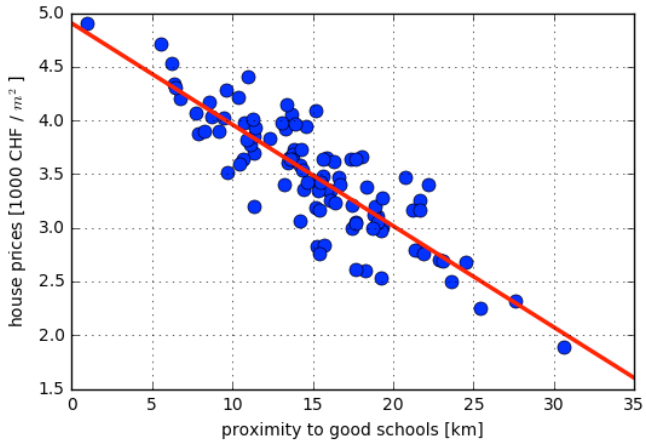
$l(\mathbf{f}) = \sigma(\mathbf{f}^T\mathbf{a}+b) = \sigma(a_1 f_1 + a_2 f_2 + b)$

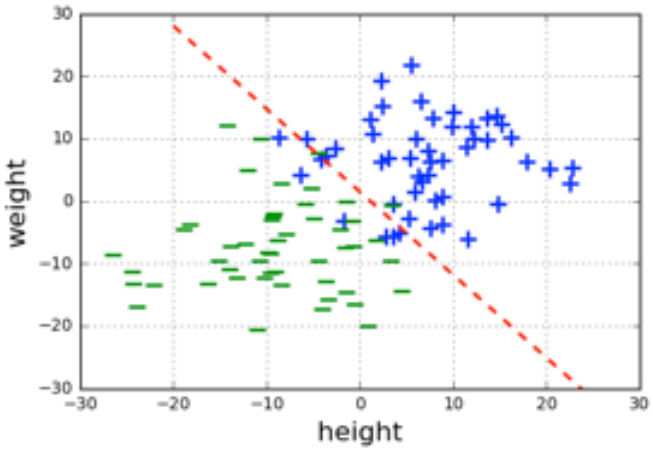
$\sigma(z) = \frac{1}{1+e^{-z}}$

$\theta = \{a_1, a_2, b\}$

$$\begin{aligned} d(l(\mathbf{f};\theta), y) &= -y \log(l(\mathbf{f};\theta)) \\ &- (1-y)\log(1-l(\mathbf{f};\theta)) \end{aligned}$$

Predictors

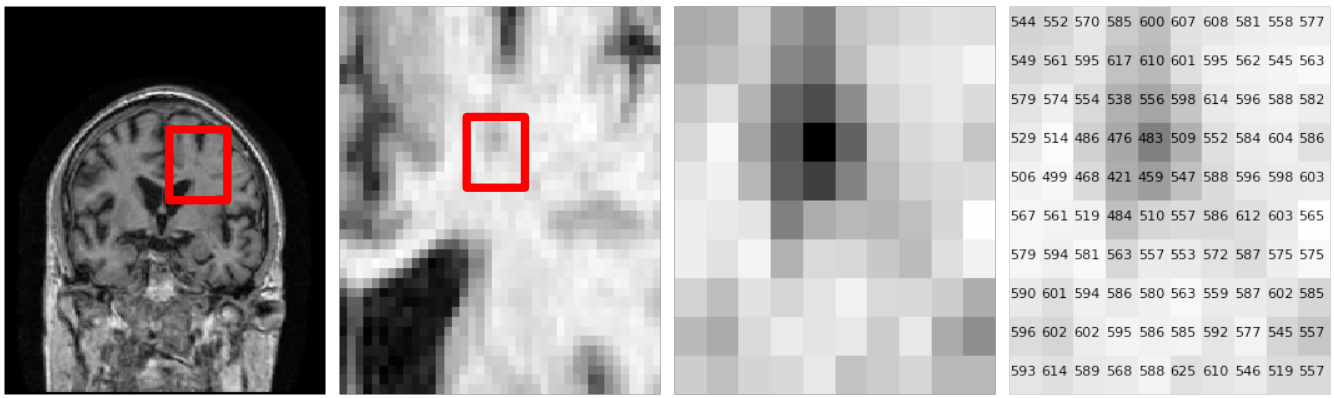




Proximity to good schools
Number of rooms
Surface area
View
Garden
...

Weight
Height
Age
Shoe size
Hair length
...

Predictors in images



- Multiple predictors
- Complex spatial correlations between predictors
- Two challenges to address:
  - $\mathbf{f} = ?$: Should we use raw intensities as features or extract derived features from the intensities?
  - $I(\mathbf{x}; \theta) = ?$: What model can we use with such high-dimensional predictors?

Question / discussion

What were the predictors in

Question / discussion

What were the predictors in

- EM-based segmentation?

Question / discussion

What were the predictors in

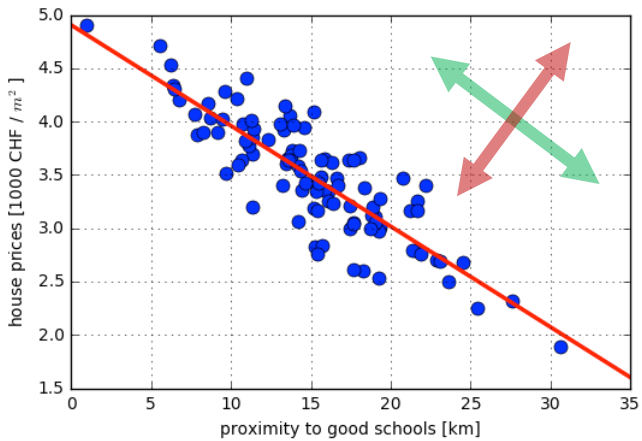
- EM-based segmentation?
- Patch-based segmentation?

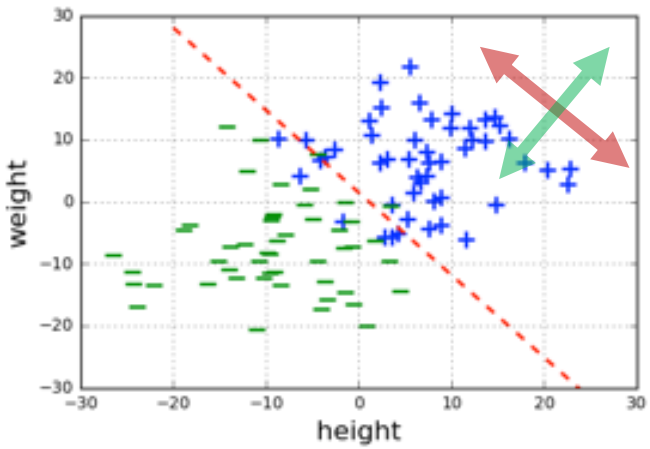
Question / discussion

What were the predictors in

- EM-based segmentation?
- Patch-based segmentation?
- Atlas-based segmentation?

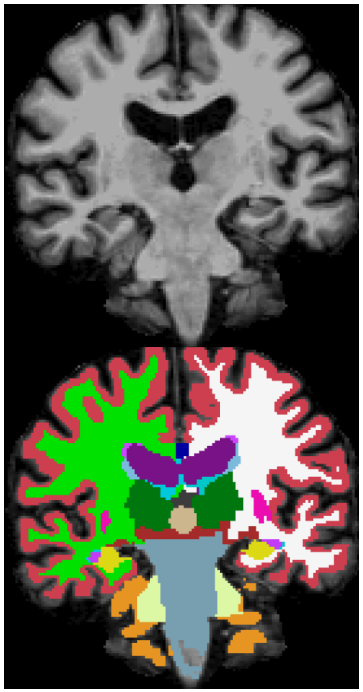
Variability





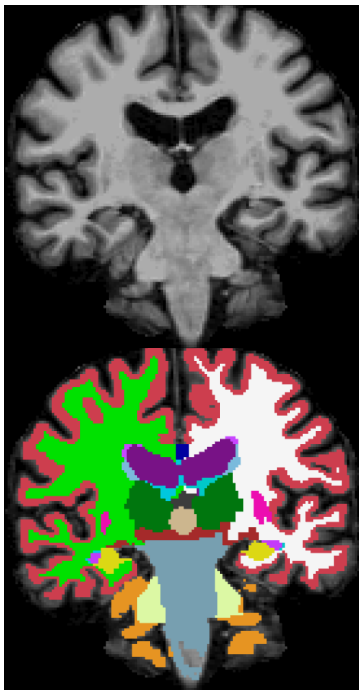
- Two types of variation in predictors
- Inter-group variation – useful for prediction (green arrows)
- Intra-group variation – not useful for prediction (red arrows)
- Models need to learn to use one and gain invariance to the other

Historical view - 1. Statistical univariate analysis



- f: volume of one anatomical structure
  y: age of the person

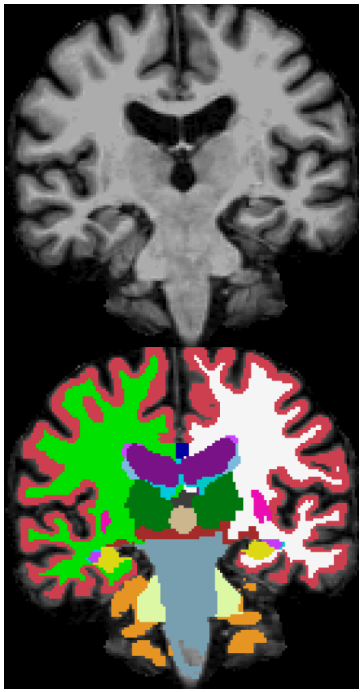
Historical view - 1. Statistical univariate analysis



- f: volume of one anatomical structure
  y: age of the person

$$l(f|\theta) = af + b$$

Historical view - 1. Statistical univariate analysis

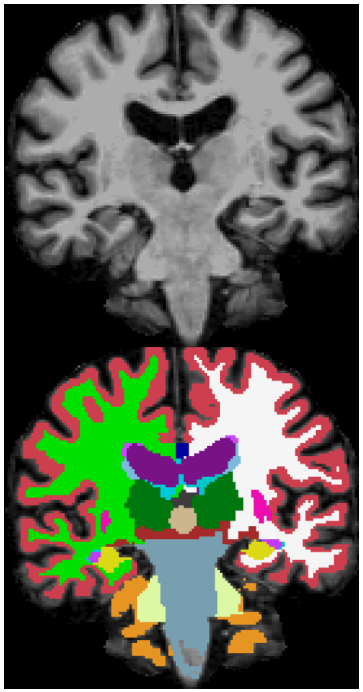


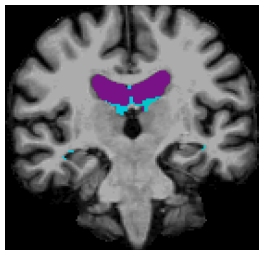
- f: volume of one anatomical structure
  y: age of the person

$$l(f|\theta) = af + b$$

- Hand-crafted measurements
- Univariate regression analysis / correlation analysis
- Small number of parameters
- Discovering statistical relationships

Historical view - 1. Statistical univariate analysis





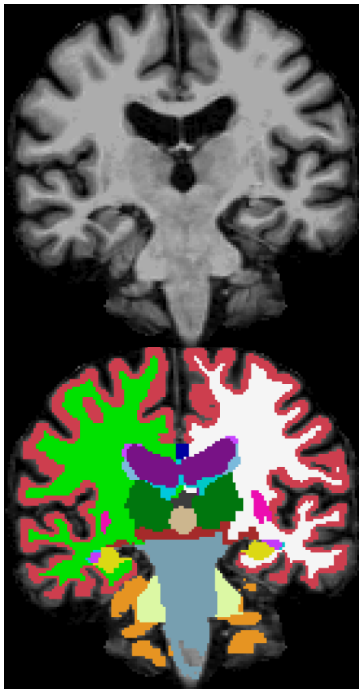
Pearson's Correlation Coef.
Left Lateral Ventricle – 0.72
Right Lateral Ventricle – 0.72
Left Choroid Plexus – 0.71
Right Choroid Plexus – 0.73

- f: volume of one anatomical structure
  y: age of the person

$$l(f|\theta) = af + b$$

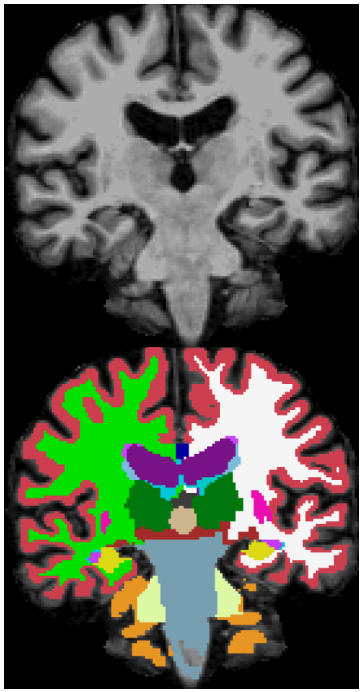
- Hand-crafted measurements
- Univariate regression analysis / correlation analysis
- Small number of parameters
- Discovering statistical relationships

Historical view - 2. Multivariate predictive methods



- $\mathbf{f}$: volumes of all anatomical structures
  y: age of the person

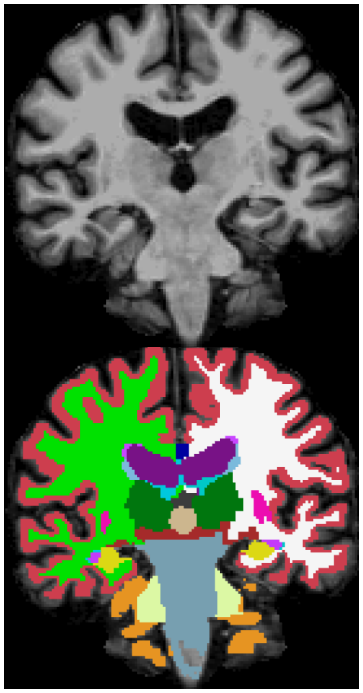
Historical view - 2. Multivariate predictive methods



- **f**: volumes of all anatomical structures
  y: age of the person

$$I(f|\theta) = \left\{ \begin{array}{c} \text{Support vector machines} \\ \text{K-nearest neighbors} \\ \text{decision trees, LDA, ...} \end{array} \right.$$

Historical view - 2. Multivariate predictive methods

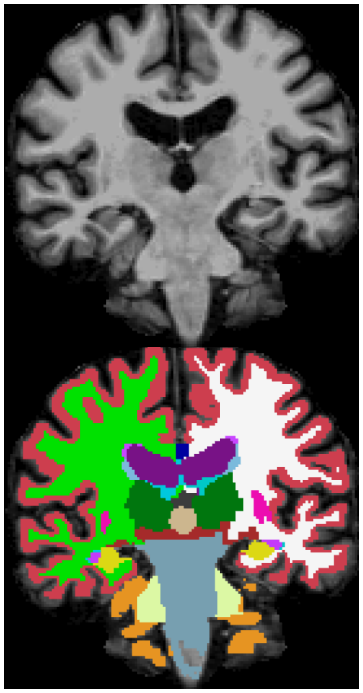


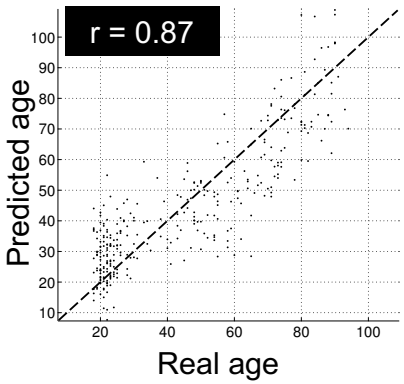
- **f**: volumes of all anatomical structures
  y: age of the person

$$I(f|\theta) = \left\{ \begin{array}{c} \text{Support vector machines} \\ \text{K-nearest neighbors} \\ \text{decision trees, LDA, ...} \end{array} \right.$$

