Chapter 6: Generative Probabilistic Models in Multimedia Retrieval

Sami Virpioja
sami.virpioja@tkk.fi

Adaptive Informatics Research Centre
Helsinki University of Technology

February 22, 2008

Outline

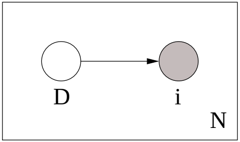
Outline

Classification Tasks in MR

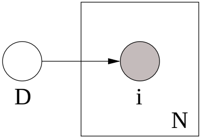
- Many multimedia retrieval tasks can be seen as problems of decision theory:
  - Classification into finite set of classes
    (e.g. motorbike or bicycles or people or car)
  - Detection of objects or events (true or false)
  - Retrieval tasks (relevant or not relevant)
- Probabilistic approach is a very natural one:
  Given a sample x, return the most probable class c

Graphical Notation for Probabilistic Models

- Nodes are variables
  - Observed variables with solid shading
- Edges are dependencies
- Repetition indicated by boxes



Sampling with repeated choice of dice



Sampling from a single dice

Discriminative vs. Generative Classification

- Goal: Find the best document $d \in \{d_1, d_2, \ldots, d_n\}$ given the query q
- Discriminative classification:
  - Choose the most probable document given the query
  - Build a model for $P(D = d_i \mid Q = q)$
  - Typical methods: LDA, SVM, perceptrons
- Generative classification:
  - Choose the document that most likely generated the query
  - Build a model for $P(Q = q \mid D = d_i)$
  - Allows sampling from the joint distribution $P(Q, D)$

Discriminative classification

Generative classification

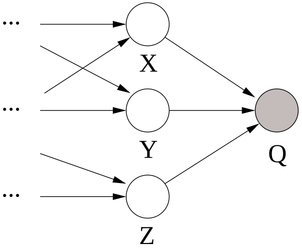
Jigsaw Example

- A set of jigsaw puzzles in their boxes
- One piece is found outside the boxes
- How to select the correct box (without solving all the puzzles)?
- Clue: appereance (color, texture, etc.) of the piece compared to the images of the solved puzzles
- Select the puzzle that maximizes $P(\text{piece comes from the puzzle}) = P(\text{piece} \mid \text{puzzle})$

Generative Models

- Generative models can be thought as sources that generate samples from a probability distribution
  - Jigsaw example: puzzles = sources, pieces = samples
- Limited number of sample types → discrete distribution
- Unlimited sample types → continuous distribution

- Samples are from observed variables; a number of hidden variables may affect them
- Model class $\mathcal{M}$ determines the model structure; parameters $\boldsymbol{\theta}$ differentiate the models within a class



Retrieval with Generative Models

- A set of models of class $\mathcal{M}$ with parameters $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_N$
- An observation O is generated from one of the models
- Models can be ranked by sample likelihood $P(O \mid \boldsymbol{\theta}_i)$
- If observation is a set of samples $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_S\}$ (e.g. words, visual features) and the models are memoryless:

$$P(O \mid \boldsymbol{\theta}_i) = \prod_{j=1}^{S} P(\boldsymbol{v}_j \mid \boldsymbol{\theta}_i)$$

Estimating Model Parameters - ML

- A set of training samples O from a model of class $\mathcal{M}$
- How to estimate parameters $\boldsymbol{\theta}$ for the model?
- Maximum likelihood (ML) estimate
  - Select such $\boldsymbol{\theta}$ that the probability of the training data is maximized
  - Overlearning may be a problem

$$\hat{\boldsymbol{\theta}}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} P(O \mid \boldsymbol{\theta}) \tag{1}$$

Estimating Model Parameters - MAP

- Maximum a posteriori (MAP) estimate
  - Select such $\boldsymbol{\theta}$ that the probability of the model given the training data is maximized
  - Model prior $P(\boldsymbol{\theta})$ is needed

$$\hat{\boldsymbol{\theta}}_{\mathrm{MAP}} = \arg\max_{\boldsymbol{\theta}} P(\boldsymbol{\theta} \mid O) = \arg\max_{\boldsymbol{\theta}} P(\boldsymbol{\theta}) P(O \mid \boldsymbol{\theta}) \qquad (2)$$

