Fast search for Dirichlet process mixture model

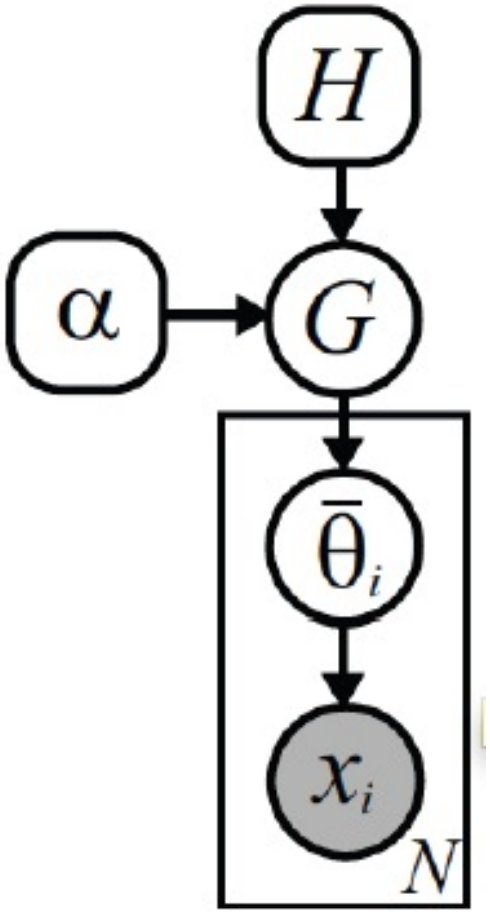
Hal Daume III

Presented by Yun Zhang

Oct 13 2011

Dirichlet Process Mixture Model

$$
\begin{aligned}
G \mid \alpha, G_0 &\sim \mathcal{DP}(\alpha, G_0) \\
\theta_n \mid G &\sim G \\
x_n \mid \theta_n &\sim F(\theta_n)
\end{aligned}
$$



Existing Algorithms

- Sampling methods – MCMC
  - Produce a true representation of the posterior
  - Convergence can be difficult to diagnose

- Variational techniques
  - Deterministic
  - Produce an approximation to the true posterior

  - Both of them run not very fast

Motivation

- Learning posteriors is expensive
- In some cases of DPMM, we only need to know about an approximate MAP cluster assignment for each observation, which lead to a limited search space
- We may need an efficient way of finding a high probability region as initial settings of MCMC

Problem setup

- Input: $\mathbf{x}_{1:N}$

- Output: $\mathbf{c}_{1:N}$, where:

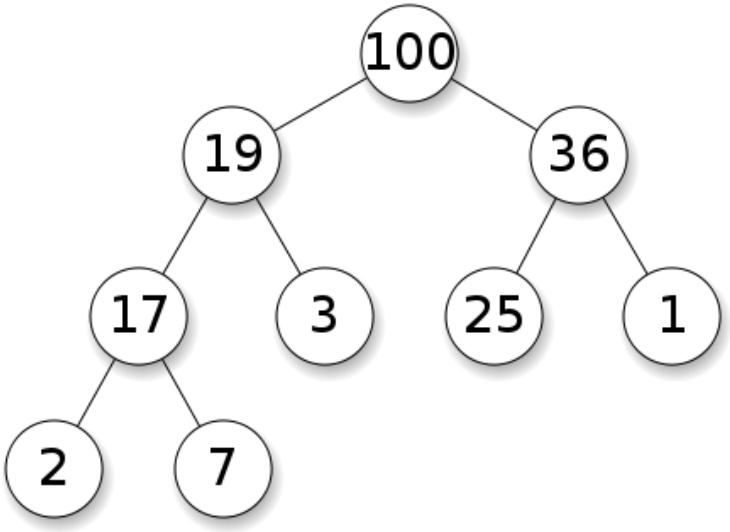
$$\mathbf{c} = \text{argmax } p(\mathbf{c},\mathbf{x})$$