- Hand-crafted measurements
- Multivariate analysis
- Larger number of parameters
- Predicting the label well

Historical view - 2. Multivariate predictive methods



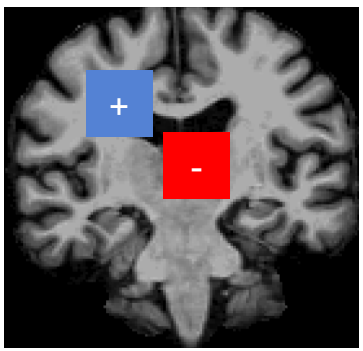


- **f**: volumes of all anatomical structures
  y: age of the person

$$I(f|\theta) = \left\{ \begin{array}{c} \text{Support vector machines} \\ \text{K-nearest neighbors} \\ \text{decision trees, LDA, ...} \end{array} \right.$$

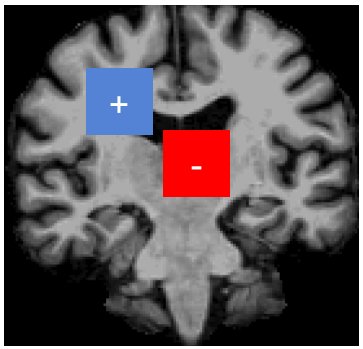
- Hand-crafted measurements
- Multivariate analysis
- Larger number of parameters
- Predicting the label well

Historical view - 3. Towards primitive features



- $\mathbf{f}$: intensity differences between regions
  y: age of the person

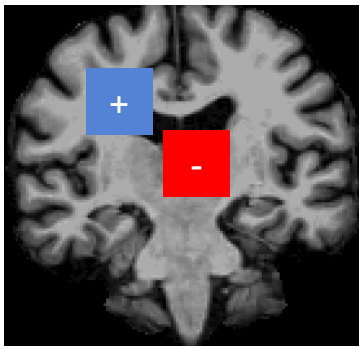
Historical view - 3. Towards primitive features



- **f**: intensity differences between regions
  y: age of the person

$$I(f|\theta) = \begin{cases} \text{Random Forest} \\ \text{Boosting Trees,...} \end{cases}$$

Historical view - 3. Towards primitive features

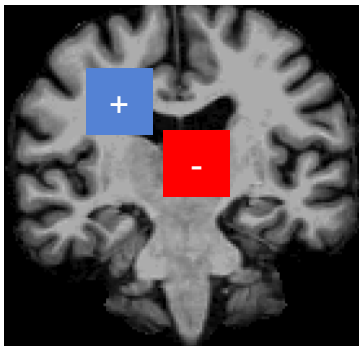


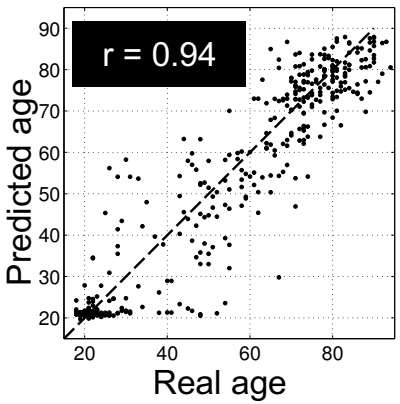
- **f**: intensity differences between regions
  y: age of the person

$$I(f|\theta) = \begin{cases} \text{Random Forest} \\ \text{Boosting Trees,...} \end{cases}$$

- Less hand-crafted and lots of measurements
- Automatic selection of relevant features
- Even more model parameters
- Predicting the label well
- Easily extendable to other problems, including segmentation

Historical view - 3. Towards primitive features



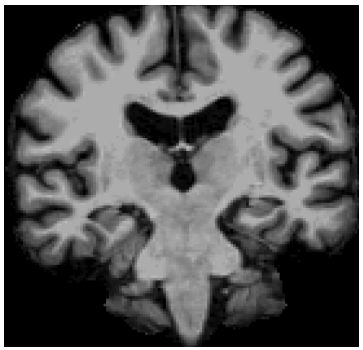


- **f**: intensity differences between regions
  y: age of the person

$$I(f|\theta) = \begin{cases} \text{Random Forest} \\ \text{Boosting Trees,...} \end{cases}$$

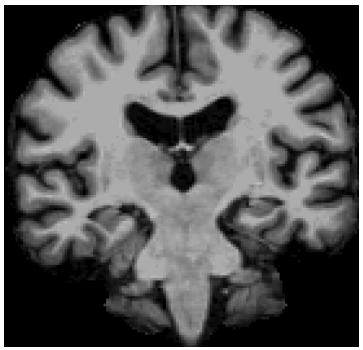
- Less hand-crafted and lots of measurements
- Automatic selection of relevant features
- Even more model parameters
- Predicting the label well
- Easily extendable to other problems, including segmentation

Historical view - 4. Raw intensities



- $\mathbf{f}$: raw intensities
  y: age of the person

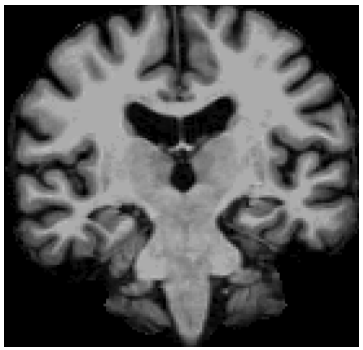
Historical view - 4. Raw intensities



- $\mathbf{f}$: raw intensities
  y: age of the person

$$l(f|\theta) = \begin{cases} \text{Neural networks} \\ \text{Convolutional neural networks} \\ \text{Dictionary learning, ...} \end{cases}$$

Historical view - 4. Raw intensities

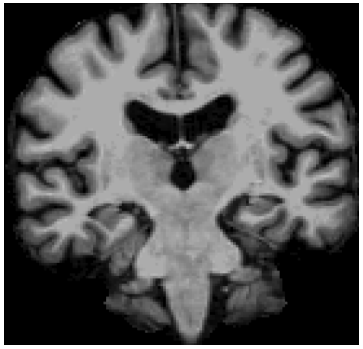


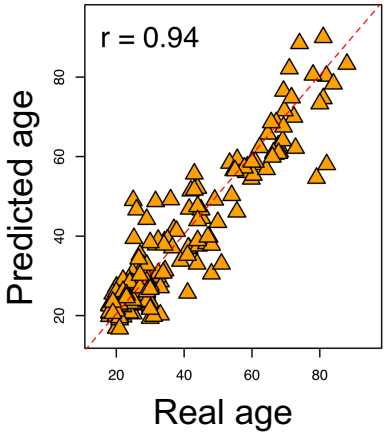
- $\mathbf{f}$: raw intensities
  y: age of the person

$$l(f|\theta) = \left\{ \begin{array}{c} \text{Neural networks} \\ \text{Convolutional neural networks} \\ \text{Dictionary learning, ...} \end{array} \right.$$

- No pre-defined measurements - completely data-driven
- Automatic extraction of task-specific features
- Many, many more model parameters
- Easily extendable to other problems, including segmentation

Historical view - 4. Raw intensities





- **f**: raw intensities
  y: age of the person

$$l(f|\theta) = \begin{cases} \text{Neural networks} \\ \text{Convolutional neural networks} \\ \text{Dictionary learning, ...} \end{cases}$$

- No pre-defined measurements - completely data-driven
- Automatic extraction of task-specific features
- Many, many more model parameters
- Easily extendable to other problems, including segmentation

Segmentation with Random Decision Forests

Revisiting notation

- Mapping from intensities to labels

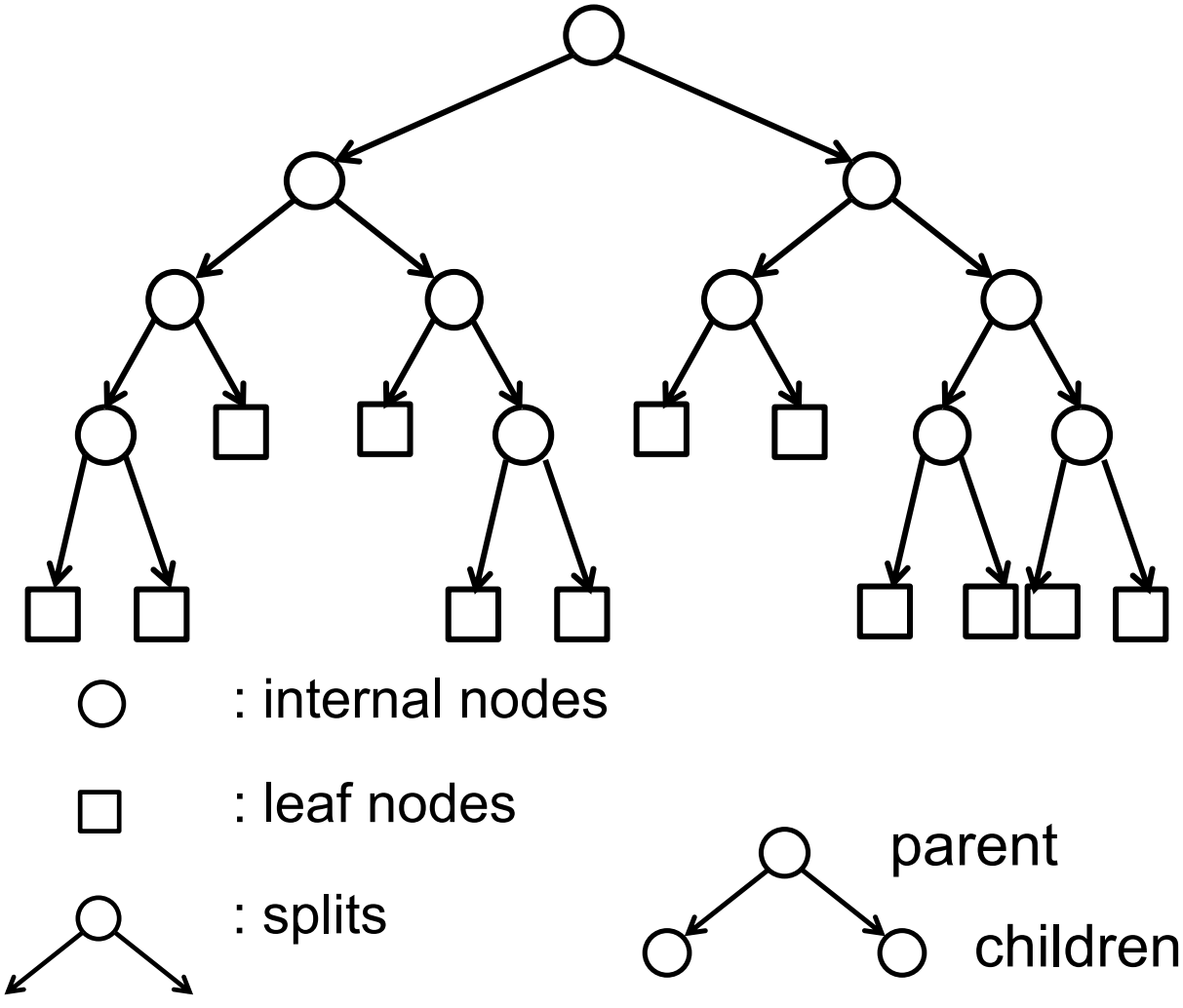
$$y(x) = I(\mathbf{f}(x); \theta)$$

- Let us assume features are given for now.
- Parameterized with θ
- Determine the parameters that yield the best prediction in a *training set*, i.e. set of example images along with their *ground-truth* labels

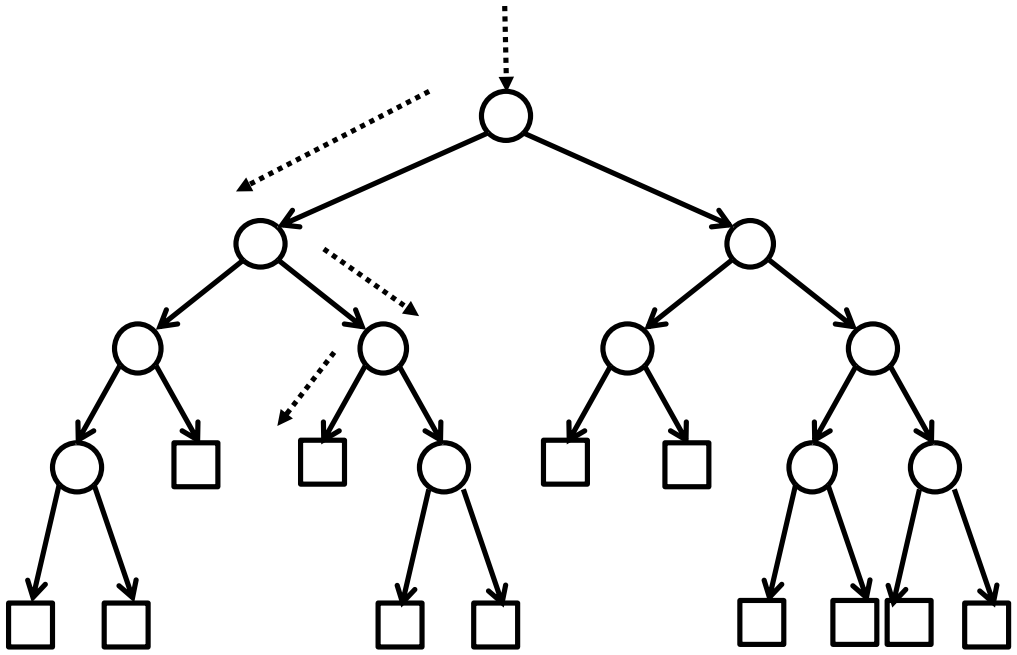
$$\theta^* = \arg_\theta \min \sum_{s=1}^{S} \sum_{x} d\left(I(\mathbf{f}^{(s)}(x); \theta), y_s(x)\right)$$

- **Random Decision Forests**: Ensemble of decision trees
- [Crimisi, Shotton, Konukoglu; Now Publications 2012]

