EM and mixture models

D.A. Forsyth, UIUC

Example problems

- Examples:
  - Estimate a mean in the presence of outliers
  - Segmentation using a mixture of Gaussians
  - A topic model for words (or image patches)
- Characteristic:
  - missing information which would simplify the problem, if known
    - which data points are outliers
    - which segment each pixel comes from
    - which topic each word comes from

Mean in the presence of outliers

- Observations:
  - either 1D data or 1D outlier
  - data is normal, with known variance
  - mixture model, which is generative
    - i.e. a model of how data is generated

prob of getting data

$$P(\mathbf{x}|\mu,\sigma,\pi) = \pi P_1(\mathbf{x}|\mu,\sigma) + (1-\pi)P_2(\mathbf{x})$$

data

outlier

Loglikelihood

- Unpromising, because a sum of logs of sums

$$\sum_i \log P(\mathbf{x}_i|\mu,\sigma,\pi) = \sum_i \log\left[\pi P_1(\mathbf{x}_i|\mu,\sigma) + (1-\pi)P_2(\mathbf{x}_i)\right]$$

Models of data, outliers

- Data:
  - simplest is normal
- Outliers:
  - simplest is a constant
  - duck issues with normalization for the moment
- doesn't help

$$\sum_i \log P(\mathbf{x}_i|\mu,\sigma,\pi) = \sum_i \log\left[\pi K_1 \exp\frac{-(\mathbf{x}_i-\mu)^2}{2\sigma^2} + (1-\pi)K_2\right]$$

Much easier if...

- We knew where each data item came from
  - Introduce a “switch” d which is 0 if it came from data, 1 if outlier
  - we don’t know these, but pretend we do
  - now the probability of a data item x, d is:

$$P(\mathbf{x}, \delta | \mu, \sigma, \pi) = \left[\pi K_1 \exp \frac{-(\mathbf{x}_i - \mu)^2}{2\sigma^2} \right]^{(1-\delta)} \left[(1-\pi) K_2 \right]^{\delta}$$

Complete data loglikelihood

- Pretend we knew a d for each x
  - the loglikelihood would be

$$\sum_i \log\left(P(\mathbf{x}_i, \delta_i | \mu, \sigma, \pi)\right) = \sum_i \left\{ \begin{array}{c} (1-\delta_i)\left[\log\pi + \log K_1 + \frac{-(\mathbf{x}_i-\mu)^2}{2\sigma^2}\right] + \\ \delta_i\left[\log(1-\pi) + \log K_2\right] \end{array} \right\}$$

- (for reference, expression from previous slide below)

$$P(\mathbf{x}, \delta | \mu, \sigma, \pi) = \left[\pi K_1 \exp\frac{-(\mathbf{x}_i-\mu)^2}{2\sigma^2}\right]^{(1-\delta)} \left[(1-\pi)K_2\right]^{\delta}$$

Complete data loglikelihood

- This is a function of parameters
  - (and all the d, because we don't know them)
  - write as

$$L_c(\theta, \delta)$$

For our example, this has the form:

$$\sum_i \log\left(P(\mathbf{x}_i, \delta_i | \mu, \sigma, \pi)\right) = \sum_i \left\{ \begin{array}{c} (1-\delta_i)\left[\log \pi + \log K_1 + \frac{-(\mathbf{x}_i - \mu)^2}{2\sigma^2}\right] + \\ \delta_i \left[\log(1-\pi) + \log K_2\right] \end{array} \right\}$$

Expectation-Maximization

- Notice:
  - with an estimate of the parameters, can estimate whether x is outlier, noise
    - this could be a soft estimate
    - we could plug this in, then reestimate the parameters
- Formal procedure:
  - parameters are theta
  - start with estimate theta^n
  - form Q function, below (The E-step)
    - maximise in parameters (The M-step)
  - Iterate

$$Q(\theta; \theta^{(n)}) = E_{P(\delta|\mathbf{x},\theta^{(n)})} \left(L_c(\theta, \delta) \right)$$

