Basic Segmentation

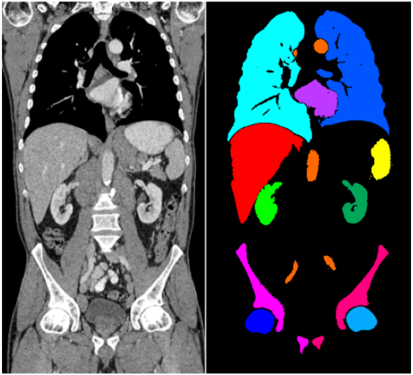
Ender Konukoglu

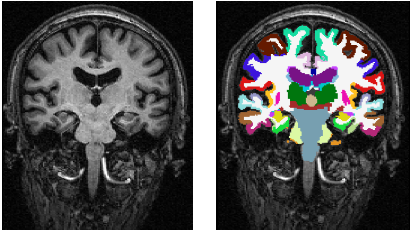
ETH Zürich

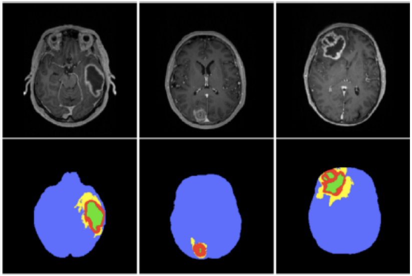
March 24, 2020

Basic Segmentation

From image intensities to anatomical structures and semantic information







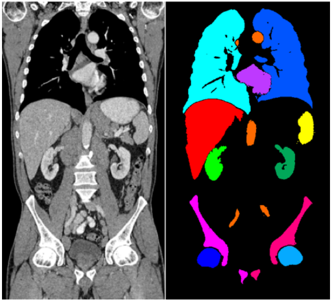
Outline

- Segmentation in general
- From thresholding to K-means
- Expectation-Maximization Segmentation
- Exercises

Section 1

Segmentation

Segmentation principle



- At each pixel / voxel assign a label
- Labels can be
  - Organ: liver, spine, ...
  - Part of organs: individual vertebrae, ...
  - Lesions: tumors, pathologies, ...
- Information at each pixel
  - Intensity at and around the pixel
  - Intensity at different modalities

$x \in \Omega \subset \mathbb{R}^d$, $d = 2$ or $d = 3$
$f(x) = [I(x), I(\mathcal{N}(x)), J(\mathcal{N}(x)), \dots]$: Features
$L(x) \in \{0, \dots, N\}$: Labels
$L(x) \approx S(f(x))$: Segmentation approximating labels

Segmentation techniques overview

- Thresholding and histogram
- Clustering
  - K-means
  - Unsupervised learning
  - Non-parametric modeling
  - ...
- Graph partitioning methods
  - Watershed
  - Graph-cuts
  - Random walker
  - Minimum spanning forest
  - ...
- Region growing
- Variational and PDE-based
  - Chan-Vese model
  - Mumford-Shah model
  - Active-shape models
  - Level sets
  - Fast marching
  - ...
- Discriminative modeling based
  - Random forest
  - Conditional random fields
  - Supervised convolutional neural networks
  - ...
- Generative modeling based
  - Expectation-maximization
  - Markov random fields
  - Atlas-based segmentation
  - Variational inference
  - ...

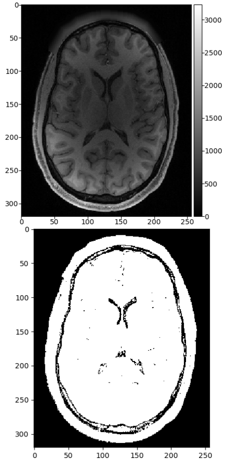
Simpler approaches that are often used

- Thresholding
- K-means - generalizing thresholding
- Expectation-Maximization - principled KNN

Subsection 1

Thresholding

Thresholding

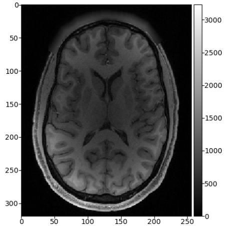


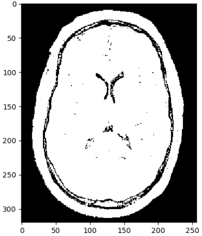
- Simple model

$$L(x) = \begin{cases} 1, & I(x) > \tau \\ 0, & I(x) \leq \tau \end{cases}$$

or vice-versa.