Binary decision trees



Binary decision trees

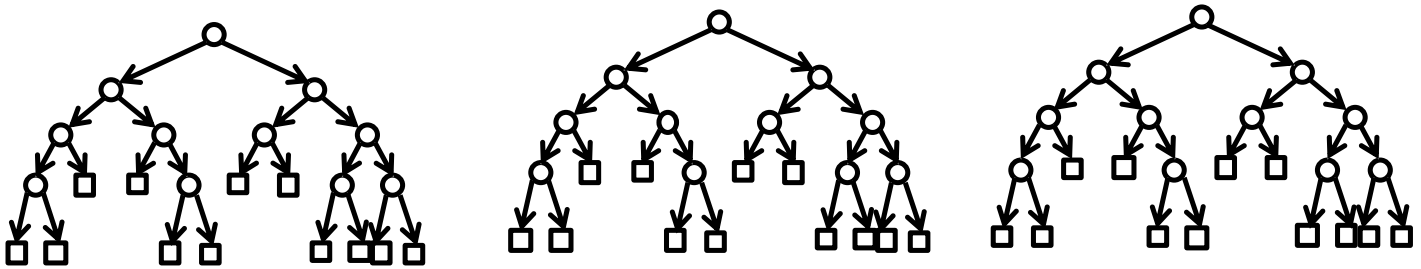


- At each internal node n there is a binary question on the features f
- At each leaf node l there is a prediction and it changes across leaf nodes

$$\tau_n(\mathbf{f}) = \begin{cases} 0 \rightarrow & \text{go right} \\ 1 \rightarrow & \text{go left} \end{cases}$$

$$\tau_l(\mathbf{f}) = \hat{y}(\mathbf{f})$$

Forest is an ensemble of trees



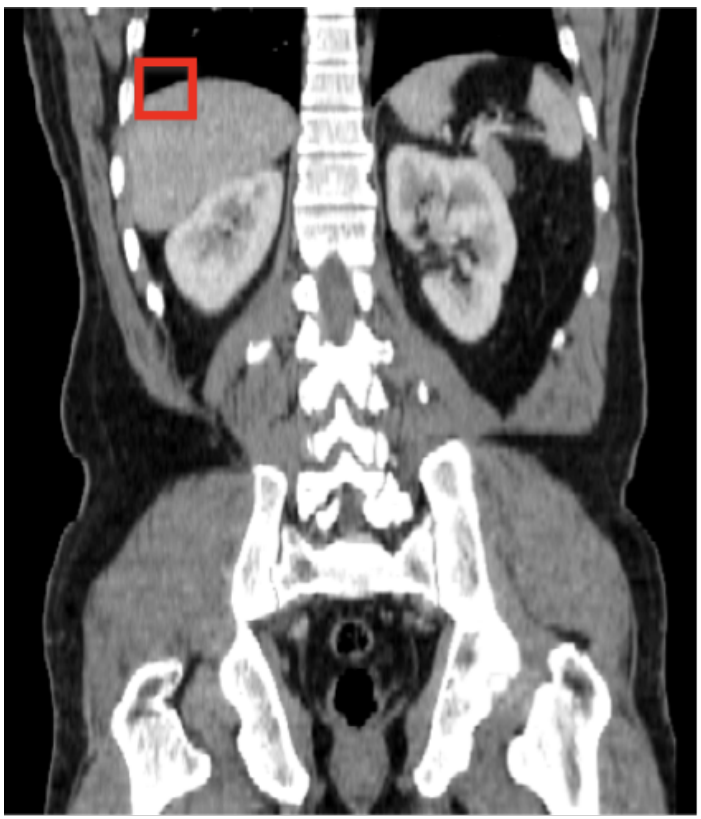
- Each tree is different than others
- Focus on a different set of features

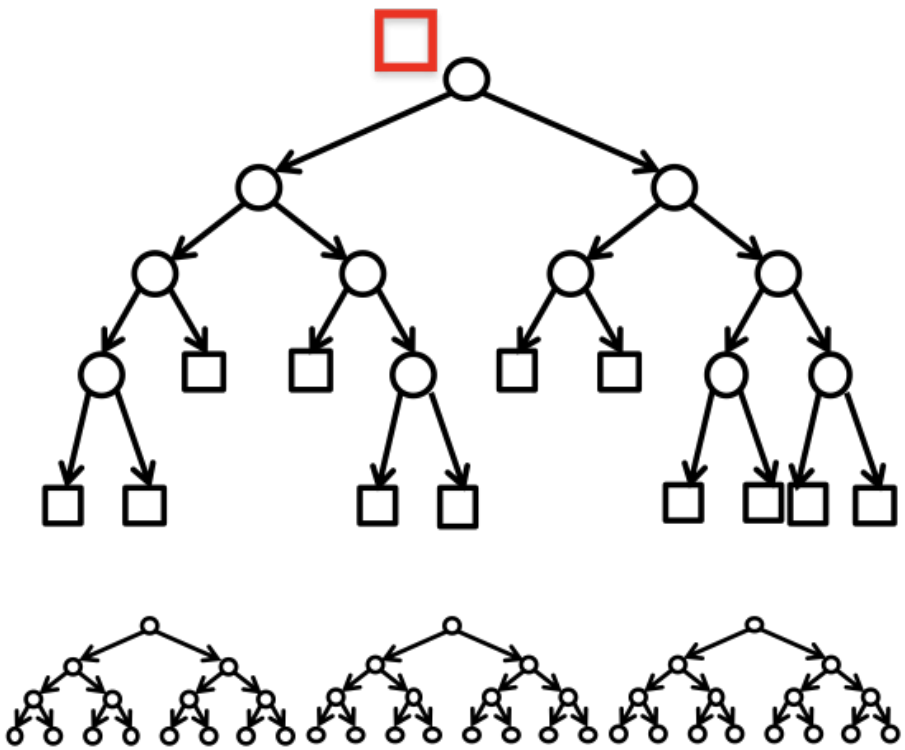
$$\begin{aligned} \mathbf{f}(x) &= \{f_1(x), f_2(x), \ldots, f_d(x)\} \\ \mathbf{f}_i(x) &= \{f_{i_1}(x), f_{i_2}(x), \ldots, f_{i_D}(x)\} \end{aligned}$$

with $d >> D$

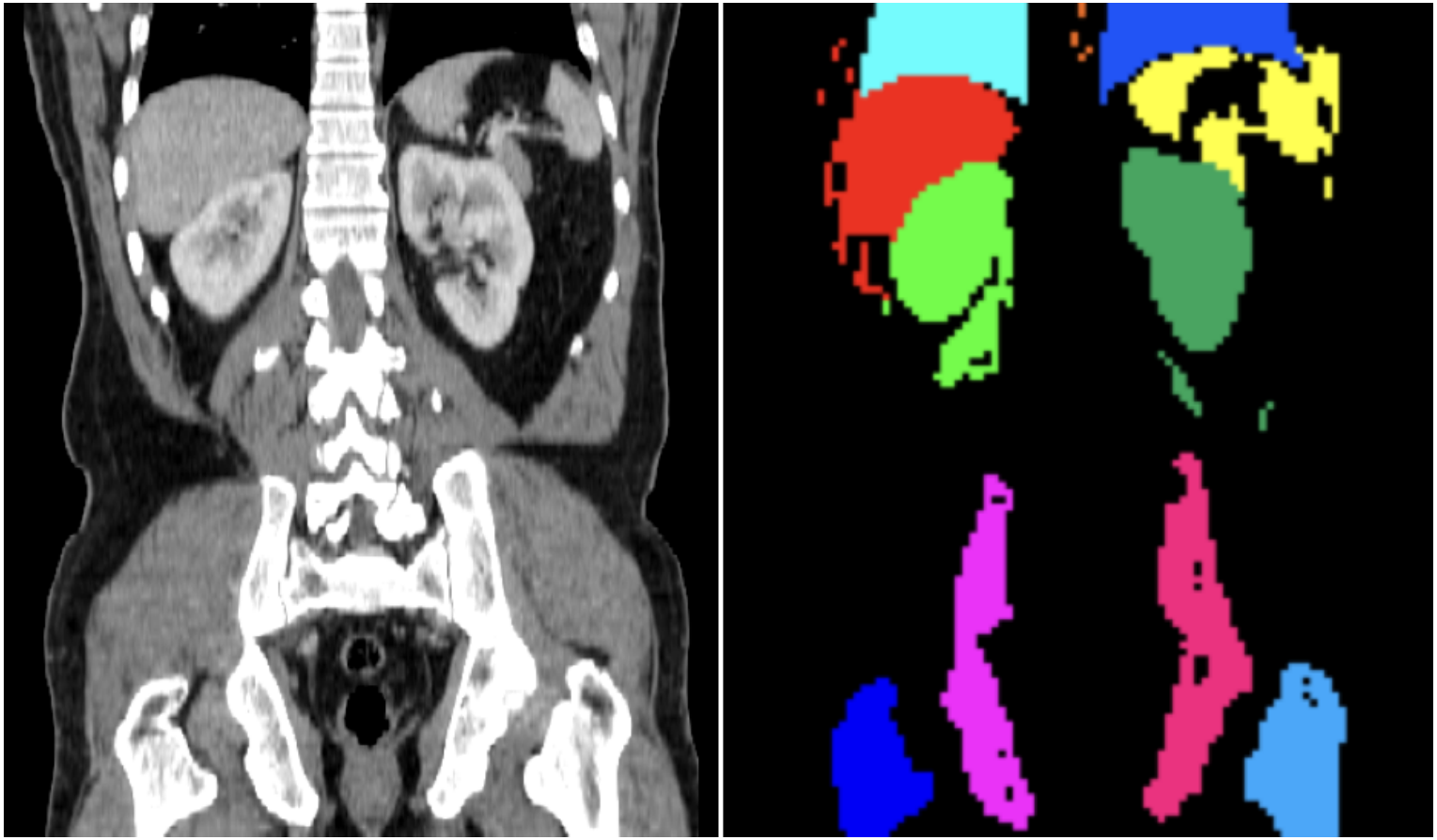
- An approach to deal with high-dimensionality
- Taking into account many many features

How does it segment images?





How does it segment images?



Parameterization

Two levels of parameterization

- What are the test?

$$\tau_n(\mathbf{f}) = \begin{cases} 0 \rightarrow & \text{go right} \\ 1 \rightarrow & \text{go left} \end{cases}$$

- How are the predictions done?

$$\tau_l(\mathbf{f}) = \hat{y}(\mathbf{f})$$

- Both are learned during training
- Various alternatives for both

A commonly used example

A simple example that is commonly used

- Binary stump

$$\tau_n(\mathbf{f}) = \begin{cases} 0, & f_k < t_n \\ 1, & f_k \geq t_n \end{cases}$$

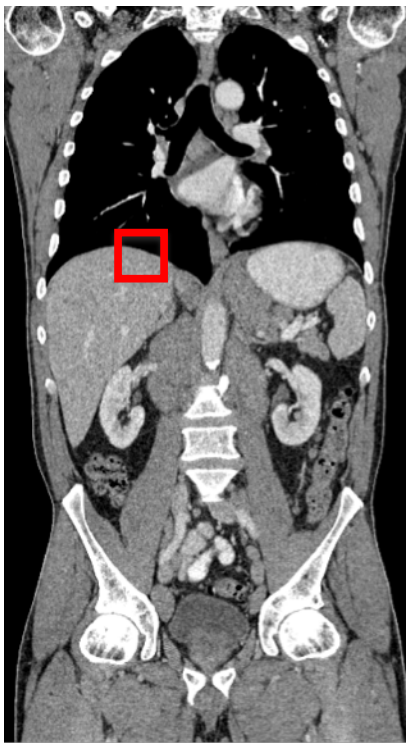
  where k is the feature index and t_n is the threshold.

- Expected label at leaf node

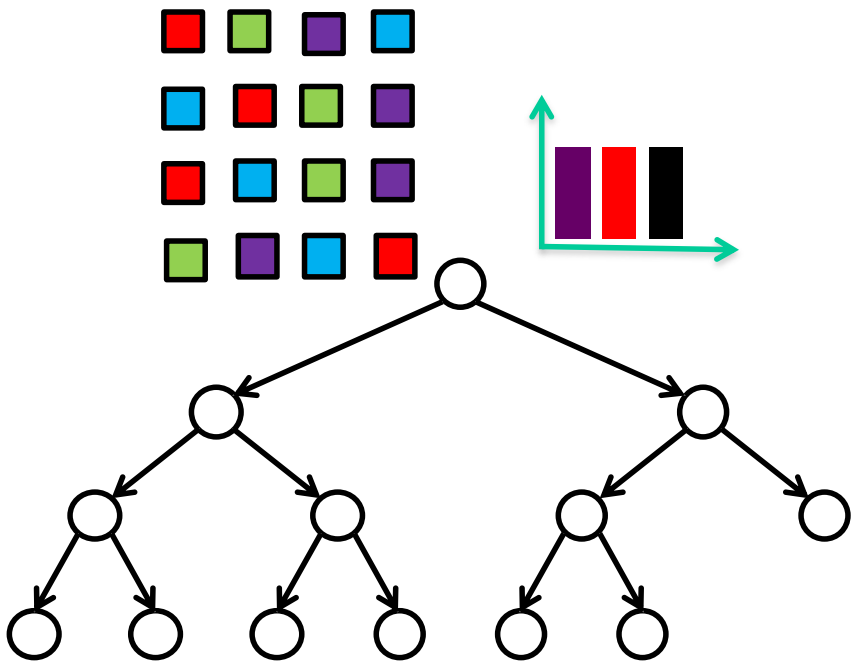
$$\tau_l(\mathbf{f}) = \mathbb{E}_{\mathcal{S}_{training}}\left[y|\text{leaf } l\right]$$

- Parameters k and t_n are learned for each node
- For a given set of parameters, each training images will land on a different leaf node
- At each leaf the label probability and the expected value is different

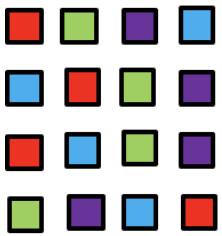
Start of the training

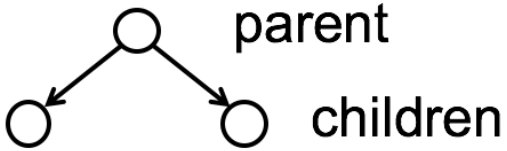


$$\blacksquare \rightarrow f^{(1)} = \{f_1^{(1)}, \ldots, f_d^{(1)}\}$$
$$\blacksquare \rightarrow f^{(2)} = \{f_1^{(2)}, \ldots, f_d^{(2)}\}$$
$$\blacksquare \rightarrow f^{(n)} = \{f_1^{(n)}, \ldots, f_d^{(n)}\}$$