A* Search Algorithm

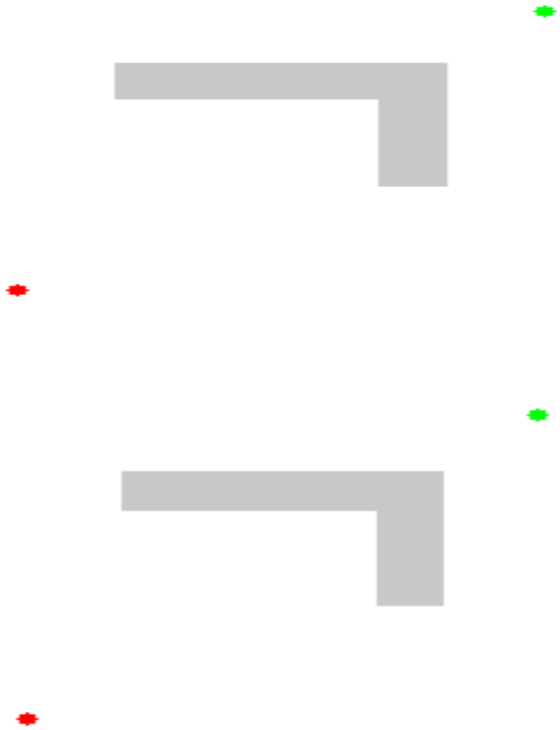
- A greedy search algorithm which uses a distance-plus-cost heuristic function to determine the order of nodes to visit
  - $g(x) = d(x) + h(x)$
- An extension of Dijikstra's algorithm
- One efficient implementation uses a heap (max-queue) to keep track of current searching progress

A* Search Algorithm

A max-heap



A* v.s. Dijikstra's



Guarantees of global optima

- The heuristic function is “admissible”
  - The estimated cost must always be lower than or equal to the actual cost of reaching the goal state

- Heap size is unlimited
  - Search paths are never cut off

DP Search

- g: distance-plus-cost function
- b: heap size

function DPSearch
input: a scoring function g, beam size b, data $x_{1:N}$
output: a clustering c

initialize max-queue: $Q \leftarrow [\langle\rangle]$
while Q is not empty **do**
  remove state $c_{1:N^0}$ from the front of Q
  if $N^0 = N$ **then return** c
  for all clusters d in c and a new cluster **do**
    let $c^0 = c \oplus \langle d \rangle$
    compute the score $s = g(c^0, x)$
    update queue: $Q \leftarrow \text{Enqueue}(Q, c^0, s)$
  end for
  if $b < \infty$ and $|Q| > b$ **then**
    Shrink queue: $Q \leftarrow Q_{1:b}$
      (drop lowest-scoring elements)
  end if
end while

Scoring Function

$$
\begin{aligned}
g(\mathbf{c}^0, \mathbf{x}) &\geq \max p(\mathbf{c}, \mathbf{x}) \\
g &= \max p(\mathbf{c}) p(\mathbf{x}|\mathbf{c}) \\
&= \max p(\mathbf{c}) p(\mathbf{x}|\mathbf{c}) \\
&= \max p(\mathbf{c}) g_{Trivial}(\mathbf{x}|\mathbf{c}^0) heu(\mathbf{x}|\mathbf{c}^0)
\end{aligned}
$$

- Scoring function g() is an estimation of the posterior probability p(c, x)

- Our goal here is to find out a setting of cluster assignments c that can maximize the posterior probability

Maximize the prior p(c)

$$P(\boldsymbol{m} \mid \alpha, N) = \frac{N!}{\alpha^{(N)}} \frac{\alpha^{\sum_{i=1}^{I} m_i}}{\prod_{i=1}^{I} i^{m_i}(m_i!)}$$

- Coming from Chinese restaurant process

Trivial Scoring Function

$$g_{\text{Trivial}}(\boldsymbol{x} \mid \boldsymbol{c}^0) \triangleq \prod_{k \in \boldsymbol{c}^0} H(\boldsymbol{x}_{\boldsymbol{c}^0 = k})$$

- heuristic function is zero in log space
- A* becomes a Dijikstra's algorithm, which lead to an inefficient search

Admissible Function

$$g_{\text{Trivial}}(x \mid c^0) \tag{11}$$

$$\prod_{n=N^0+1}^{N} \max_{1 \le k \le K^0+1} \max_{\substack{c: \\ c \restriction N^0 = c^0 \\ c_n = k}} H(x_n \mid x_{c_{1:n-1}=c_m})$$

- Treat unclustered data point independently
- For each unclustered data point x, choose most like cluster label k for x, then cluster remaining points as to only whether they fall into k or not
- Can be considered as admissible when a “replica” trick is used