Thresholding





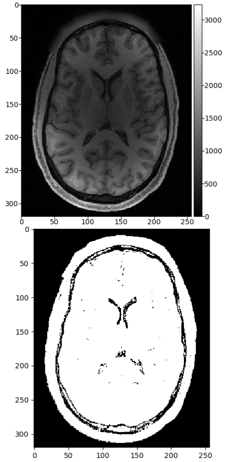
- Simple model

$$L(x) = \begin{cases} 1, & I(x) > \tau \\ 0, & I(x) \leq \tau \end{cases}$$

  or vice-versa.

- Particularly useful in easy foreground-background subtraction

Thresholding



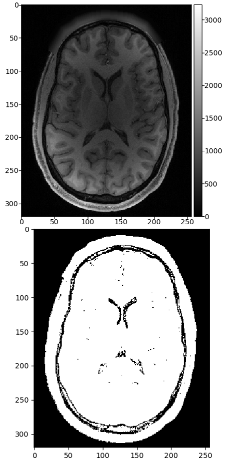
- Simple model

$$L(x) = \begin{cases} 1, & I(x) > \tau \\ 0, & I(x) \leq \tau \end{cases}$$

  or vice-versa.

- Particularly useful in easy foreground-background subtraction
- Preprocessing method for region of interest determination

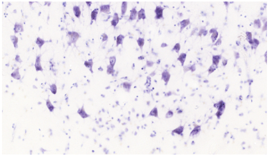
Thresholding



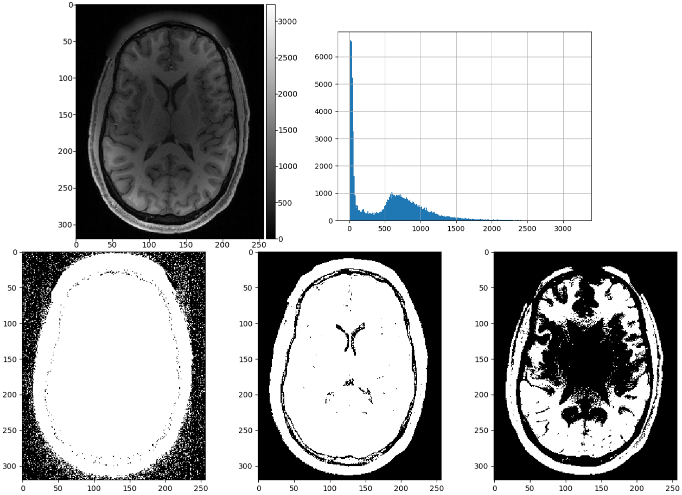
- Simple model

$$L(x) = \begin{cases} 1, & I(x) > \tau \\ 0, & I(x) \leq \tau \end{cases}$$

  or vice-versa.
- Particularly useful in easy foreground-background subtraction
- Preprocessing method for region of interest determination
- Used routinely in histology and other microscopic images where stains provide the necessary contrast

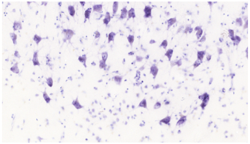


How to determine the threshold: histogram analysis

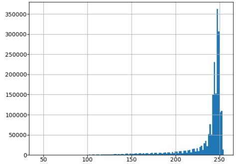


$\tau = 50$ $\tau = 250$ $\tau = 750$

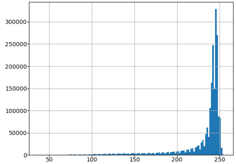
How to determine the threshold: histogram analysis



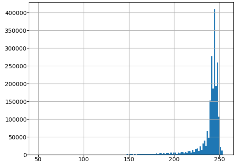
Original image



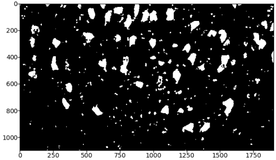
Red



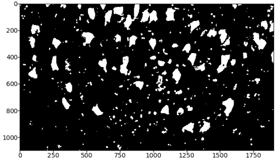
Green



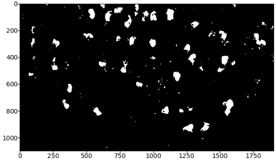
Blue



Red



Green



Blue

How to determine the threshold: automatic methods

- Methods to determine peaks and trough of the histogram

How to determine the threshold: automatic methods

- Methods to determine peaks and trough of the histogram
- Otsu's thresholding method

$$\min p_1\sigma_1^2 + p_2\sigma_2^2$$

where $p_{1,2}$ are the number of pixels in fore and background. $\sigma_{1,2}^2$ are the intensity variances in the groups.