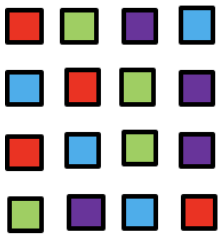


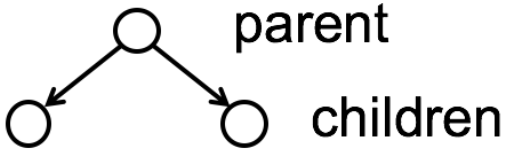
Node split



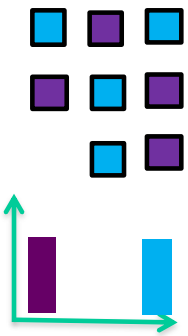


Node split

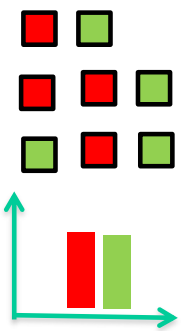




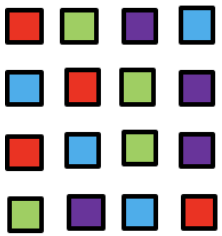
Left child

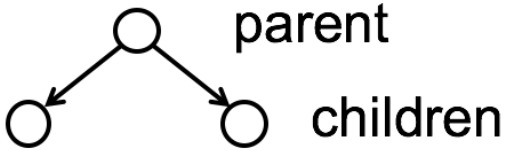


Right child



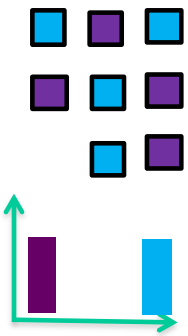
Node split



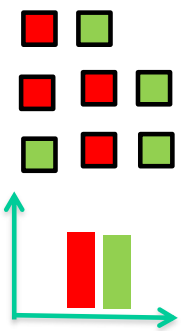


$$D_n = \{(f^{(s)}, L_s) \in \text{node } n\}$$

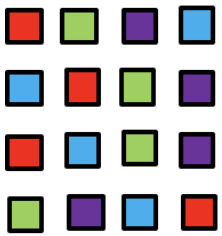
Left child

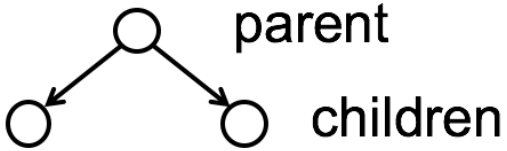


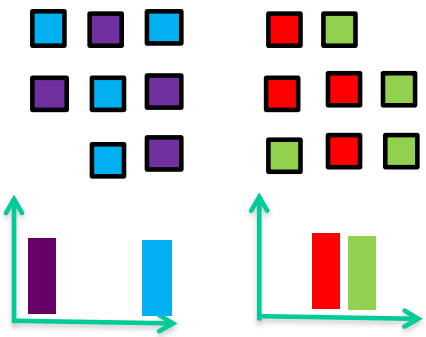
Right child



Node split



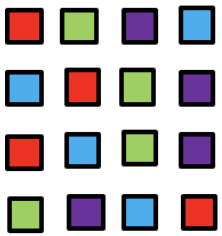


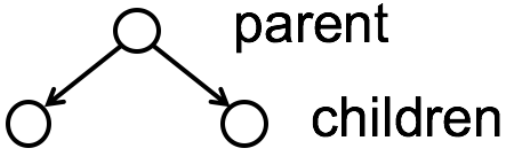


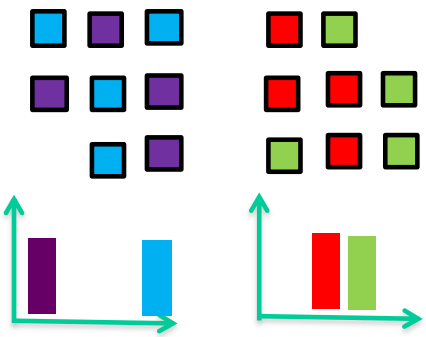
$$D_n = \{(f^{(s)}, L_s) \in \text{node } n\}$$

$$G(k, t_n) = H(\text{parent}) - \frac{\#D_{rc}}{\#D_n} H(\text{rc}) - \frac{\#D_{lc}}{\#D_n} H(\text{lc})$$

Node split

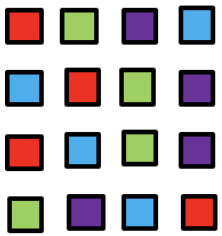


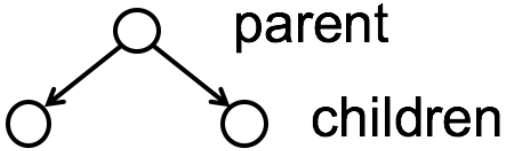


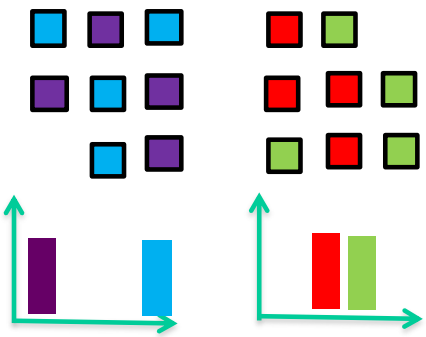


$$
\begin{aligned}
D_n &= \{(f^{(s)}, L_s) \in \text{node } n\} \\
G(k, t_n) &= H(\text{parent}) - \frac{\#D_{rc}}{\#D_n} H(\text{rc}) - \frac{\#D_{lc}}{\#D_n} H(\text{lc}) \\
H(\text{node } n) &= \sum_{j=1}^{J} p(w_j) \log p(w_j)
\end{aligned}
$$

Node split

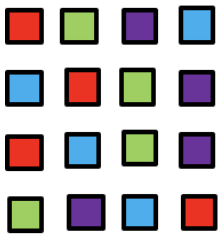


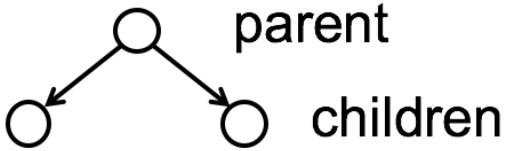


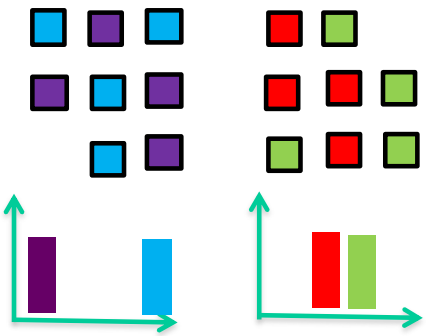


$$
\begin{aligned}
D_n &= \{(f^{(s)}, L_s) \in \text{node } n\} \\
G(k, t_n) &= H(\text{parent}) - \frac{\#D_{rc}}{\#D_n} H(\text{rc}) - \frac{\#D_{lc}}{\#D_n} H(\text{lc}) \\
H(\text{node } n) &= \sum_{j=1}^{J} p(w_j) \log p(w_j) \\
p(w_j) &= \sum_{i=1}^{\#D_n} \delta(y_i = w_j)/\#D_n
\end{aligned}
$$

Node split





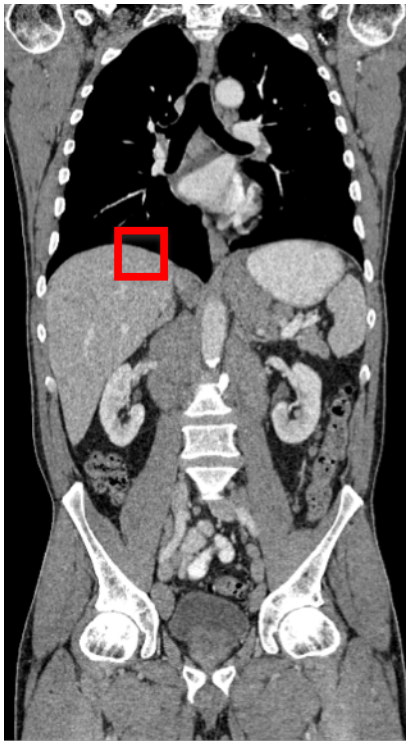


$$
\begin{aligned}
D_n &= \{(f^{(s)}, L_s) \in \text{node } n\} \\
G(k, t_n) &= H(\text{parent}) - \frac{\#D_{rc}}{\#D_n} H(\text{rc}) - \frac{\#D_{lc}}{\#D_n} H(\text{lc}) \\
H(\text{node } n) &= \sum_{j=1}^{J} p(w_j) \log p(w_j) \\
p(w_j) &= \sum_{i=1}^{\#D_n} \delta(y_i = w_j)/\#D_n \\
k^*, t_n^* &= \arg\max G(k, t_n)
\end{aligned}
$$

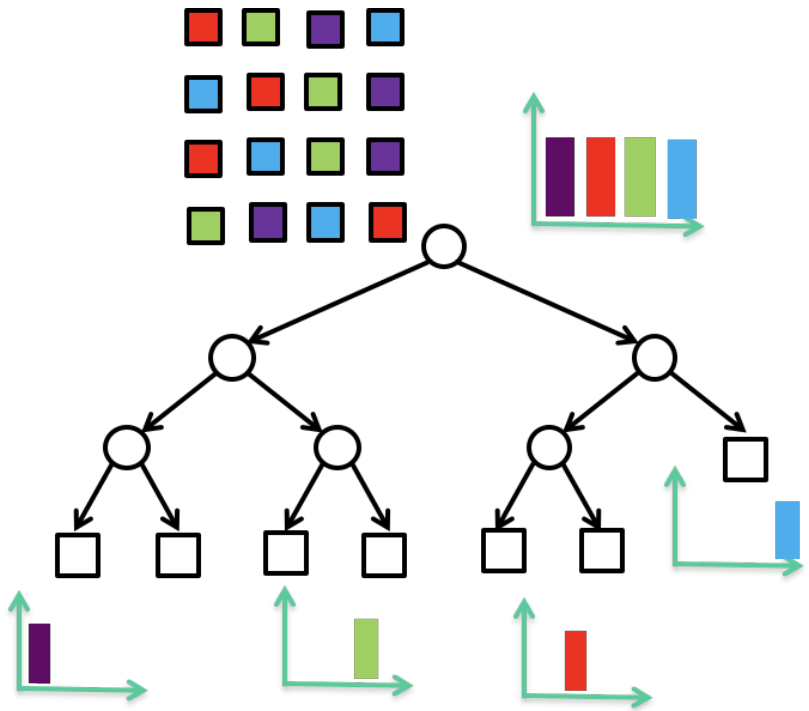
While training

- Progressively grow the tree
- At each node perform node split
- Stop when there are few samples in a node and call it leaf
- Apply the same for each tree independently
- *Only use a random subset of features for each tree*

End of the training

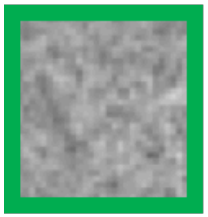


$$\blacksquare \rightarrow f^{(1)} = \{f_1^{(1)}, \ldots, f_d^{(1)}\}$$
$$\blacksquare \rightarrow f^{(2)} = \{f_1^{(2)}, \ldots, f_d^{(2)}\}$$
$$\blacksquare \rightarrow f^{(n)} = \{f_1^{(n)}, \ldots, f_d^{(n)}\}$$

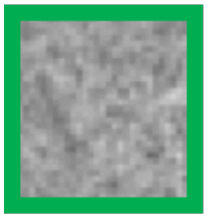


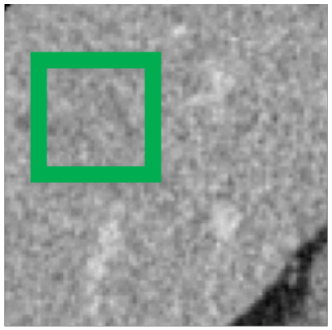
$$\tau_l(f) = \mathbb{E}\left[L|\text{leaf } l\right]$$

Importance of context for features

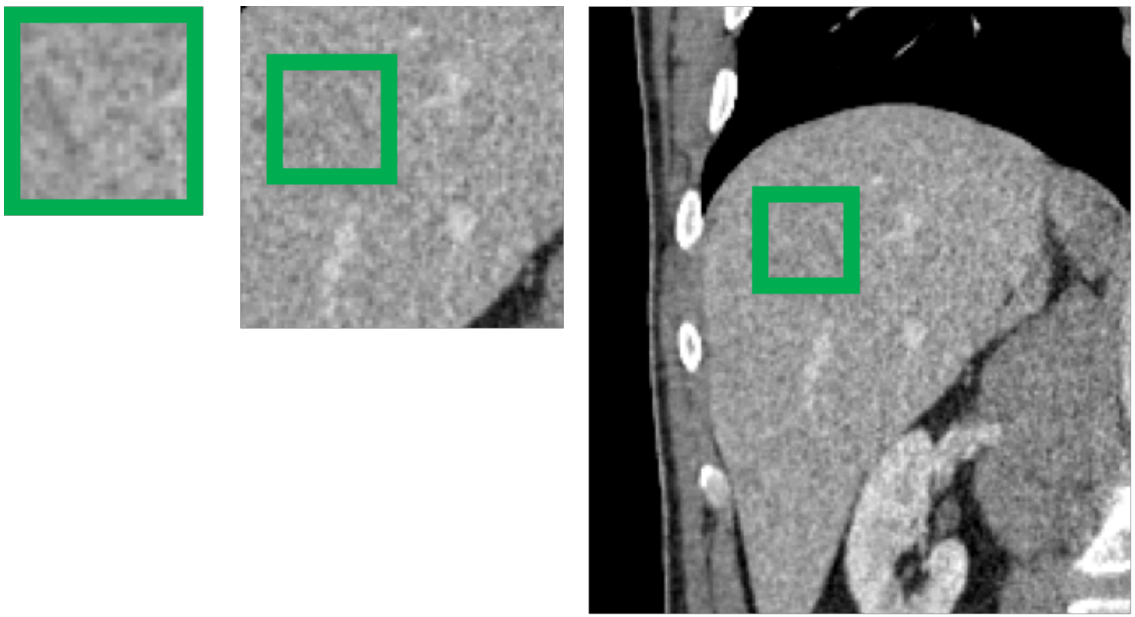


Importance of context for features

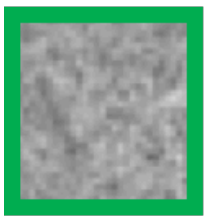
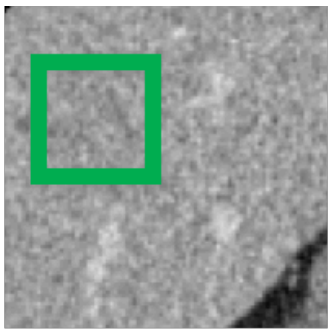
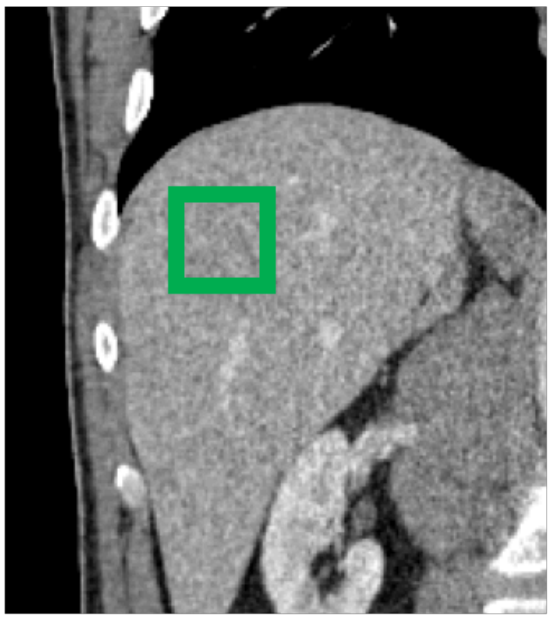
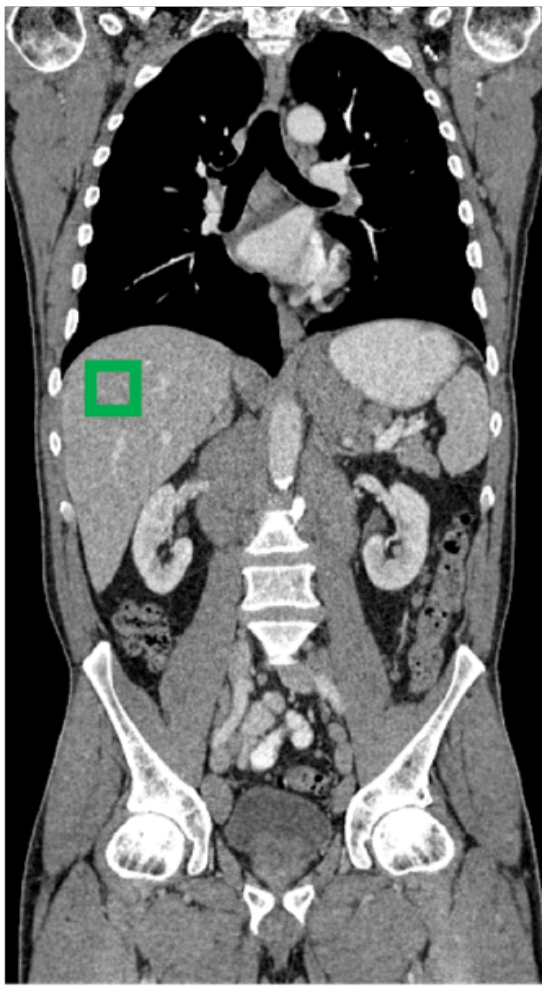