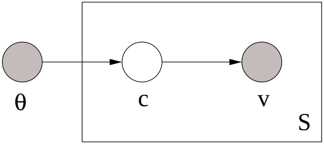
Outline

Mixture Models

- Mixture model is a class of generative models
- We assume that samples $\boldsymbol{v}$ come from a mixture of C independent sources:

$$P(\boldsymbol{v}|\boldsymbol{\theta}) = \sum_{i=1}^{C} P(c_i|\boldsymbol{\theta}) P(\boldsymbol{v}|c_i, \boldsymbol{\theta}) \tag{3}$$

- $\boldsymbol{\theta}$ includes the prior probabilities $P(c_i|\boldsymbol{\theta})$ and parameters of the density functions $P(\boldsymbol{v}|c_i, \boldsymbol{\theta})$ of the sources c_i



Sampling with a mixture model

Gaussian Mixture Models

- If sources generate samples from normal distributions, we have a Gaussian Mixture Model (GMM)
- Normal distribution is useful approximation especially when the observed variable is affected by many independent variables (Central Limit theorem)
- Observed samples come from a set of different objects/categories $\rightarrow$ GMM
  - Direct application: C categories really exist
  - Indirect application: just a tool for get a tractable model
  - By increasing the number of mixtures, we can go arbitrarily close to the any distribution

GMM for Images

- Assume that a image are generated as follows:
  - Take a GMM with parameters $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$
  - For each sample v in the image
    - Pick a random component c_i according to the prior distribution $P(c_i|\boldsymbol{\theta}) = \pi_i$
    - Draw a random sample from c_i according to $N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$
- Probability of a given sample is thus

$$
\begin{aligned}
P(\boldsymbol{v}|\boldsymbol{\theta}) &= \sum_{i=1}^{C} P(c_i|\boldsymbol{\theta}) \times P(\boldsymbol{v}|c_i, \boldsymbol{\theta}) \\
&= \sum_{i=1}^{C} \pi_i \times \frac{1}{\sqrt{(2\pi)^n |\boldsymbol{\Sigma}_i|}} \exp(-\frac{1}{2}(\boldsymbol{v} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i (\boldsymbol{v} - \boldsymbol{\mu}_i))
\end{aligned}
$$

Parameter Estimation for GMMs

- GMM parameters $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$
- Maximum likelihood estimation from a set of samples $O = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_S\}$:

$$\begin{aligned}
\hat{\boldsymbol{\theta}}_{\mathrm{ML}} &= \arg\max_{\boldsymbol{\theta}} P(O \,|\, \boldsymbol{\theta}) \\
&= \arg\max_{\boldsymbol{\theta}} \prod_{j=1}^{S} P(\boldsymbol{v}_j \,|\, \boldsymbol{\theta}) \\
&= \arg\max_{(\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})} \prod_{j=1}^{S} \sum_{i=1}^{C} \pi_i |\boldsymbol{\Sigma}_i|^{-\frac{1}{2}} \exp(-\frac{1}{2}(\boldsymbol{v}_j - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i (\boldsymbol{v}_j - \boldsymbol{\mu}_i))
\end{aligned}$$

EM Algorithm

- No analytical solution for $\boldsymbol{\theta}_{\mathrm{ML}}$
- Expectation-Maximization (EM) algorithm finds a local optimum iteratively
- A hidden variable $\boldsymbol{H}$ gives assignments of samples to components: $h_{ij} = P(c_i \,|\, \boldsymbol{v}_j)$
- Initialize $\boldsymbol{H}$ randomly
- Iterate selecting ML estimates for parameters $\boldsymbol{\theta}$ (M-step) and calculating new expectation values for $\boldsymbol{H}$ (E-step)
- Iterate until likelihood of the training data $P(O \,|\, \boldsymbol{\theta})$ stops increasing