How to determine the threshold: automatic methods

- Methods to determine peaks and trough of the histogram
- Otsu's thresholding method

$$\min p_1\sigma_1^2 + p_2\sigma_2^2$$

  where $p_{1,2}$ are the number of pixels in fore and background. $\sigma_{1,2}^2$ are the intensity variances in the groups.
- Global thresholding (for the entire image) or local thresholding (per ROI or patch)

How to determine the threshold: automatic methods

- Methods to determine peaks and trough of the histogram
- Otsu's thresholding method

  $$\min p_1\sigma_1^2 + p_2\sigma_2^2$$

  where $p_{1,2}$ are the number of pixels in fore and background. $\sigma_{1,2}^2$ are the intensity variances in the groups.
- Global thresholding (for the entire image) or local thresholding (per ROI or patch)
- Image noise can lead to isolated FG or BG islands

How to determine the threshold: automatic methods

- Methods to determine peaks and trough of the histogram
- Otsu's thresholding method

$$\min p_1\sigma_1^2 + p_2\sigma_2^2$$

  where $p_{1,2}$ are the number of pixels in fore and background. $\sigma_{1,2}^2$ are the intensity variances in the groups.
- Global thresholding (for the entire image) or local thresholding (per ROI or patch)
- Image noise can lead to isolated FG or BG islands
- Multiple objects would require multiple thresholds

How to determine the threshold: automatic methods

- Methods to determine peaks and trough of the histogram
- Otsu's thresholding method

$$\min p_1\sigma_1^2 + p_2\sigma_2^2$$

  where $p_{1,2}$ are the number of pixels in fore and background. $\sigma_{1,2}^2$ are the intensity variances in the groups.
- Global thresholding (for the entire image) or local thresholding (per ROI or patch)
- Image noise can lead to isolated FG or BG islands
- Multiple objects would require multiple thresholds
- A nice generalization is K-means algorithm, we will see next...

Subsection 2

K-Means

K-Means segmentation

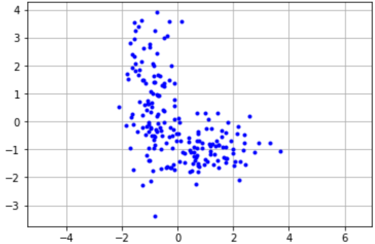
- Unsupervised clustering algorithm

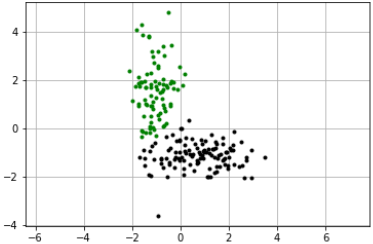
K-Means segmentation

- Unsupervised clustering algorithm
- Voxels/pixels are clustered according to *features*
  - intensity
  - color
  - temporal sequence
  - other features, e.g. gradient, wavelet transform,...

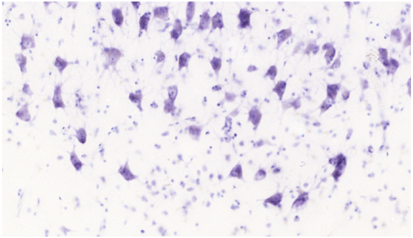
K-Means segmentation

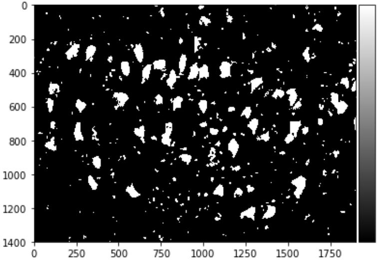
- Unsupervised clustering algorithm
- Voxels/pixels are clustered according to *features*
  - intensity
  - color
  - temporal sequence
  - other features, e.g. gradient, wavelet transform,...
- Automatically assigns each voxel/pixel to one of N clusters
  Assume we have two features (shown in x and y axis), multiple pixels (each point) and two clusters