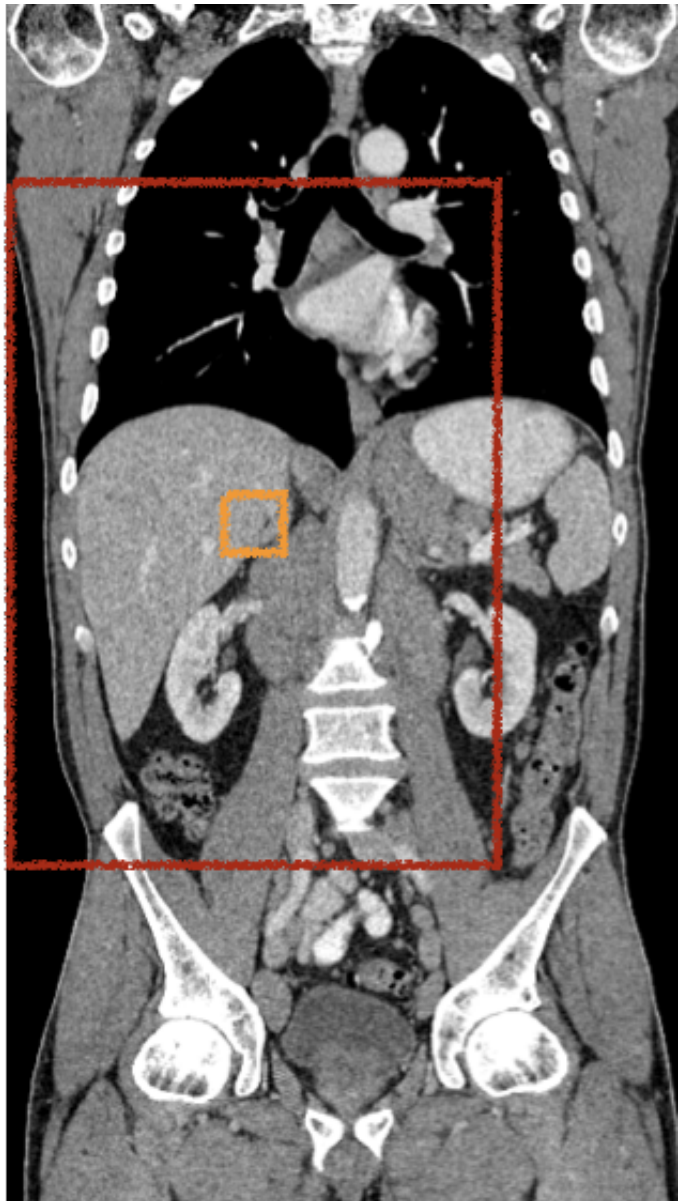
Importance of context for features



Importance of context for features

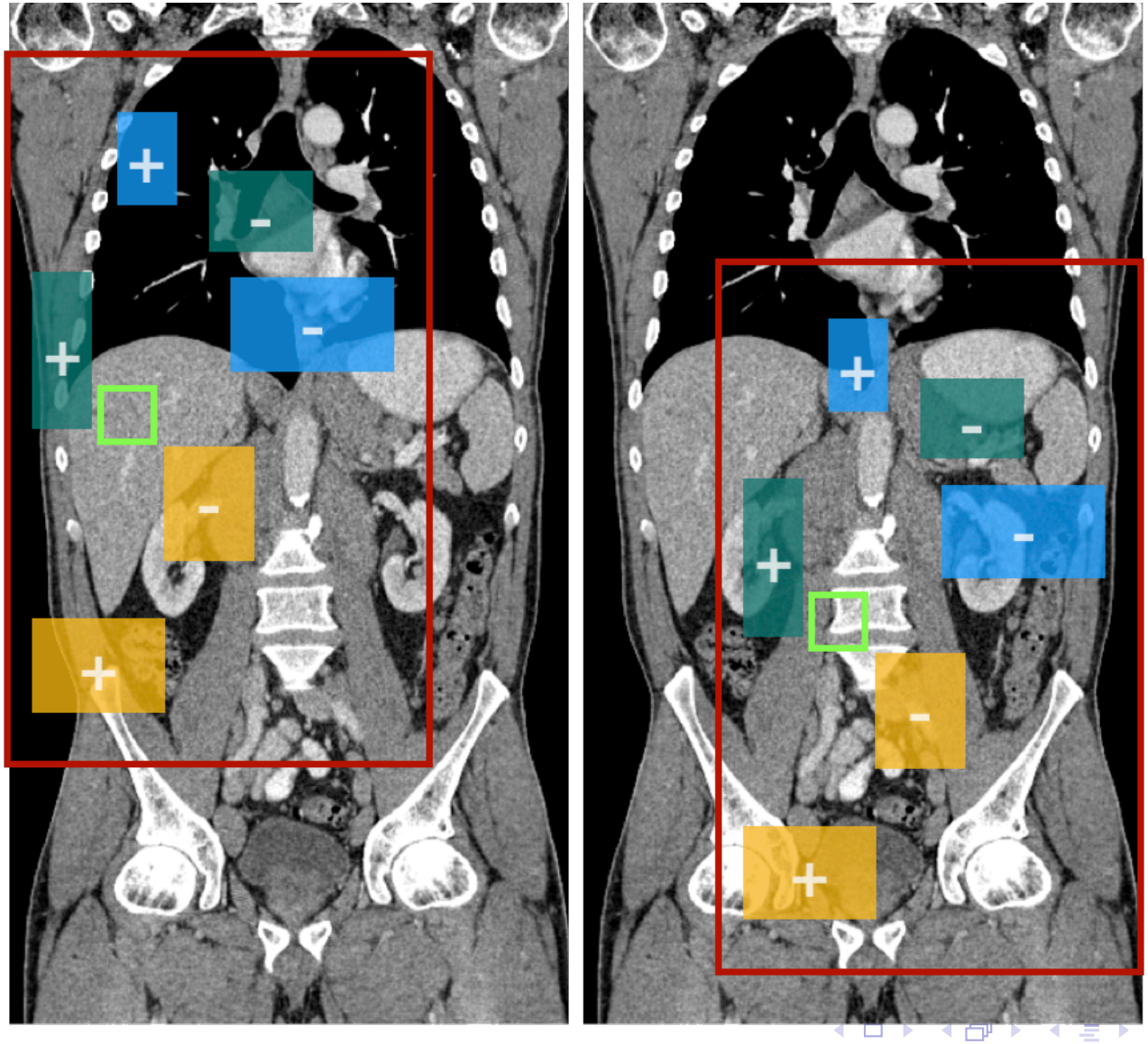





Encoding context

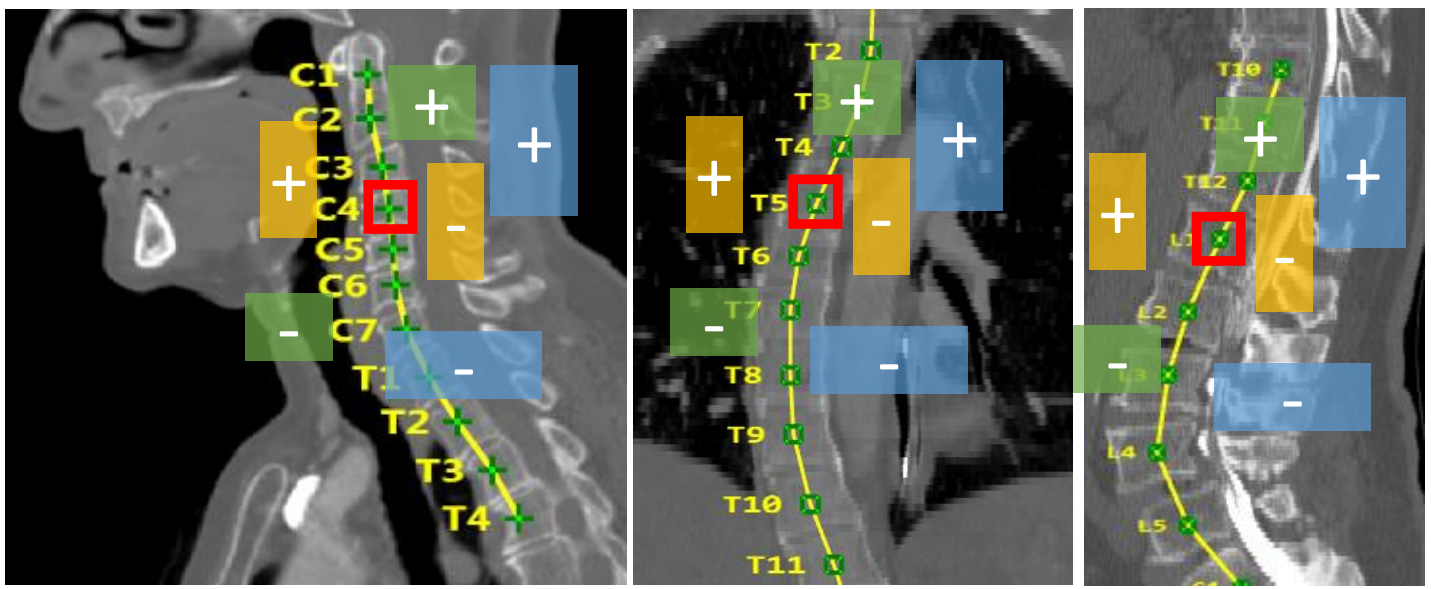


- To represent the orange box, extract features from the red one
- Only simple features, i.e. intensity values at random locations
- One common choice is to use *integral images*
- Encodes context around the point
- Removes need for registration to a common template
- Extract LOTS of features, the algorithm will select the best ones during the training.

Simple features to encode context



Same approach not just for segmentation



Analysis

- Easy to implement
- VERY efficient at test time
- Technology behind first generation Kinect
- No registration

Analysis

- Easy to implement
- VERY efficient at test time
- Technology behind first generation Kinect
- No registration
- Lots of parametric choices
- Needs large number of examples
- Training can take time

Question / Discussion

What context was used and how in:

- Pixel-wise EM segmentation?

Question / Discussion

What context was used and how in:

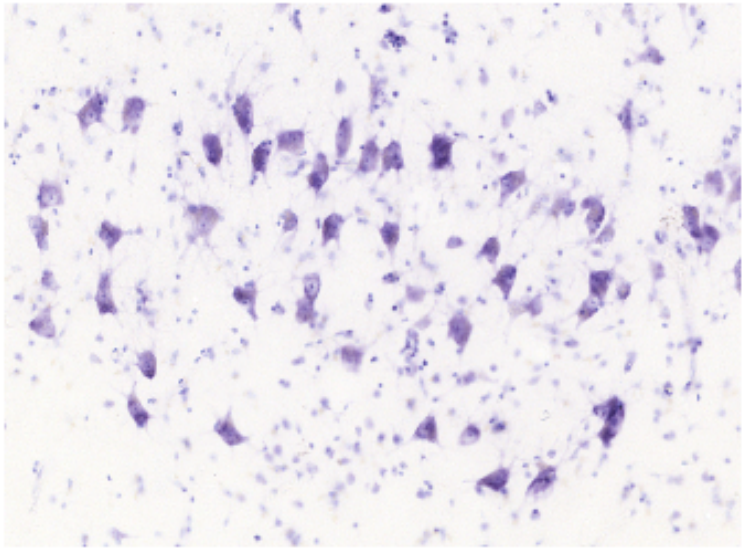
- Pixel-wise EM segmentation?
- Patch-based segmentation?

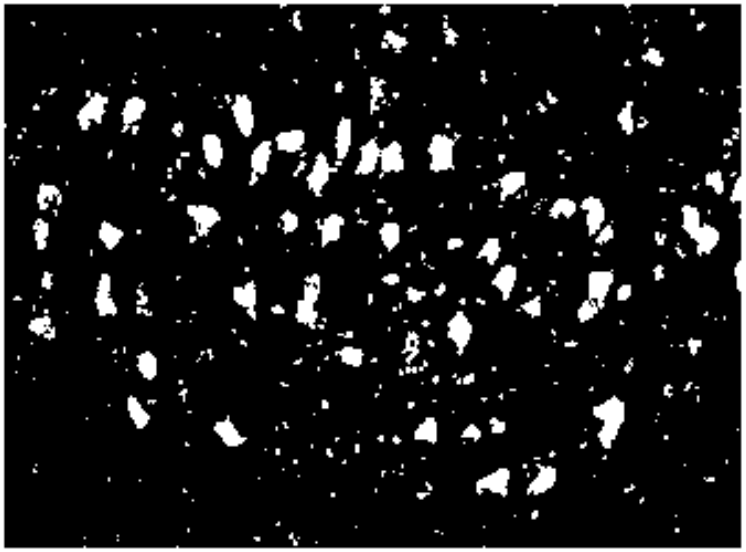
Question / Discussion

What context was used and how in:

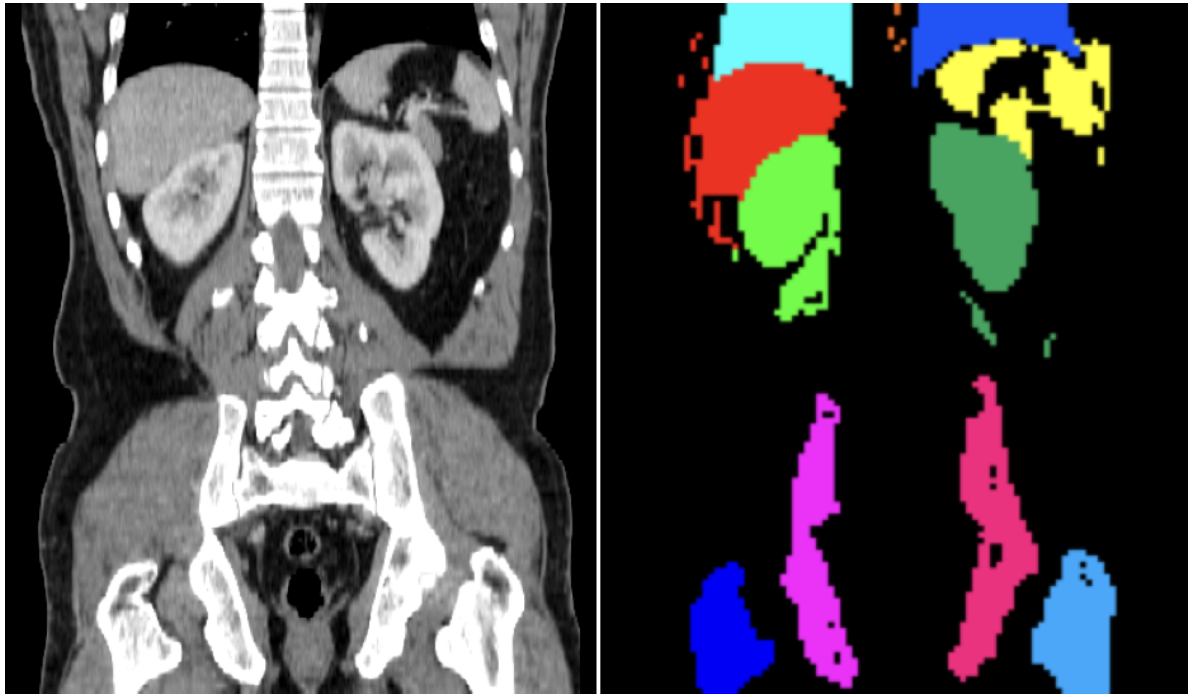
- Pixel-wise EM segmentation?
- Patch-based segmentation?
- Atlas-based segmentation?