Expectation-Maximization

- Notice:
  - Q incorporates all we know about d given our current estimate of theta
  - so
    - estimate theta= theta^n
    - now average out d using this estimate
      - but leaving theta alone
      - to get a new function of theta
    - now this contains all we know about d given estimate, so maximize
    - repeat till convergence.

$$Q(\theta;\theta^{(n)}) = E_{P(\delta|\mathbf{x},\theta^{(n)})}\left(L_c(\theta,\delta)\right)$$

In our case ....

$$
\begin{aligned}
E_{P(\delta|\mathbf{x},\theta^{(n)})} L_c(\theta,\delta) &= \sum_{\delta} P(\delta|\mathbf{x},\theta^{(n)}) \log\left(P(\mathbf{x},\delta|\mu,\sigma,\pi)\right) \\
&= \sum_{\delta} P(\delta|\mathbf{x},\theta^{(n)}) \left\{ \sum_{i} \log\left(P(\mathbf{x}_i,\delta_i|\mu,\sigma,\pi)\right) \right\} \\
&= \sum_{i} \left[\begin{array}{c} P(\delta_i = 0|\mathbf{x},\theta^{(n)}) \log P(\mathbf{x}_i,\delta_i = 0|\mu,\sigma,\pi) + \\ P(\delta_i = 1|\mathbf{x},\theta^{(n)}) \log P(\mathbf{x}_i,\delta_i = 1|\mu,\sigma,\pi) \end{array} \right]
\end{aligned}
$$

In our case...

$$
\begin{aligned}
P(\delta_i = 1|\mathbf{x}_i, \theta^{(n)}) &= \frac{P(\mathbf{x}, \delta_i = 1|\theta^{(n)})}{P(\mathbf{x}_i|\theta^{(n)})} \\
&= \frac{P(\mathbf{x}, \delta_i = 1|\theta^{(n)})}{P(\mathbf{x}_i, \delta_i = 0|\theta^{(n)}) + P(\mathbf{x}_i, \delta_i = 1|\theta^{(n)})} \\
&= \frac{P(\mathbf{x}|\delta_i = 1, \theta^{(n)})P(\delta_i = 1|\theta^{(n)}}{\left[\begin{array}{c} P(\mathbf{x}_i|\delta_i = 0, \theta^{(n)})P(\delta_i = 0|\theta^{(n)}) + \\ P(\mathbf{x}_i|\delta_i = 1, \theta^{(n)})P(\delta_i = 1|\theta^{(n)}) \end{array}\right]} \\
&= \frac{K_2(1 - \pi^{(n)})}{K_1 \exp \frac{-(\mathbf{x}_i - \mu^{(n)})^2}{2\sigma^2}\pi^{(n)} + K_2(1 - \pi^{(n)})}
\end{aligned}
$$

Our Q function

$$\sum_i \left[\begin{array}{c} P(\delta_i = 0|\mathbf{x}, \theta^{(n)}) \log P(\mathbf{x}_i, \delta_i = 0|\mu, \sigma, \pi) + \\ P(\delta_i = 1|\mathbf{x}, \theta^{(n)}) \log P(\mathbf{x}_i, \delta_i = 1|\mu, \sigma, \pi) \end{array} \right]$$

Substitute expression below

$$P(\mathbf{x}, \delta|\mu, \sigma, \pi) = \left[\pi K_1 \exp \frac{-(\mathbf{x}_i - \mu)^2}{2\sigma^2} \right]^{(1-\delta)} [(1-\pi)K_2]^{\delta}$$

To get:

$$\sum_i \left[\begin{array}{c} P(\delta_i = 0|\mathbf{x}, \theta^{(n)}) \log \left[\pi K_1 \exp \frac{-(\mathbf{x}_i - \mu)^2}{2\sigma^2} \right] + \\ P(\delta_i = 1|\mathbf{x}, \theta^{(n)}) \log [(1-\pi)K_2] \end{array} \right]$$