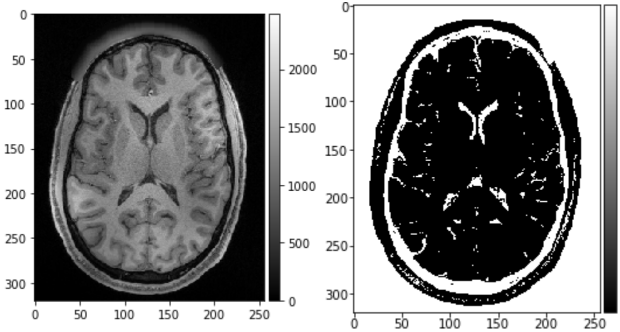


K-Means segmentation - 2 clusters, features=RGB





K-Means segmentation - 2 clusters, features=intensity



How does it work?

- We want to distribute pixels to N groups using features

How does it work?

- We want to distribute pixels to N groups using features
- Criteria:
  - Homogeneity within groups
  - Reducing variance over features within group
  - Take into account multiple features

How does it work?

- We want to distribute pixels to N groups using features
- Criteria:
  - Homogeneity within groups
  - Reducing variance over features within group
  - Take into account multiple features
- Unsupervised - no information on the structures nor groups

How does it work?

- We want to distribute pixels to N groups using features
- Criteria:
  - Homogeneity within groups
  - Reducing variance over features within group
  - Take into account multiple features
- Unsupervised - no information on the structures nor groups
- Iterative algorithm that assigns pixels to groups, computes group means and reassigns based on distance to group centers

K-Means Algorithm

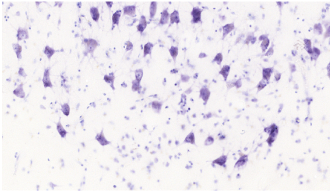
Algorithm 1 K-Means

1: Describe each pixel with features $f(x) \in \mathbb{R}^m$, e.g. for only intensity $m = 1$ and for RGB $m = 3$
2: Randomly choose N points as the group centers, $m_i^0 \in \mathbb{R}^m$, $i = 1, \ldots, N$
3: **while** $\|m_i^t - m_i^{t-1}\| > \epsilon$ for any i **do**
4: Assign groups: $c(x) = \arg_i \min \|f(x) - m_i\|_2$
5: Recompute means:

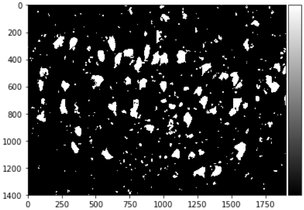
$$m_i = \frac{\sum_{x \in \Omega} \delta_{i=c(x)} f(x)}{\sum_{x \in \Omega} \delta_{i=c(x)}}, \quad \delta_{i=c(x)} = \begin{cases} 0, & i \neq c(x) \\ 1, & i = c(x) \end{cases}$$

6: **end while**

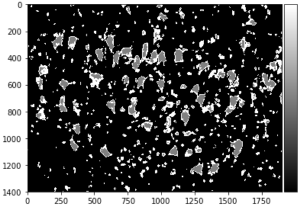
Changing number of clusters



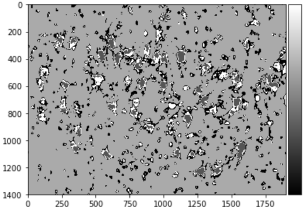
image



2 clusters

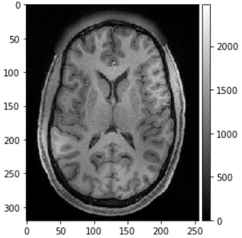


3 clusters

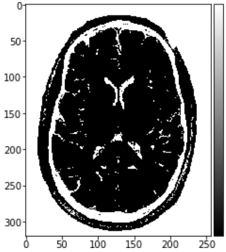


4 clusters

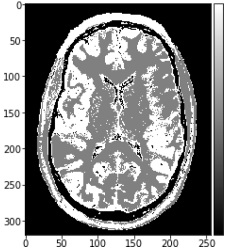
Changing number of clusters



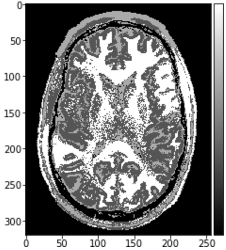
image



2 clusters

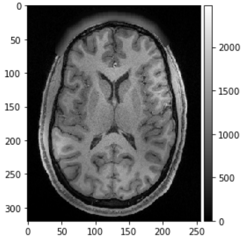


3 clusters

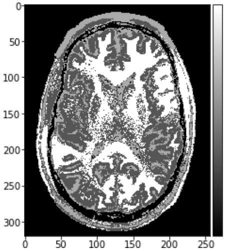


4 clusters

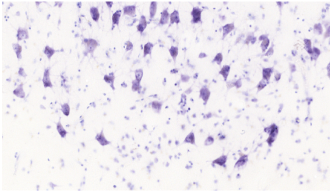
Question: Why does this happen?