Post-processing



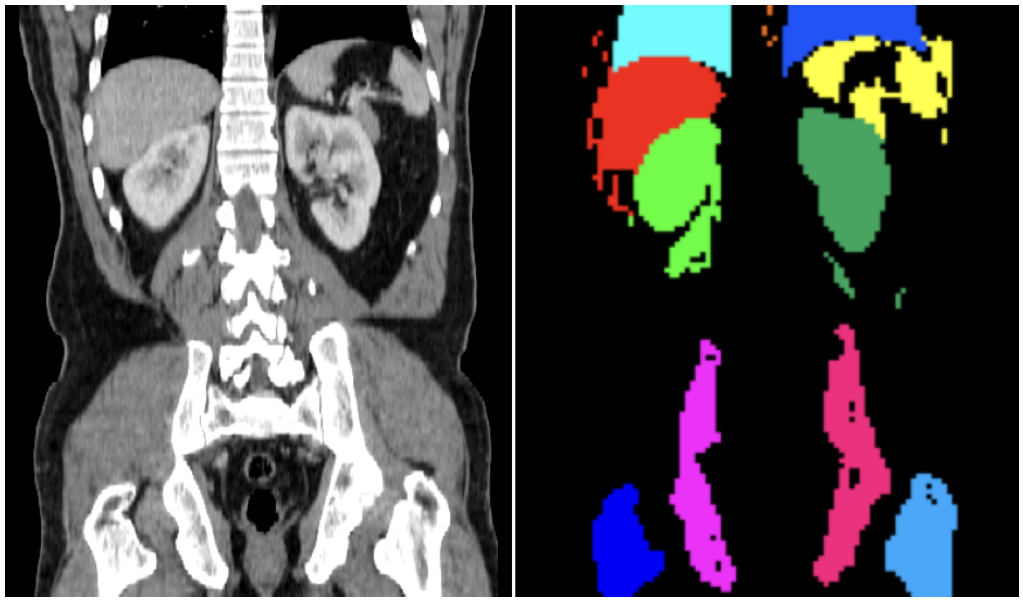


- Isolated islands
- Broken structures
- Some causes:
  - noise, uninteresting small structures



- Isolated islands
- Broken structures
- Some causes:
  - noise, uninteresting small structures
  - low contrast boundaries
  - intensity overlap between structures

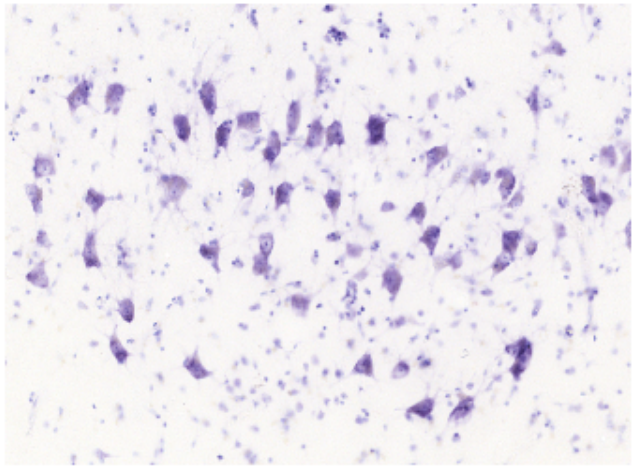
Need for extra consistency

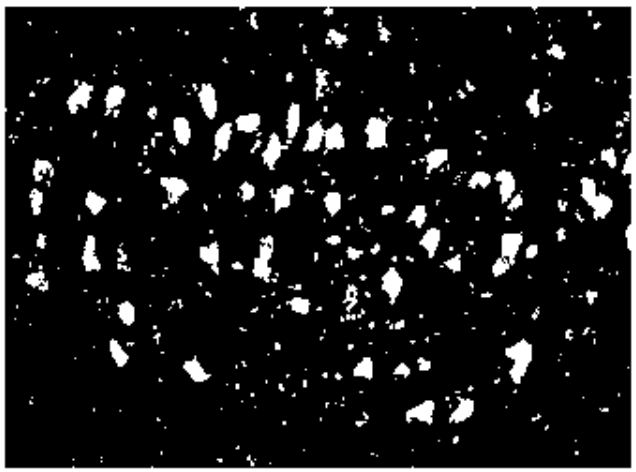


Main approaches to integrate consistency

- Smoothing the images before hand / median filtering the predictions
- Morphological operations
- Random fields
  - Markov random fields
  - Conditional random fields

Morphological operations on binary images





- Operations on the predicted segmentation
- Remove islands
- Fill in gaps
- Extensions to other gray level and multi-class possible

Basic operations in mathematical morphology

- Shift-invariant
- Non-linear
- Based on neighboring pixels defined through structural elements
- Applied in a sliding window approach, e.g. structural element slides across the image
- View binary images as sets of pixels
- Defined in binary but extended to gray-level images
- Two main operations

Dilation

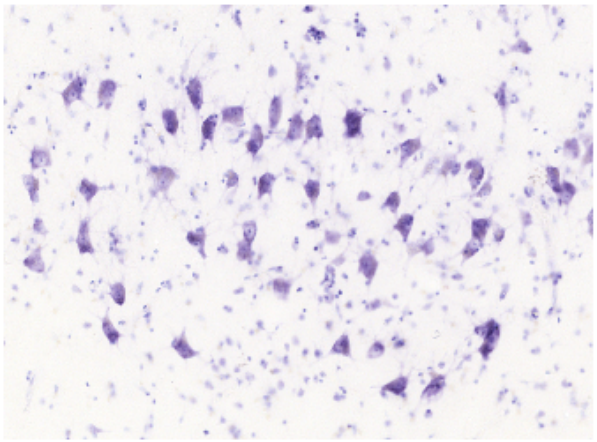
$$C = A \oplus B \triangleq \{x | B_x \cap A \neq 0\}$$

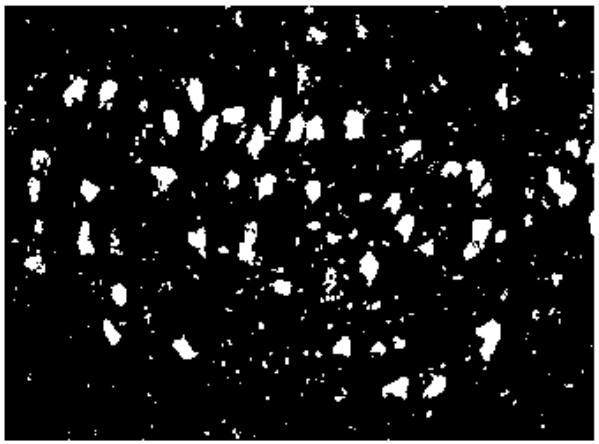
Erosion

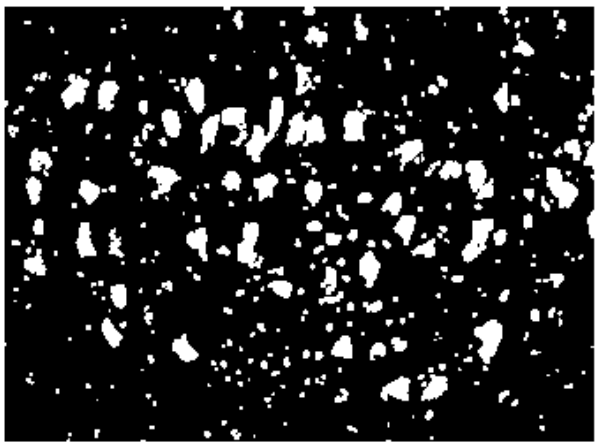
$$C = A \ominus B \triangleq \{x | B_x \subseteq A\}$$

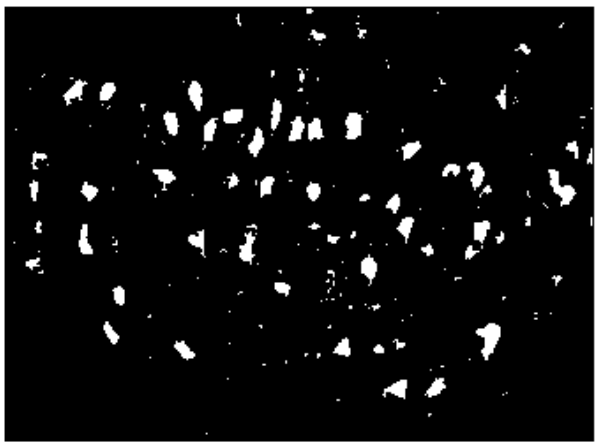
A is the image and B is a structural element, e.g. $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

Dilation and Erosion - $B = \mathbf{1}_{7x7}$





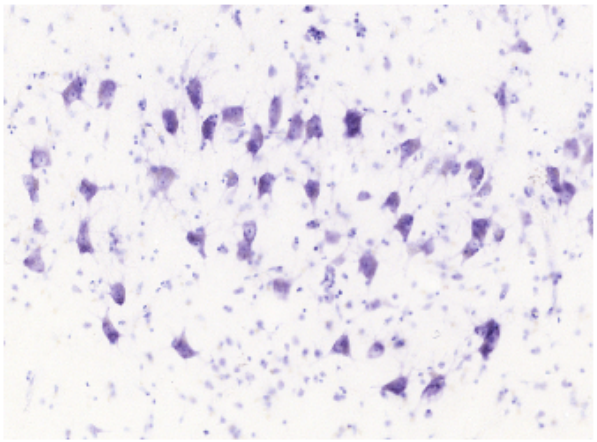


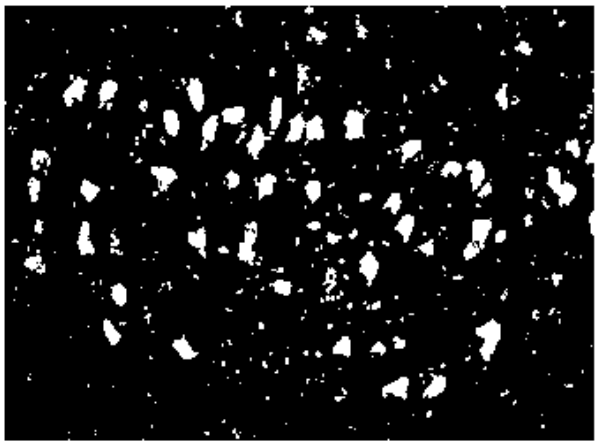


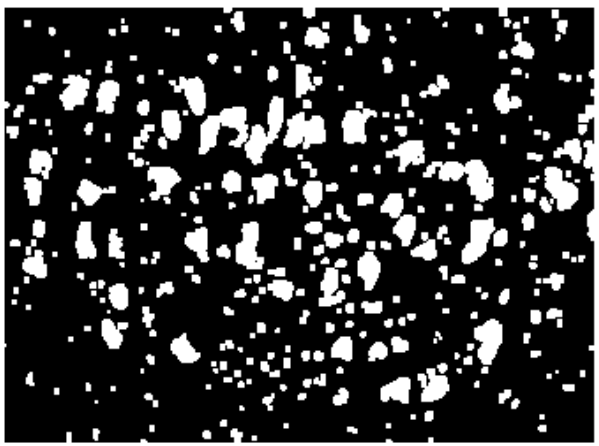
Dilated

Eroded

Dilation and Erosion - $B = \mathbf{1}_{13x13}$







Dilated

Eroded

Derived operations

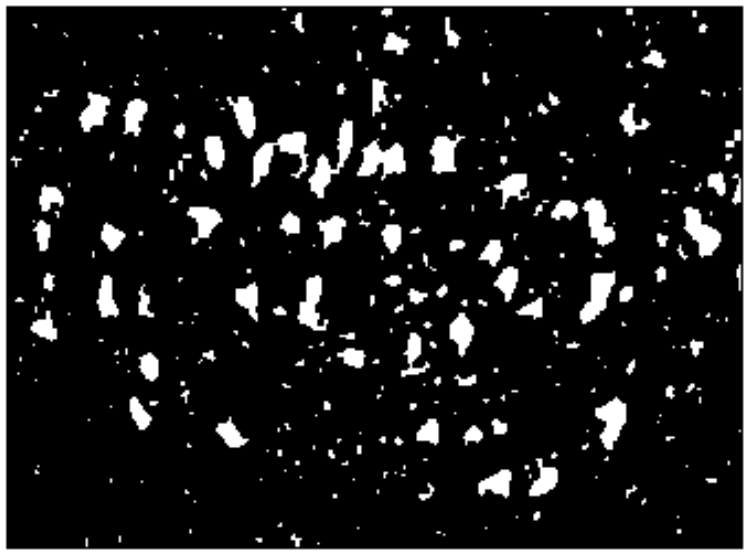
- Combining the two to get useful operations