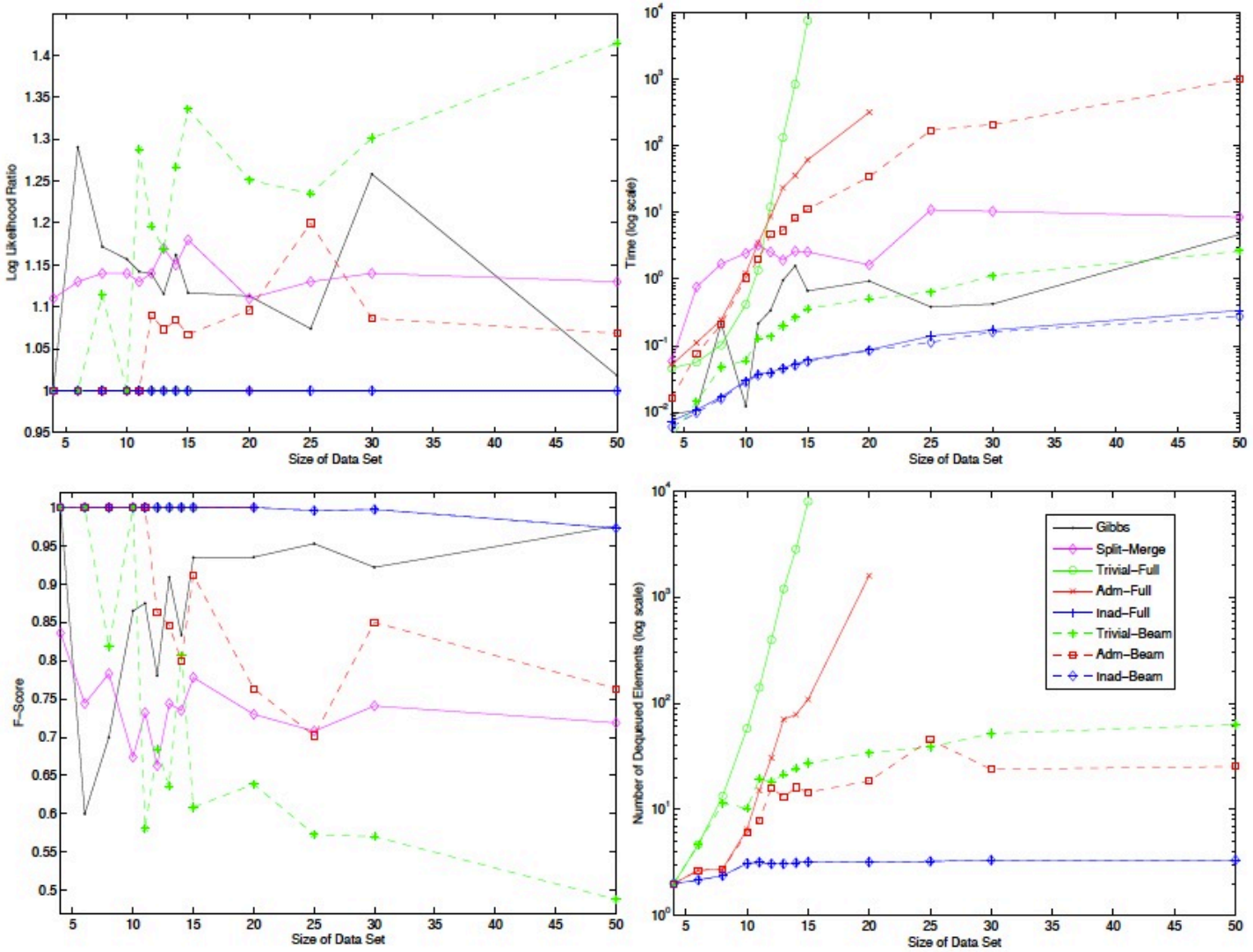
Inadmissible Heuristic Function

$$g_{\text{Inad}}(\boldsymbol{x} \mid \boldsymbol{c}^0) \triangleq g_{\text{Trivial}}(\boldsymbol{x} \mid \boldsymbol{c}^0) \prod_{n=N^0+1}^{N} H(x_n)$$

- Use the marginal likelihood as heuristic, which means for each unclustered data point, assign them a new cluster number

- No longer overestimate the posterior probability, therefore not admissible

Experiment 1: Artificial data (DPGMM)



Experiment 2: Handwritten data

- Dirichlet/Multinomial setting

- Conclusion is our search algorithm runs much faster than Gibbs sampler, and gives better MAP estimation of cluster assignments.

Experiment 3: NIPS documents

- Still Dirichlet/Multinomial setting

- Conclusion is our search algorithm runs much much faster than Gibbs sampler

Advantages & Limitations

- Fast to find a MAP cluster assignment for each data point
- Cannot represent the true posterior
- Applies only to exponential families with conjugate prior (or at least speed will be slow down when apply to non-conjugate distributions)