EM Algorithm

- E-step:

$$h_{ij} = \frac{p(\boldsymbol{v}_j|c_i)\pi_i}{\sum_{c=1}^{C} p(\boldsymbol{v}_j|c_c)\pi_c} \tag{4}$$

- M-step:

$$\boldsymbol{\mu}_i = \frac{\sum_j h_{ij}\boldsymbol{v}_j}{\sum_j h_{ij}} \tag{5}$$

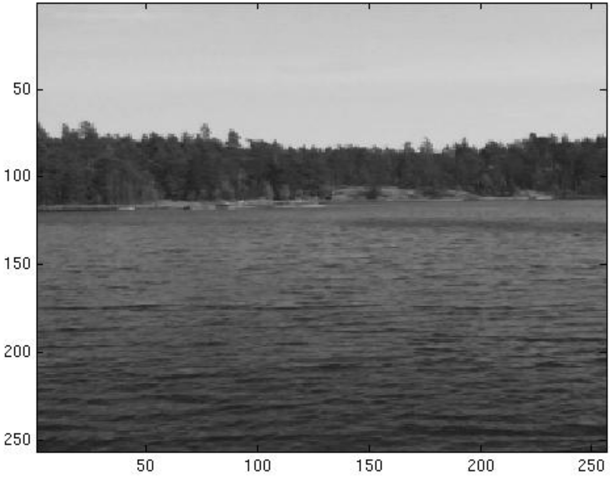
$$\boldsymbol{\Sigma}_i = \frac{\sum_j h_{ij}(\boldsymbol{v}_j - \boldsymbol{\mu}_i)(\boldsymbol{v}_j - \boldsymbol{\mu}_i)^T}{\sum_j h_{ij}} \tag{6}$$

$$\pi_i = \frac{1}{S}\sum_j h_{ij} \tag{7}$$

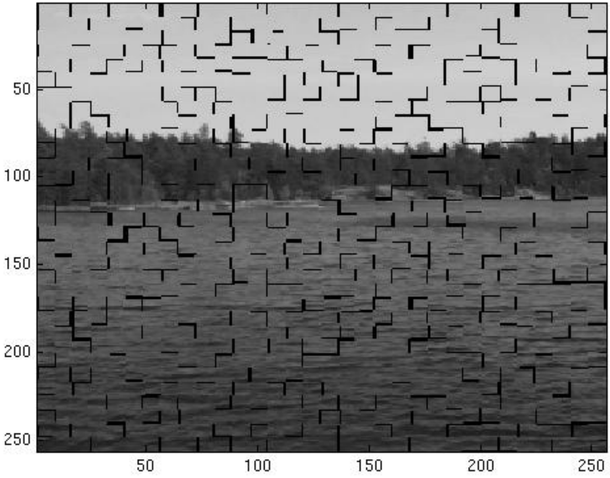
Example: GMM for a Single Image

- 256×256 grayscale image
- Samples: 8×8 blocks uniformly from the image
- Features: (x, y) coordinates of the middle point, DCT transform of the pixel intensities