In our case, E-step

- Obtain P(d|x, \theta^n) terms
  - often called soft weights
- Substitute to get a Q function that looks like:

$$\sum_i \left[\begin{array}{c} w_i^{(n)} \left[\log \pi + \log K_1 + \frac{-(\mathbf{x}_i - \mu)^2}{2\sigma^2} \right] + \\ (1 - w_i^{(n)}) \left[\log(1 - \pi) + \log K_2 \right] \end{array} \right]$$

Maximizing

- For the m-step, the P(dltheta^n) terms are constants

$$\sum_i \left[\begin{array}{c} w_i^{(n)} \left[\log \pi + \log K_1 + \frac{-(\mathbf{x}_i - \mu)^2}{2\sigma^2} \right] + \\ (1 - w_i^{(n)}) \left[\log(1 - \pi) + \log K_2 \right] \end{array} \right]$$

- Gradient, set to zero

$$\begin{aligned} \pi^{(n+1)} &= \frac{\sum_i w_i^{(n)}}{N} \\ \mu^{(n+1)} &= \frac{\sum_i w_i^{(n)} \mathbf{x}_i}{N} \end{aligned}$$

All together...

- Start with estimate of mean, pi
  - get soft weights from estimate
  - reestimate mean, pi
- until (convergence)

Mixture models and segmentation

- In k-means, we clustered pixels using hard assignments
  - each pixel goes to closest cluster center
  - but this may be a bad idea
    - pixel may help estimate more than one cluster
- We will build a probabilistic mixture model

$$P(\mathbf{x}|\mu_1, \ldots, \mu_k, \pi_1, \ldots, \pi_k, \Sigma) = \sum_i \pi_i P(\mathbf{x}|\mu_i, \Sigma)$$

Mixture model

- Interpretation:
  - obtain pixel by
    - choosing a mixture component
    - given that component, choosing pixel
- Natural to have each mixture component be a Gaussian

$$P(\mathbf{x}|\mu_1, \ldots, \mu_k, \pi_1, \ldots, \pi_k, \Sigma) = \sum_i \pi_i P(\mathbf{x}|\mu_i, \Sigma)$$

Mixture components

- Gaussians
  - are oriented “blobs” in the feature space
  - we will assume covariance is known, and work with mean
  - expression below

$$P(\mathbf{x}|\mu_i, \Sigma) \propto \exp \frac{-(\mathbf{x} - \mu_i)\Sigma^{-1}(\mathbf{x} - \mu_i)}{2}$$

Problem: Learning and IDLL

- We must estimate the mixture weights and means
- Maximising likelihood is very hard
  - in this form, sometimes known as incomplete data log-likelihood

$$\boxed{L(\theta) = \sum_i \log P(\mathbf{x}_i|\theta) = \sum_i \log \left(\sum_j \pi_j P(\mathbf{x}_i|\mu_j, \Sigma) \right)}$$

Complete Data Log-likelihood

- Learning would be easy if we knew which blob each data item came from
  - weights: count what fraction came from which blob
  - means: average data items
- Introduce a set of variables
  - to tell which mixture component a data item came from
  - d_{ij} is 1 if data item i comes from blob j
    - 0 otherwise

$$\mathrm{P}(\mathbf{x}_i|\delta_i,\theta) = \prod_j P(\mathbf{x}_i|\mu_j)^{\delta_{ij}}$$

Complete Data Log-likelihood

- Write the probability for x_i, d_ij conditioned on params

$$
\begin{aligned}
P(\mathbf{x}_i, \delta_i | \theta) &= P(\mathbf{x}_i | \delta_i, \theta) P(\delta_i | \theta) \\
&= (\prod_j [P(x_i | \mu_j) \pi_j]^{\delta_{ij}})
\end{aligned}
$$