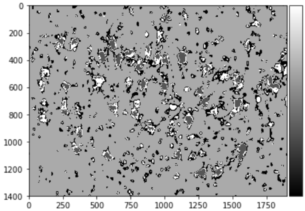
image



4 clusters



image



4 clusters

Remarks

- Very easy to implement

Remarks

- Very easy to implement
- Choice of number of clusters has a major influence
  Methods exist to choose it automatically:
  - Heuristic methods based on variance
  - Non-parametric Bayesian methods

Remarks

- Very easy to implement
- Choice of number of clusters has a major influence
  Methods exist to choose it automatically:
  - Heuristic methods based on variance
  - Non-parametric Bayesian methods
- Initialization is important
  K-Means can get stuck in local minima:
  - Run it multiple times with different initializations
  - Multi-scale methods for robustness

Remarks

- Very easy to implement
- Choice of number of clusters has a major influence
  Methods exist to choose it automatically:
  - Heuristic methods based on variance
  - Non-parametric Bayesian methods
- Initialization is important
  K-Means can get stuck in local minima:
  - Run it multiple times with different initializations
  - Multi-scale methods for robustness
- Probabilistic formulation possible, we'll see it next...

Subsection 3

Expectation-Maximization Segmentation

Mixture Model

- We will work on the same idea as K-Means but take a probabilistic view.

Mixture Model

- We will work on the same idea as K-Means but take a probabilistic view.
- This will allow us to create accurate *atlas*-based segmentation methods and motivate registration.

Mixture Model

- We will work on the same idea as K-Means but take a probabilistic view.
- This will allow us to create accurate *atlas*-based segmentation methods and motivate registration.
- Start by assuming all pixels are independent and model the intensities as a **mixture model**

Mixture Model

- We will work on the same idea as K-Means but take a probabilistic view.
- This will allow us to create accurate *atlas*-based segmentation methods and motivate registration.
- Start by assuming all pixels are independent and model the intensities as a **mixture model**
- Mixture model of features

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

$p(c)$ is the *prior probability* of observing label c. Also called **mixture coefficients**.

Mixture Model

- We will work on the same idea as K-Means but take a probabilistic view.
- This will allow us to create accurate *atlas*-based segmentation methods and motivate registration.
- Start by assuming all pixels are independent and model the intensities as a **mixture model**
- Mixture model of features

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

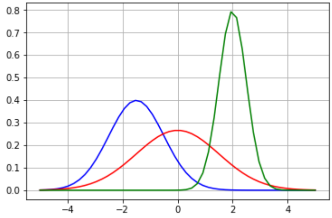
  $p(c)$ is the *prior probability* of observing label c. Also called **mixture coefficients**.

- $p(f|c)$ is the *likelihood model* defined at that point. Often taken as a Gaussian

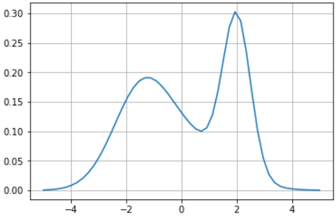
$$p(f|c=n) = \mathcal{N}(f|\mu_n, \Sigma_n) = \frac{1}{\sqrt{2\pi|\Sigma_n|}} \exp(-\frac{1}{2}(f-\mu_n)^T \Sigma_n^{-1}(f-\mu_n))$$

  μ_n and Σ_n are class specific parameters $\rightarrow$ Gaussian mixture model

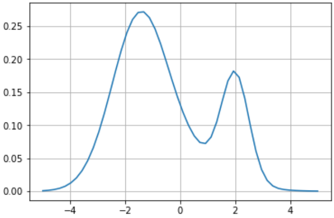
Mixture Model - 1D Example with 3 labels