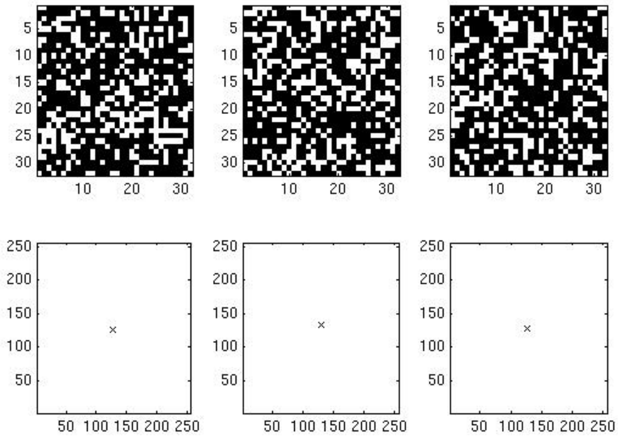
Example: GMM for a Single Image



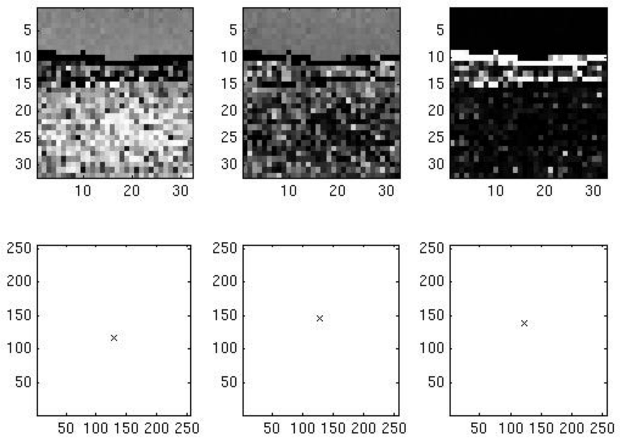
Example: GMM for a Single Image



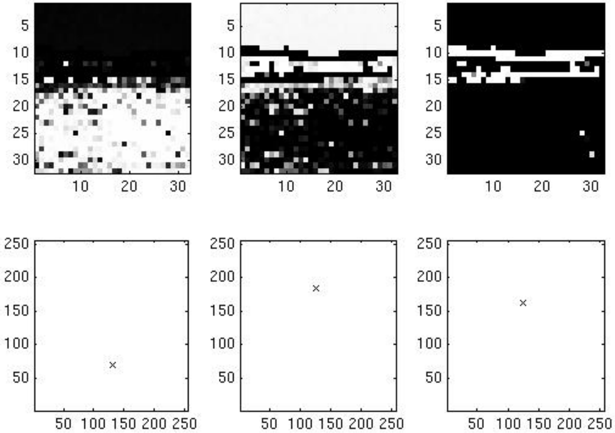
Example: GMM for a Single Image



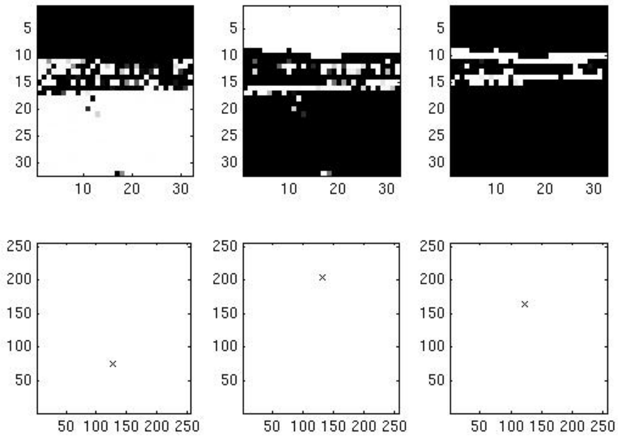
Example: GMM for a Single Image



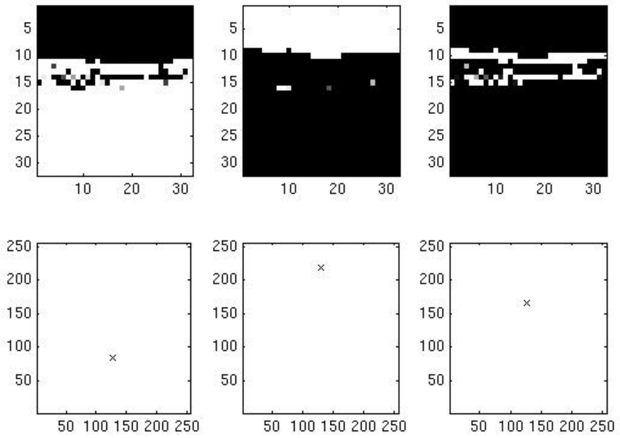
Example: GMM for a Single Image



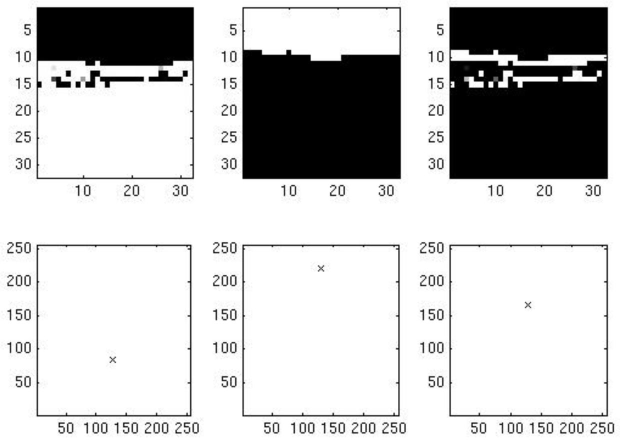
Example: GMM for a Single Image



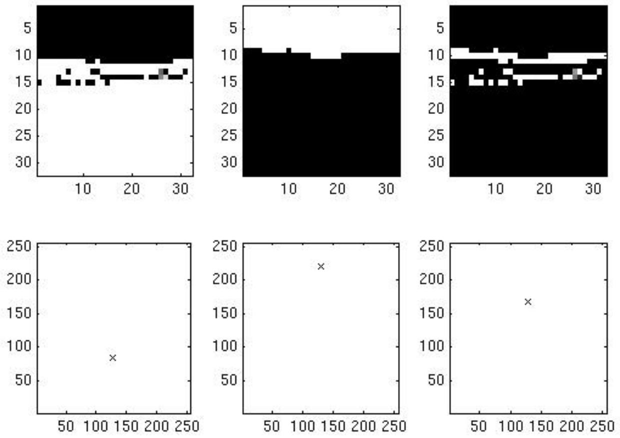
Example: GMM for a Single Image



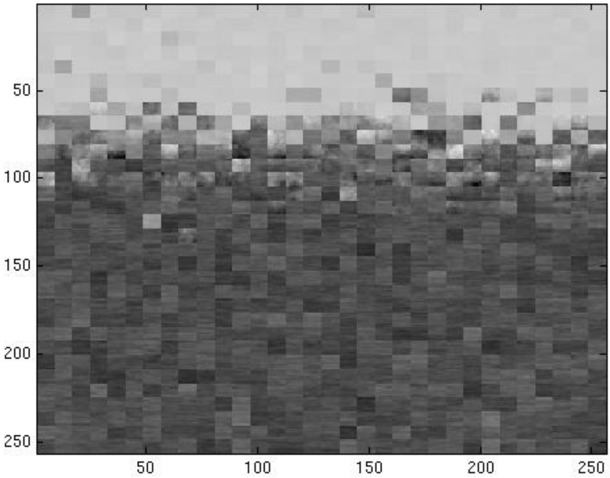
Example: GMM for a Single Image



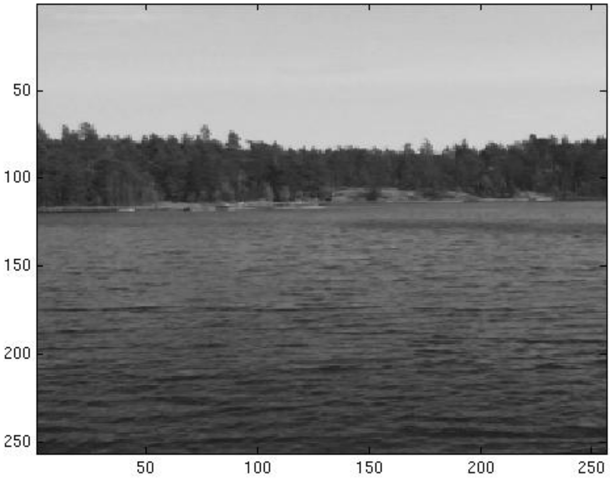
Example: GMM for a Single Image



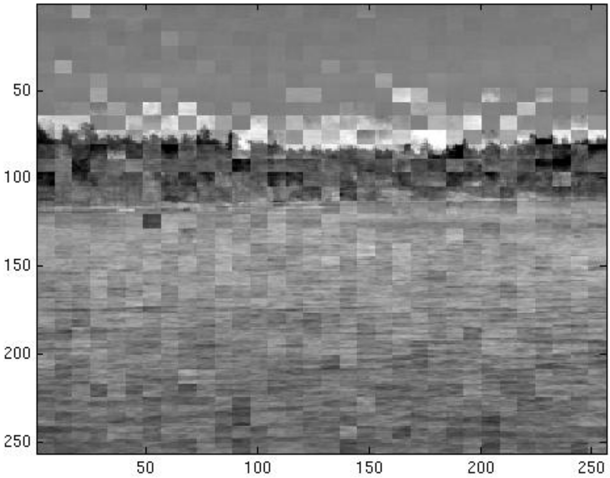
Example: GMM for a Single Image



Example: GMM for a Single Image



Example: GMM for a Single Image



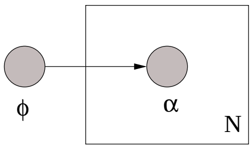
Outline

Statistical Language Models