Complete Data Log-likelihood

- Log likelihood of x, allocation, conditioned on parameters
- Not obviously helpful
  - but notice the useful form - think of d as switches

$$\begin{aligned}\log P(\mathbf{x}, \delta|\theta) &= \sum_i \log P(\mathbf{x}_i, \delta_i|\theta) \\ &= \sum_{ij} \left[\delta_{ij} \left\{\log P(x_i|\mu_j) + \log \pi_j\right\}\right]\end{aligned}$$

Learning and CDLL

- Introduce hidden variables to get complete data log-likelihood
  - d_{ij} is 1 if data item i comes from blob j
    - 0 otherwise
  - Learning would be easy if we knew which blob each data item came from
    - weights: count what fraction came from which blob
    - means: average data items
  - But we don't

$$L_c(\theta, \delta_{ij}) = \sum_{ij} \delta_{ij} \log \left(\pi_j P(\mathbf{x}_i | \mu_j, \Sigma) \right)$$

Working with CDLL

- Notice:
  - with an estimate of the parameters, can estimate blobs data came from
    - this could be a soft estimate
    - we could plug this in, then reestimate the parameters
- Formal procedure:
  - Quite general
  - start with estimate
  - form Q function, below (The E-step)
    - maximise in parameters (The M-step)
  - Iterate

$$Q(\theta; \theta^{(n)}) = E_{P(\delta|\mathbf{x},\theta^{(n)})} \left(L_c(\theta, \delta) \right)$$

The E-step for Gaussian mixtures

- Notice that the expression for the Q function simplifies

$$
\begin{aligned}
E_{P(\delta|\mathbf{x},\theta^{(n)})}\left(L_c(\theta,\delta)\right) &= E_{P(\delta|\mathbf{x},\theta^{(n)})}\left(\sum_{ij}\delta_{ij}\left[\log\pi_j+\log P(\mathbf{x}_i|\mu_j,\Sigma)\right]\right) \\
&= \left(\sum_{ij}P(\delta_{ij}=1|\mathbf{x},\theta^{(n)})\left[\log\pi_j+\log P(\mathbf{x}_i|\mu_j,\Sigma)\right]\right)
\end{aligned}
$$

The E-step for Gaussian mixtures

- We rearrange using probability identities

$$
\begin{aligned}
P(\delta_{ij} = 1 | \mathbf{x}, \theta^{(n)}) &= \frac{P(\mathbf{x}, \delta_{ij} = 1 | \theta^{(n)})}{P(\mathbf{x} | \theta^{(n)})} \\
&= \frac{P(\mathbf{x} | \delta_{ij} = 1, \theta^{(n)}) P(\delta_{ij} = 1 | \theta^{(n)})}{P(\mathbf{x} | \theta^{(n)})} \\
&= \frac{P(\mathbf{x} | \delta_{ij} = 1, \theta^{(n)}) P(\delta_{ij} = 1 | \theta^{(n)})}{P(\mathbf{x} | \delta_{ij} = 1, \theta^{(n)}) P(\delta_{ij} = 1 | \theta^{(n)}) + P(\mathbf{x}, \delta_{ij} = 0 | \theta^{(n)})}
\end{aligned}
$$

The E step for Gaussian mixtures

- And substitute

$$P(\mathbf{x}_i, \delta_{ij} = 0 | \theta^{(n)}) = \sum_{k \neq j} \pi_k \frac{1}{Z} \exp \frac{-(\mathbf{x}_i - \mu_k^{(n)}) \Sigma^{-1} (\mathbf{x}_i - \mu_k^{(n)})}{2}$$

$$P(\mathbf{x}_i | \delta_{ij} = 1, \theta^{(n)}) = \frac{1}{Z} \exp \frac{-(\mathbf{x}_i - \mu_j^{(n)}) \Sigma^{-1} (\mathbf{x}_i - \mu_j^{(n)})}{2}$$

$$P(\delta_{ij} = 1 | \theta^{(n)}) = \pi_j$$

The M step for Gaussian mixtures

- We must maximise

$$\sum_{ij} P(\delta_{ij} = 1|\mathbf{x}, \theta^{(n)}) \left[\log \pi_j + \log P(\mathbf{x}_i|\mu_j, \Sigma)\right] =$$

$$\sum_{ij} P(\delta_{ij} = 1|\mathbf{x}, \theta^{(n)}) \left[\log \pi_j - \frac{-(\mathbf{x}_i - \mu_j)\Sigma^{-1}(\mathbf{x}_i - \mu_j)}{2} - \log Z\right]$$