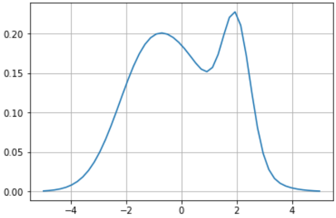
Components



$p(c=0) = p(c=1) = p(c=2) = 1/3$

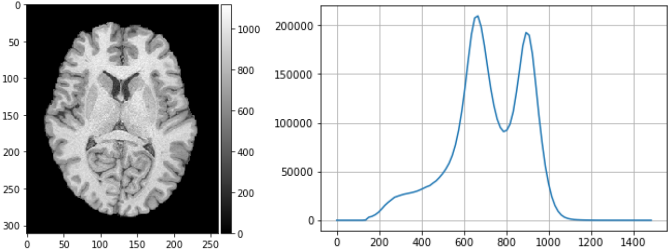


$p(c=0) = 3/5,\ \ p(c=1) = p(c=2) = 1/5$

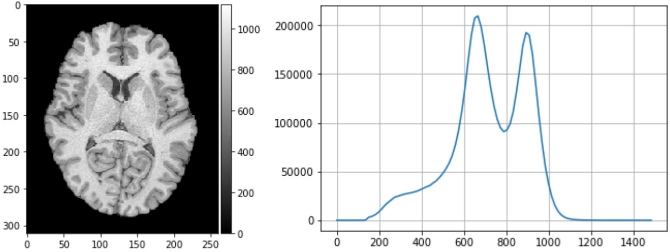


$p(c=1) = 3/5,\ \ p(c=0) = p(c=2) = 1/5$

Does it look like real image distribution?



Does it look like real image distribution?



Gaussian Mixture Model may not be a bad approximate model for the intensities

How to use the mixture model?

- Fit the model parameters to the image intensities in an unsupervised way, we will only assume a number of classes N and assume intensities at different pixels are independent from each other

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

How to use the mixture model?

- Fit the model parameters to the image intensities in an unsupervised way, we will only assume a number of classes N and assume intensities at different pixels are independent from each other

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

  θ is the set of model parameters:
  - $p(c(x)=n)=\pi_n$: class probabilities
  - μ_n, Σ_n: likelihood parameters
- After fitting, segmentation is defined through posterior-distribution using optimal parameters

$$c^*(x) = \arg\max p(c(x)|I(x))$$

How to use the mixture model?

- Fit the model parameters to the image intensities in an unsupervised way, we will only assume a number of classes N and assume intensities at different pixels are independent from each other

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

  θ is the set of model parameters:
  - $p(c(x)=n)=\pi_n$: class probabilities
  - μ_n, Σ_n: likelihood parameters
- After fitting, segmentation is defined through posterior-distribution using optimal parameters

$$c^*(x) = \arg\max p(c(x)|I(x))$$

- We can optimize using *gradient descent* - possible but difficult to optimize

How to use the mixture model?

- Fit the model parameters to the image intensities in an unsupervised way, we will only assume a number of classes N and assume intensities at different pixels are independent from each other

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

  θ is the set of model parameters:
  - $p(c(x) = n) = \pi_n$: class probabilities
  - μ_n, Σ_n: likelihood parameters
- After fitting, segmentation is defined through posterior-distribution using optimal parameters

$$c^*(x) = \arg\max p(c(x)|I(x))$$

- We can optimize using *gradient descent* - possible but difficult to optimize
- Better alternative: Expectation-Maximization

Key idea - two alternating steps

- **E-step** - Expectation-step: Assume a set of likelihood parameters and "soft" assign samples to labels
- **M-step** - Maximization-step: Assume soft assignments and maximize likelihood parameters, e.g. find the best mean and standard deviations of the Gaussians.

Expectation-Maximization - derivation

Expectation-Maximization - derivation

Dropping the dependence on x for simplicity for now and using Bayes' rule

Expectation-Maximization - derivation

Dropping the dependence on x for simplicity for now and using Bayes' rule

$$\log p(I|\theta) = \log p(I, c|\theta) - \log p(c|I, \theta)$$

Expectation-Maximization - derivation

Dropping the dependence on x for simplicity for now and using Bayes' rule

$$\log p(I|\theta) = \log p(I, c|\theta) - \log p(c|I, \theta)$$

Let us assume we are given an θ_{old}, then

$$\mathbb{E}_{p(c|I,\theta_{old})}[\log p(I|\theta)] = \mathbb{E}_{p(c|I,\theta_{old})}[\log p(I, c|\theta)] - \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)]$$