Opening

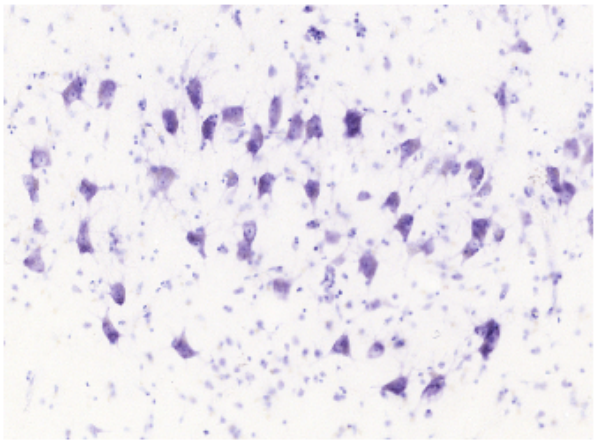
$$(A \ominus B) \oplus B$$

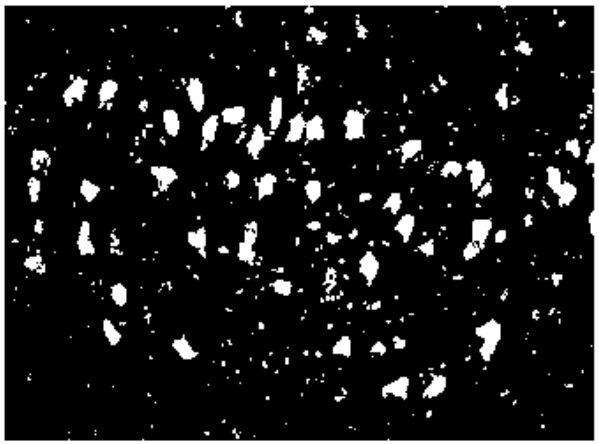
Closing

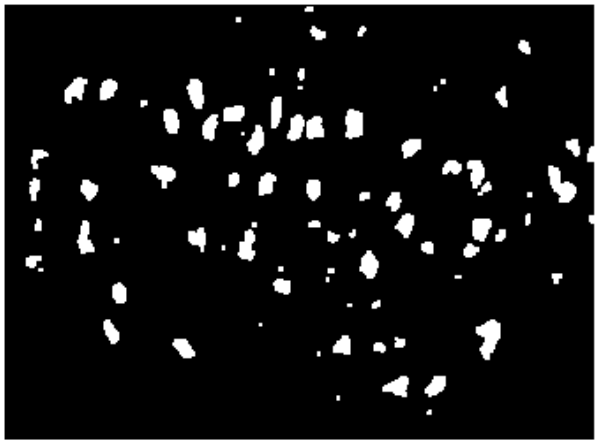
$$(A \oplus B) \ominus B$$

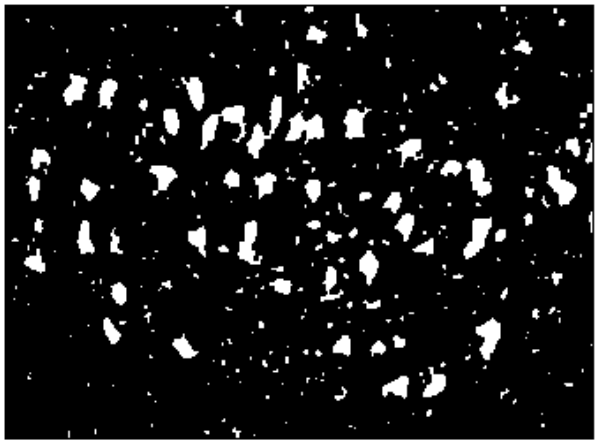


Opening and Closing are useful - $B = \mathbf{1}_{13x13}$







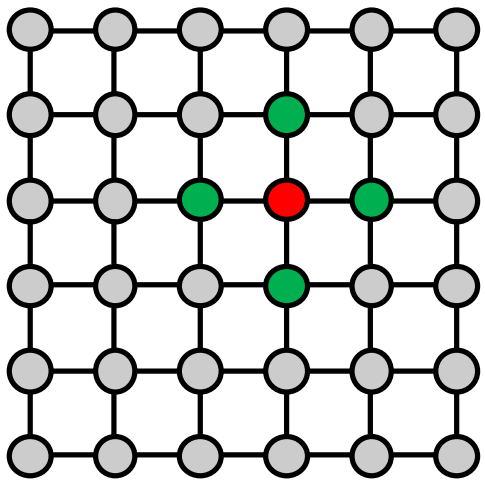


Opening

Closing

Neighborhood consistency

- Formulating neighborhood consistency in a probabilistic model / energy model



- Main idea is to punish inconsistency in labeling of neighboring voxels
- We will use the Markovian property to define the punishment
- Let's start with the probabilistic formulation

Markov Random Fields: principle

- So far our probabilistic models were mostly a mixture model. Let's focus on the joint distribution of labels and intensities

$$p(I(x)) = \sum_{n=1}^{N} p(I(x)|c(x)=n)p(c(x)=n)$$

Markov Random Fields: principle

- So far our probabilistic models were mostly a mixture model. Let's focus on the joint distribution of labels and intensities

$$
\begin{aligned}
p(I(x)) &= \sum_{n=1}^{N} p(I(x)|c(x)=n)p(c(x)=n) \\
p(I(x),c(x)) &= p(I(x)|c(x))p(c(x))
\end{aligned}
$$

Markov Random Fields: principle

- So far our probabilistic models were mostly a mixture model. Let's focus on the joint distribution of labels and intensities

$$
\begin{aligned}
p(I(x)) &= \sum_{n=1}^{N} p(I(x)|c(x)=n)p(c(x)=n) \\
p(I(x),c(x)) &= p(I(x)|c(x))p(c(x)) \\
p(I,c) &= \prod_{x\in\Omega} p(I(x)|c(x))p(c(x))
\end{aligned}
$$

where all pixels are independent.

Markov Random Fields: principle

- So far our probabilistic models were mostly a mixture model. Let's focus on the joint distribution of labels and intensities

$$\begin{aligned} p(I(x)) &= \sum_{n=1}^{N} p(I(x)|c(x)=n)p(c(x)=n) \\ p(I(x),c(x)) &= p(I(x)|c(x))p(c(x)) \\ p(I,c) &= \prod_{x\in\Omega} p(I(x)|c(x))p(c(x)) \end{aligned}$$

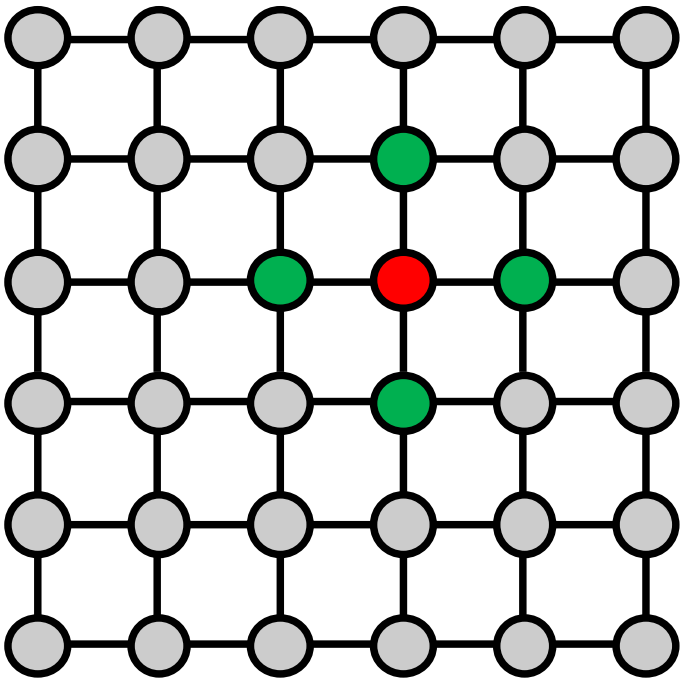
  where all pixels are independent.

- In order to model connection between pixels we change the prior

$$p(I,c) = p(c)\prod_{x\in\Omega} p(I(x)|c(x)).$$

  Joint distribution changes.

Markovian property



- MRF sets up the prior distribution based on the Markovian property
- Specific form depends on the neighborhood structure

$$\begin{aligned} G(x) &= \text{neighbors of } x \\ p(c(x)|c(/x)) &= p(c(x)|c(G(x))) \end{aligned}$$

$p(c(x))$ only depends on its immediate neighbors if all others are given.

Energy formulation

$$p(c(x)|c(/x)) \quad = \quad p(c(x)|c(G(x)))$$

- We do not directly model the above distribution

Energy formulation

$$p(c(x)|c(/x)) \quad = \quad p(c(x)|c(G(x)))$$

- We do not directly model the above distribution
- A common way to define it is through defining an energy:

$$E(c) = \sum_{x\Omega} \sum_{y \in G(x)} d(c(x), c(y))$$

$d(c(x), c(y))$ is a distance between the labels.

Energy formulation

$$p(c(x)|c(/x)) \quad = \quad p(c(x)|c(G(x)))$$

- We do not directly model the above distribution
- A common way to define it is through defining an energy:

$$E(c) = \sum_{x\Omega} \sum_{y \in G(x)} d(c(x), c(y))$$

$d(c(x), c(y))$ is a distance between the labels.

- Given the energy we can define a probability distribution

$$p(c) = \frac{1}{Z} \exp\{-E(c)\}$$

Z is the normalization constant. Lower distance means higher probability. It enforces consistency.

Gibbs distribution

$$E(c) = \sum_{x\Omega} \sum_{y \in G(x)} d(c(x), c(y)) \longleftrightarrow p(c) = \frac{1}{Z} \exp\{-E(c)\}$$

Is this even allowed?

Gibbs distribution

$$E(c) = \sum_{x\Omega} \sum_{y \in G(x)} d(c(x), c(y)) \longleftrightarrow p(c) = \frac{1}{Z} \exp\{-E(c)\}$$

Is this even allowed?

Hammersley-Clifford Theorem

Any probability distribution that satisfies a Markovian property is a Gibbs distribution for an appropriate locally-defined energy and vice-versa.

Segmentation via posterior maximization

Given the energy function $E(c)$

$$p(c) = \frac{1}{Z} \exp\{-E(c)\}$$

Posterior distribution becomes

$$p(c|I) = \frac{p(I|c)p(c)}{p(I)}$$

Note that the posterior is not point-wise but for all the pixels jointly.

Segmentation via posterior maximization

Given the energy function $E(c)$

$$p(c) = \frac{1}{Z} \exp\{-E(c)\}$$

Posterior distribution becomes

$$p(c|I) = \frac{p(I|c)p(c)}{p(I)}$$

Note that the posterior is not point-wise but for all the pixels jointly. Segmentation via posterior maximization:

$$\begin{aligned} \arg_c \max p(c|I) &= \arg_c \max p(I|c)p(c) \\ &= \arg_c \max \exp\{-E(c)\} \prod_{x \in \Omega} p(I(x)|c(x)) \end{aligned}$$

assuming given the label of a voxel, its intensity is independent from the intensities of the other voxels.

Posterior maximization - energy minimization

$$\arg_c \max p(c|I) = \arg_c \max \exp\{-E(c)\} \prod_{x \in \Omega} p(I(x)|c(x))$$

Posterior maximization - energy minimization

$$\arg_c \max p(c|I) = \arg_c \max \exp\{-E(c)\} \prod_{x \in \Omega} p(I(x)|c(x))$$

If the data model is also exponential of another energy

$$p(I(x)|c(x)) \propto \exp\{-g(I(x)|\theta_{c(x)})\}$$

Posterior maximization - energy minimization

$$\arg_c \max p(c|I) = \arg_c \max \exp\{-E(c)\} \prod_{x \in \Omega} p(I(x)|c(x))$$

If the data model is also exponential of another energy

$$p(I(x)|c(x)) \propto \exp\{-g(I(x)|\theta_{c(x)})\}$$

Then we end up with an energy minimization formulation

$$\begin{aligned}
\arg_c \max p(c|I) &= \arg_c \max \ln p(c|I) \\
&= \arg_c \max \left\{ -\sum_{x \in \Omega} g(I(x)|\theta_{c(x)}) - \sum_{x \in \Omega} \sum_{y \in G(x)} d(c(x), c(y)) \right\} \\
&= \arg_c \min \left\{ \sum_{x \in \Omega} g(I(x)|\theta_{c(x)}) + \sum_{x \in \Omega} \sum_{y \in G(x)} d(c(x), c(y)) \right\}
\end{aligned}$$

Different terms

$$\arg_c \min \underbrace{\sum_{x \in \Omega} g(I(x)|\theta_{c(x)})}_{\text{Unary term}} + \underbrace{\sum_{x \in \Omega} \sum_{y \in G(x)} d(c(x), c(y))}_{\text{Pairwise term}}$$

- Unary term is also the data term in this model
- However it could also include any unary prior information on the labels.
- Pairwise term is consistency term

Simple example

Observation model

$$p(I(x)|c(x)) = \mathcal{N}(I(x)|\mu_{c(x)}, \Sigma_{c(x)})$$

where $g(I(x)|\theta_{c(x)}) = (I(x) - \mu_{c(x)})^T \Sigma_{c(x)}^{-1} (I(x) - \mu_{c(x)})$. Energy model for the prior distribution - **Ising / Potts Model**

$$d(c(x), c(y)) = \lambda \delta(c(x) \neq c(y)) = \begin{cases} 0, & c(x) = c(y) \\ 1, & c(x) \neq c(y) \end{cases}$$

Segmentation is the solution of

$$\arg_c \min \sum_{x \in \Omega} (I(x) - \mu_{c(x)})^T \Sigma_{c(x)}^{-1} (I(x) - \mu_{c(x)}) + \lambda \sum_{x \in \Omega} \sum_{y \in G(x)} \delta(c(x) \neq c(y))$$

where λ becomes a trade-off parameter between data fidelity and neighborhood consistency.

Optimization

$$p(c|I) = \frac{p(I|c)p(c)}{p(I)}$$

$$\arg_c \max p(I|c)p(c)$$

$$\arg_c \min \sum_{x\in\Omega}(I(x)-\mu_{c(x)})^T\Sigma^{-1}_{c(x)}(I(x)-\mu_{c(x)})+\lambda\sum_{x\in\Omega}\sum_{y\in G(x)}\delta(c(x)\neq c(y))$$

- Very difficult to compute the posterior distribution
- Difficult to solve the optimization exactly in 2D and higher
- NP-hard [Boykov, Veksler and Zabih 2001 TPAMI]
- Approximate energy minimization and posterior sampling
  - Gibbs sampling [Geman and Geman 1984]
  - Iterated conditional models (ICM) [Ferrari et al. 1995]
  - Graph Cuts [Boykov, Veksler and Zabih 2001]
  - ...