- Statistical language model gives a probability distribution $P(S)$ over texts strings S of a language
- Usually based on words: $P(w_1, w_2, \ldots, w_N)$
- N-gram assumption:
  $P(w_i \mid w_1, \ldots, w_{i-2}, w_{i-1}) = P(w_i \mid w_{i-n+1}, \ldots, w_{i-2}, w_{i-1})$
- n-gram model is a $(n-1)$:th order Markov model
- High-order n-gram models are needed in speech recognition and statistical machine translation
- Unigram (1-gram) model seems to be enough for information retrieval

Unigram Model for Text Retrieval

- A lexicon of T words, α_i is the i:th word in lexicon
- Model parameters $\boldsymbol{\phi} = (\phi_1, \phi_2, \ldots, \phi_T)$
- $P(w = \alpha_i | \boldsymbol{\phi}) = \phi_i$
- Generation process: For each term of the document, draw a random term from $\boldsymbol{\phi}$ according to multinomial distribution $\mathrm{mult}(\boldsymbol{\phi})$



Sampling with a unigram model

Sample Likelihood for Unigram Model

- Sampled document is denoted as $\boldsymbol{t} = (t_1, t_2, \ldots, t_T)$, where t_i is the number of occurrences of α_i
- Probability of $\boldsymbol{t}$ given the model $\boldsymbol{\phi}$ is

$$P(\boldsymbol{t} \mid \boldsymbol{\phi}) = \frac{(\sum_{i=1}^{T} t_i)!}{\prod_{i=1}^{T} t_i!} \prod_{i=1}^{T} \phi_i^{t_i} \tag{8}$$

- $\prod_{i=1}^{T} \phi_i^{t_i}$ gives the probability of words in one order
- The messy first term gives the number of permutations for the set of $N = \sum_{i=1}^{T} t_i$ words (bag-of-words model)

ML Estimate for Unigram Parameters

- Let's find the ML estimate for the parameters:

$$
\begin{aligned}
\hat{\phi}_{\mathrm{ML}} &= \arg\max_{\boldsymbol{\phi}} P(\boldsymbol{t} \mid \boldsymbol{\phi}) \\
&= \arg\max_{\boldsymbol{\phi}} \frac{(\sum_{i=1}^{T} t_i)!}{\prod_{i=1}^{T} t_i!} \prod_{i=1}^{T} \phi_i^{t_i} \\
&= \arg\max_{\boldsymbol{\phi}} \prod_{i=1}^{T} \phi_i^{t_i} \\
&= \arg\min_{\boldsymbol{\phi}} \left[-\sum_{i=1}^{T} t_i \log \phi_i \right]
\end{aligned}
$$