Expectation-Maximization - derivation

Dropping the dependence on x for simplicity for now and using Bayes' rule

$$\log p(I|\theta) = \log p(I, c|\theta) - \log p(c|I, \theta)$$

Let us assume we are given an θ_{old}, then

$$\begin{aligned}
\mathbb{E}_{p(c|I,\theta_{old})}[\log p(I|\theta)] &= \mathbb{E}_{p(c|I,\theta_{old})}[\log p(I, c|\theta)] - \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)] \\
\log p(I|\theta) &= \mathbb{E}_{p(c|I,\theta_{old})}[\log p(I, c|\theta)] - \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)]
\end{aligned}$$

Expectation-Maximization - derivation

Dropping the dependence on x for simplicity for now and using Bayes' rule

$$\log p(I|\theta) = \log p(I, c|\theta) - \log p(c|I, \theta)$$

Let us assume we are given an θ_{old}, then

$$\begin{aligned}
\mathbb{E}_{p(c|I,\theta_{old})}[\log p(I|\theta)] &= \mathbb{E}_{p(c|I,\theta_{old})}[\log p(I, c|\theta)] - \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)] \\
\log p(I|\theta) &= \mathbb{E}_{p(c|I,\theta_{old})}[\log p(I, c|\theta)] - \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)] \\
&= Q(\theta|\theta_{old}) - \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)]
\end{aligned}$$

Expectation-Maximization - derivation

Dropping the dependence on x for simplicity for now and using Bayes' rule

$$\log p(I|\theta) = \log p(I, c|\theta) - \log p(c|I, \theta)$$

Let us assume we are given an θ_{old}, then

$$\begin{aligned}
\mathbb{E}_{p(c|I,\theta_{old})}[\log p(I|\theta)] &= \mathbb{E}_{p(c|I,\theta_{old})}[\log p(I, c|\theta)] - \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)] \\
\log p(I|\theta) &= \mathbb{E}_{p(c|I,\theta_{old})}[\log p(I, c|\theta)] - \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)] \\
&= Q(\theta|\theta_{old}) - \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)]
\end{aligned}$$

One can show that

$$0 \geq \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta_{old})] \geq \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)], \ \forall\theta$$

Expectation-Maximization - derivation

Dropping the dependence on x for simplicity for now and using Bayes' rule

$$\log p(I|\theta) = \log p(I, c|\theta) - \log p(c|I, \theta)$$

Let us assume we are given an θ_{old}, then

$$\begin{aligned}
\mathbb{E}_{p(c|I,\theta_{old})}[\log p(I|\theta)] &= \mathbb{E}_{p(c|I,\theta_{old})}[\log p(I, c|\theta)] - \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)] \\
\log p(I|\theta) &= \mathbb{E}_{p(c|I,\theta_{old})}[\log p(I, c|\theta)] - \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)] \\
&= Q(\theta|\theta_{old}) - \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)]
\end{aligned}$$

One can show that

$$0 \geq \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta_{old})] \geq \mathbb{E}_{p(c|I,\theta_{old})}[\log p(c|I, \theta)], \ \forall\theta$$

This means two things:

- $\log p(I|\theta) \geq Q(\theta|\theta_{old})$
- If we find a θ that increases $Q(\theta|\theta_{old})$ we also increase $\log p(I|\theta)$

Iterative algorithm with two step process

- **E-step:** Compute $p(c|I, \theta_{old})$ given θ_{old}
- **M-step:** Maximize $Q(\theta|\theta_{old})$, i.e. $\max_\theta \mathbb{E}_{p(c|I,\theta_{old})}[\log p(I, c|\theta)]$,
- Set $\theta_{old} = \theta^*$
- Start with a random θ_{old}, iterate E and M steps

For the Gaussian Mixture Model

E-step:

$$
\begin{aligned}
p(c(x)|I(x),\theta_{old}) &= \frac{p(I(x)|c(x),\theta_{old})p(c(x)|\theta_{old})}{p(I|\theta_{old})} \\
&= \frac{\mathcal{N}(I(x)|\mu^{old}_{c(x)},\Sigma^{old}_{c(x)})\pi^{old}_{c(x)}}{\sum_{n=1}^{N}\mathcal{N}(I(x)|\mu^{old}_{n},\Sigma^{old}_{n})\pi^{old}_{n}}
\end{aligned}
$$