Stochastic relaxation / Gibbs sampling: principle

- Iterative method to approximate solution
- Monte-Carlo sampling method
- Sample from the posterior distribution is hard!

$$p(c|I) = \frac{p(I|c)p(c)}{p(I)} = \frac{p(I|c)p(c)}{\sum_c p(I|c)p(c)}$$

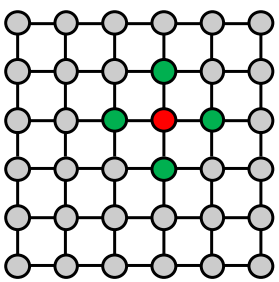
Stochastic relaxation / Gibbs sampling: principle

- Iterative method to approximate solution
- Monte-Carlo sampling method
- Sample from the posterior distribution is hard!

$$p(c|I) = \frac{p(I|c)p(c)}{p(I)} = \frac{p(I|c)p(c)}{\sum_c p(I|c)p(c)}$$

- Much easier to sample from this

$$p(c(x)|c(/x), I) = \frac{p\left(I(x)|c(x)\right) p\left(c(x)|c(G(x))\right)}{\sum_{c(x)} p\left(I(x)|c(x)\right) p\left(c(x)|c(G(x))\right)}$$



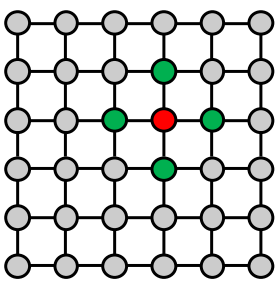
Stochastic relaxation / Gibbs sampling: principle

- Iterative method to approximate solution
- Monte-Carlo sampling method
- Sample from the posterior distribution is hard!

$$p(c|I) = \frac{p(I|c)p(c)}{p(I)} = \frac{p(I|c)p(c)}{\sum_c p(I|c)p(c)}$$

- Much easier to sample from this

$$p(c(x)|c(/x), I) = \frac{p\left(I(x)|c(x)\right) p\left(c(x)|c(G(x))\right)}{\sum_{c(x)} p\left(I(x)|c(x)\right) p\left(c(x)|c(G(x))\right)}$$



- Start from a random c_0 assignment, such as $\arg_c \max \prod_{x\in\Omega} p(I(x)|c(x))$

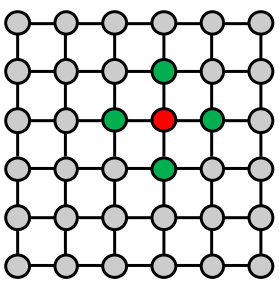
Stochastic relaxation / Gibbs sampling: principle

- Iterative method to approximate solution
- Monte-Carlo sampling method
- Sample from the posterior distribution is hard!

$$p(c|I) = \frac{p(I|c)p(c)}{p(I)} = \frac{p(I|c)p(c)}{\sum_c p(I|c)p(c)}$$

- Much easier to sample from this

$$p(c(x)|c(/x), I) = \frac{p\left(I(x)|c(x)\right) p\left(c(x)|c(G(x))\right)}{\sum_{c(x)} p\left(I(x)|c(x)\right) p\left(c(x)|c(G(x))\right)}$$



- Start from a random c_0 assignment, such as $\arg_c \max \prod_{x\in\Omega} p(I(x)|c(x))$
- Sample from the posterior of each voxel given the others

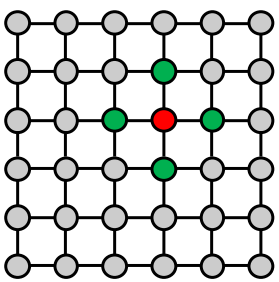
Stochastic relaxation / Gibbs sampling: principle

- Iterative method to approximate solution
- Monte-Carlo sampling method
- Sample from the posterior distribution is hard!

$$p(c|I) = \frac{p(I|c)p(c)}{p(I)} = \frac{p(I|c)p(c)}{\sum_c p(I|c)p(c)}$$

- Much easier to sample from this

$$p(c(x)|c(/x), I) = \frac{p\left(I(x)|c(x)\right) p\left(c(x)|c(G(x))\right)}{\sum_{c(x)} p\left(I(x)|c(x)\right) p\left(c(x)|c(G(x))\right)}$$



- Start from a random c_0 assignment, such as $\arg_c \max \prod_{x\in\Omega} p(I(x)|c(x))$
- Sample from the posterior of each voxel given the others
- Iterate over all voxels and some number of times

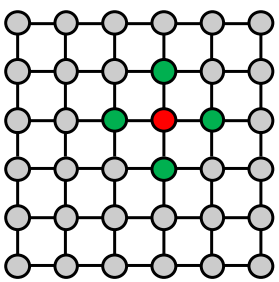
Stochastic relaxation / Gibbs sampling: principle

- Iterative method to approximate solution
- Monte-Carlo sampling method
- Sample from the posterior distribution is hard!

$$p(c|I) = \frac{p(I|c)p(c)}{p(I)} = \frac{p(I|c)p(c)}{\sum_c p(I|c)p(c)}$$

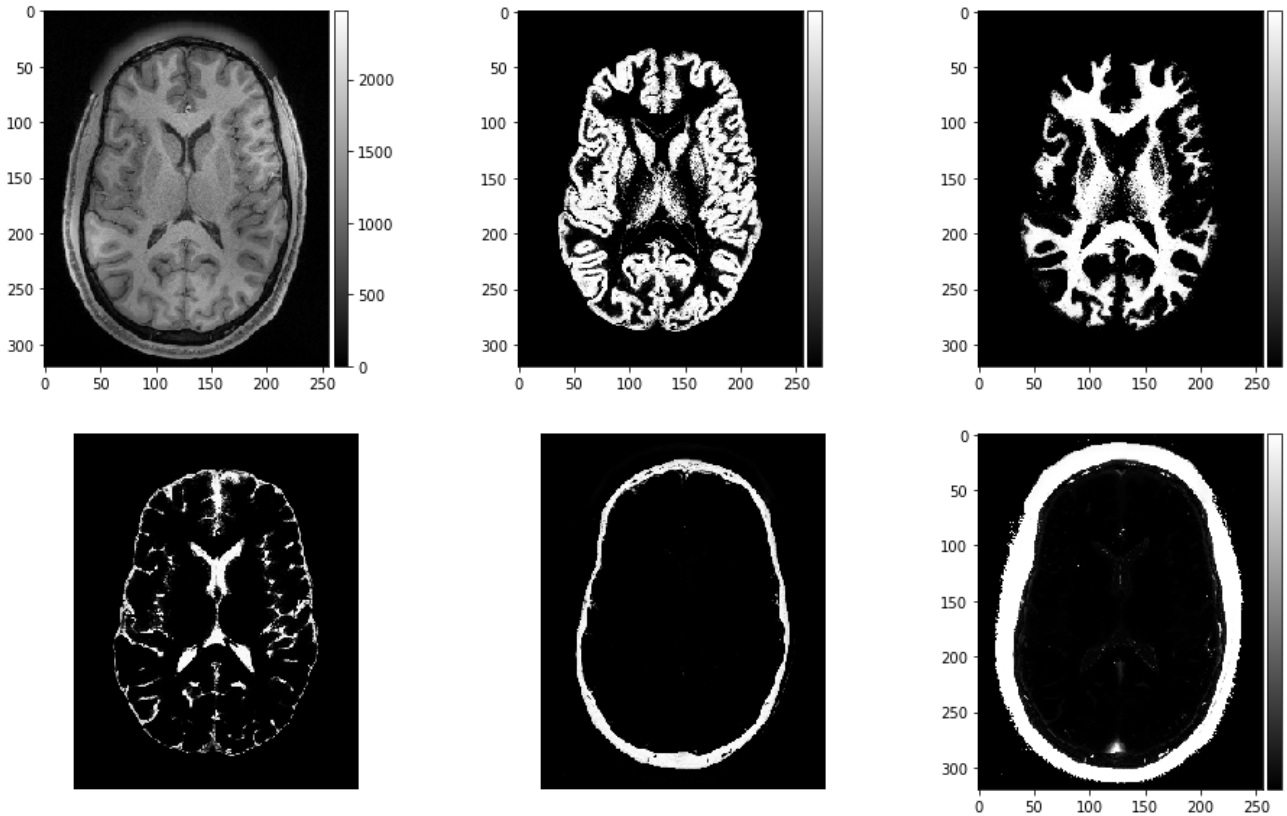
- Much easier to sample from this

$$p(c(x)|c(/x), I) = \frac{p\left(I(x)|c(x)\right) p\left(c(x)|c(G(x))\right)}{\sum_{c(x)} p\left(I(x)|c(x)\right) p\left(c(x)|c(G(x))\right)}$$

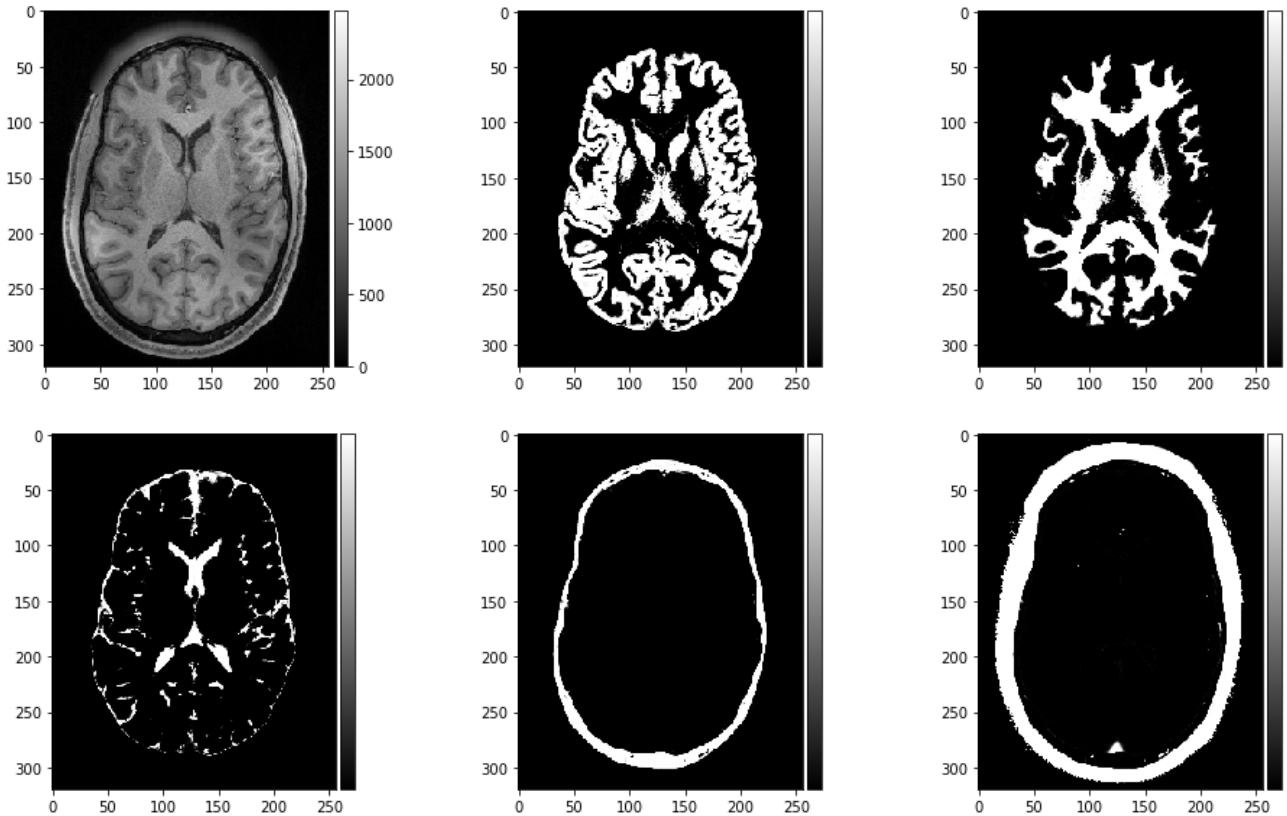


- Start from a random c_0 assignment, such as $\arg_c \max \prod_{x\in\Omega} p(I(x)|c(x))$
- Sample from the posterior of each voxel given the others
- Iterate over all voxels and some number of times
- Convergence properties [Geman and Geman 1984]

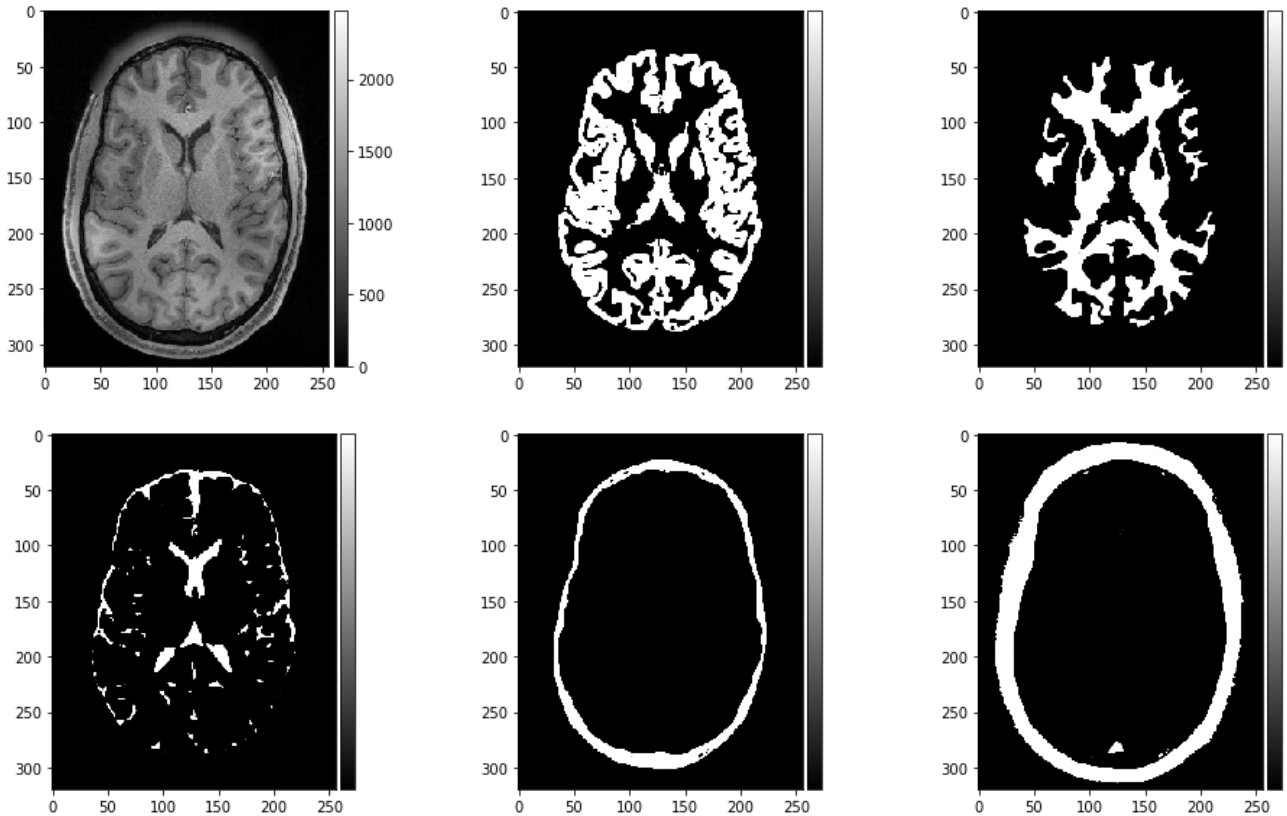
Example results - without MRF



Example results - with MRF



Example results - with MRF higher λ



Analysis of Gibbs Sampling

- Very simple implementation and effective
- Solution depends on parameters
- Solution depends on initial conditions
- May not converge to a pleasing result
- Convergence may be slow
- Stochastic in nature - every time you run you get something else
- Other methods exist and may provide better results
- Complicated optimization problem

Conditional random fields

- The energy formulation of MRF's can also be used in a discriminative way
- Instead of a probabilistic generative model we can consider a model that directly predicts $p(c(x)|I)$ without the Bayesian treatment

$$\arg_c \min \underbrace{\sum_{x \in \Omega} g(c(x)|I)}_{\text{Unary term}} + \underbrace{\sum_{x \in \Omega} \sum_{y \in G(x)} d(c(x), c(y))}_{\text{Pairwise term}}$$

- This can be used on its own for segmentation by explicitly modeling g - any discriminative model
- It can be used with ML algorithm predictions
- Used regularly to "clean up" the ML segmentation predictions