Unigram Parameters

- $-\sum_{i=1}^{T} t_i \log \phi_i$ is (empirical) cross-entropy of the source that outputs words according to distribution ϕ
- Minimized with $\phi_i = \frac{t_i}{\sum_i t_i} = \frac{t_i}{N}$
- Thus ML estimates are just relative frequencies of words in the document

Smoothing and Interpolation

- However, the ML estimate has two fundamental problems
  - Words that did not occur in the document will have zero probabilities
  - Words that occurred only few times will have overestimated probabilities
- Traditional solutions:
  - Smoothing explicitly moves probability mass from the low frequencies to the zero frequencies (more or less heuristically)
  - Interpolation or back-off with/to more general (e.g. lower order) models
- Here we discuss only interpolation

Linear Interpolation with a Background Model

- Make a background model ϕ_{BG} using all the documents as training data
- Then interpolate linearly with a coefficient λ:

$$\phi_i = \lambda\phi_{i\mathrm{ML}} + (1-\lambda)\phi_{i\mathrm{BG}} \tag{9}$$

- If $t_{d,i}$ is the number of times α_i occurred in document d, $\phi_{i\mathrm{BG}} = \sum_d t_{d,i} / \sum_d \sum_j t_{d,j}$
- Or with document frequencies: $\phi_{i\mathrm{BG}} = df_i / \sum_j df_j$

Relation to Weighting with *tf.idf*

- Let's try this in text retrieval for a query $\boldsymbol{t}$
- With some of manipulation (see p. 192 in book), we get

$$P(\boldsymbol{t} \mid \boldsymbol{\phi}) \propto \prod_{t_i > 0} [\frac{\lambda \phi_{i\mathrm{ML}}}{(1-\lambda)\phi_{i\mathrm{BG}}} + 1]^{t_i} \tag{10}$$

- Background probabilities ϕ_{BG} weight down words that are common in the collection, as the idf component does in term weighting

Hierarchical Interpolation

- Even more models can be combined with linear interpolation
- E.g. for a video subdivided into scenes which are subdivided into shots,

$$\phi_i = \lambda_{\text{Shot}} \phi_{i\text{Shot}} + \lambda_{\text{Scene}} \phi_{i\text{Scene}} + \lambda_{\text{Coll}} \phi_{i\text{Coll}} \tag{11}$$

  where $\lambda_{\text{Shot}} + \lambda_{\text{Scene}} + \lambda_{\text{Coll}} = 1$
- λ:s can be estimated from a separate development set
- Interpolation can be used as well for generative image or video models

Outline

Combining Multiple Modalities

- Generative models for different modalities can naturally be combined
- E.g. multimedia query $\boldsymbol{q} = (\boldsymbol{v}, \boldsymbol{t})$ with visual part $\boldsymbol{v}$ and textual part $\boldsymbol{t}$
- Rank documents by joint probability $P(\boldsymbol{q} \mid d) = P(\boldsymbol{v}, \boldsymbol{t} \mid d)$
- Assuming that $\boldsymbol{v}$ and $\boldsymbol{t}$ are independent $P(\boldsymbol{q} \mid d) = P(\boldsymbol{v} \mid \boldsymbol{\theta}_d) P(\boldsymbol{t} \mid \boldsymbol{\phi}_d)$
- The independence assumption is of course totally unrealistic
- However, may work well enough in practice

Outline

Conclusions

- In IR, generative models describe the process of generating samples from documents
- Given the query, documents can be ranked with the sample likelihood of their models
- Two examples were given:
  - Gaussian Mixture Model for images
    - EM algorithm for training
  - Unigram model for text documents
    - Direct ML estimates, but smoothing/interpolation required