For the Gaussian Mixture Model

E-step:

$$
\begin{aligned}
p(c(x)|I(x),\theta_{old}) &= \frac{p(I(x)|c(x),\theta_{old})p(c(x)|\theta_{old})}{p(I|\theta_{old})} \\
&= \frac{\mathcal{N}(I(x)|\mu^{old}_{c(x)},\Sigma^{old}_{c(x)})\pi^{old}_{c(x)}}{\sum_{n=1}^{N}\mathcal{N}(I(x)|\mu^{old}_{n},\Sigma^{old}_{n})\pi^{old}_{n}}
\end{aligned}
$$

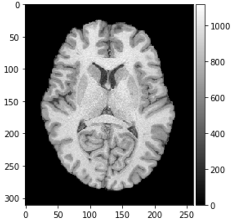
M-step:

$$\pi_n = \frac{\sum_{x\in\Omega} p\left(c(x)=n|I(x),\theta_{old}\right)}{|\Omega|}$$

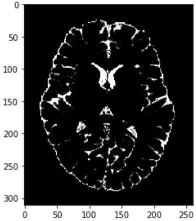
$$\mu_n = \frac{\sum_{x\in\Omega} I(x)p\left(c(x)=n|I(x),\theta_{old}\right)}{\sum_{x\in\Omega} p\left(c(x)=n|I(x),\theta_{old}\right)}$$

$$\Sigma_n = \frac{\sum_{x\in\Omega}\left(I(x)-\mu_n\right)\left(I(x)-\mu_n\right)^T p\left(c(x)=n|I(x),\theta_{old}\right)}{\sum_{x\in\Omega} p\left(c(x)=n|I(x),\theta_{old}\right)}$$

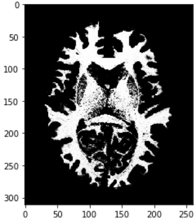
Examples - 3 components



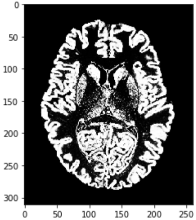
Image



$p(c = 1|I)$

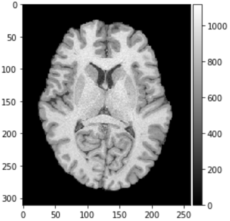


$p(c = 2|I)$

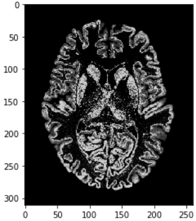


$p(c = 3|I)$

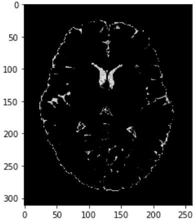
Examples - 5 components



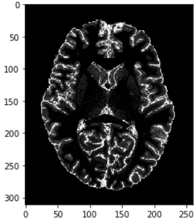
Image



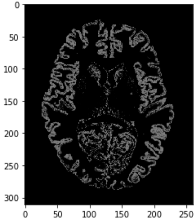
$p(c = 1|I)$



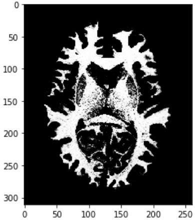
$p(c = 2|I)$



$p(c = 3|I)$



$p(c = 4|I)$

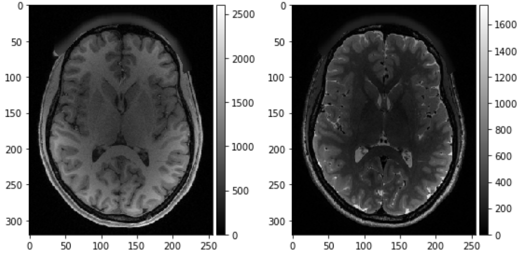


$p(c = 5|I)$

Remarks

- Quite robust method
- Initialization can be important if done badly
- Outputs already useful - gray matter density maps
- Used very regularly - in famous tools such as SPM, FSL and Freesurfer
- Extension to atlas segmentation - next week
- Number of components is important

Simple Segmentation Challenge



These two volumes are in the moodle platform - *Simple Segmentation Challenge*
They are T1-weighted and T2-weighted images of the same individual.

Goals

1. Perform bias removal in both images. Compare the bias fields. Are they different?
2. Perform EM segmentation on individual images and jointly. Compare segmentation results. Is using multiple modalities provide better defined clusters?