- in the mixture weights and in the means
  - we can drop log Z

Two ways

- differentiate, set to zero, etc.
- regard the expectations as "soft counts"
  - so mixture weights from soft counts as:

$$\boxed{\pi_j = \frac{\sum_i P(\delta_{ij} = 1 | \mathbf{x}_i, \theta^{(n)})}{\sum_{i,j} P(\delta_{ij} = 1 | \mathbf{x}_i, \theta^{(n)})}}$$

  - and means from soft counts as:

$$\boxed{\mu_j = \frac{\sum_i \mathbf{x}_i P(\delta_{ij} = 1 | \mathbf{x}_i, \theta^{(n)})}{\sum_{i,j} P(\delta_{ij} = 1 | \mathbf{x}_i, \theta^{(n)})}}$$

Segmentation with EM

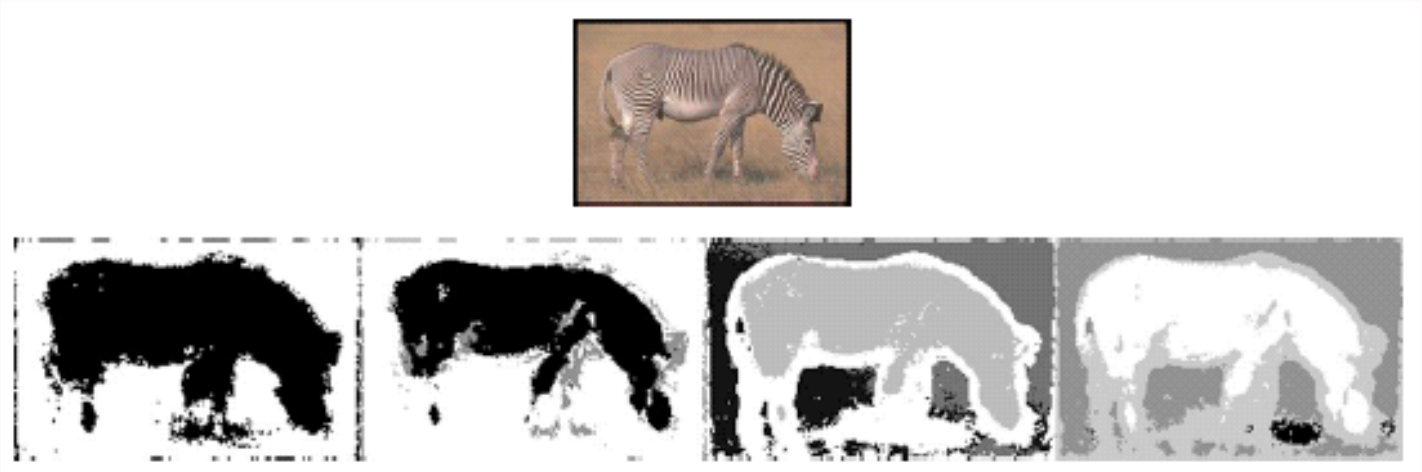

Figure from “Color and Texture Based Image Segmentation Using EM and Its Application to Content Based Image Retrieval”,S.J. Belongie et al., Proc. Int. Conf. Computer Vision, 1998, c1998, IEEE

Another Application

- Clustering binary codes
  - or modelling their source
    - document clustering
    - image clustering
    - detecting objects in attribute lists
- Example: observe ->
  - these bits probably aren't independent
  - but what do we do?
    - two independent sources, mixed

1001	x8
0110	x8
1011	x1
1101	x1
0111	x1
1110	x1

Model