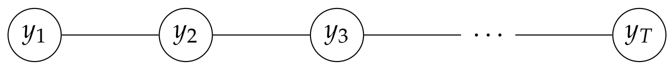
CS281: Advanced ML **October 11, 2017**

Lecture 11: Belief propagation

Lecturer: Sasha Rush *Scribes: Ismail Ben Atitallah, Hao Wu, Ziliang Che, Jiaoyang Huang*

We've seen undirected graph models, like



There are many algorithms to do inferences for undirected graph models

- Forward–backward
- Sum product algorithm
- Mean field
- Belief propagation
- Gibbs sampling
- MAP inference.

11.1 Time series

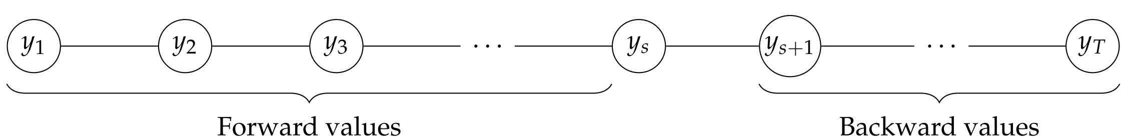
We denote the number of classes per node by V. The joint probability distribution of time series is given by

$$\begin{aligned} p(y_{1:T}) &= \exp\{\sum_t \theta_t^T(y_{t-1}, y_t) + \theta_t^o(y_t) - A(\theta)\} \\ &\propto \prod_t \psi_t(y_{t-1}, y_t)\psi_t(y_t), \end{aligned}$$

where $\psi_t(\cdot,\cdot)$ is a binary function, and $\psi_t(\cdot)$ is a unary function. The marginal distribution at y_s is given by

$$\begin{aligned} p(y_s = v) &= \sum_{\substack{y'_{1:T} \\ y'_s = v}} \prod_t \psi(y'_{t-1}, y'_t)\psi(y'_t)/Z(\theta) \\ &= Z(\theta)^{-1} \sum_{y'_T} \psi_T(y'_T) \sum_{y'_{T-1}} \psi_{T-1}(y'_{T-1})\psi_T(y'_{T-1}, y'_T) \cdots \sum_{y'_2} \psi_2(y'_2)\psi_3(y'_2, y'_3) \sum_{y'_1} \psi_1(y'_1)\psi_2(y'_1, y'_2) \end{aligned}$$

The sum can be performed in two directions, forwardly from y_1, y_2 till y_{s-1}, and backwardly from y_T, y_{T-1} till y_{s+1}.



It takes $O(TV^2)$ time to compute all margins with dynamic programming, i.e. $p(y_s = v)$ for all s, v.

11.2 Sum-product of Time Series

We can rewrite the above forward and backward computations in a fancier way. We define forward propagation:

$$\underbrace{\mathrm{bel}_t^-(y_t)}_{\text{forward belief}} \propto \psi_t(y_t)\underbrace{m_{t-1\to t}^-(y_t)}_{\text{message}},$$

where the message is

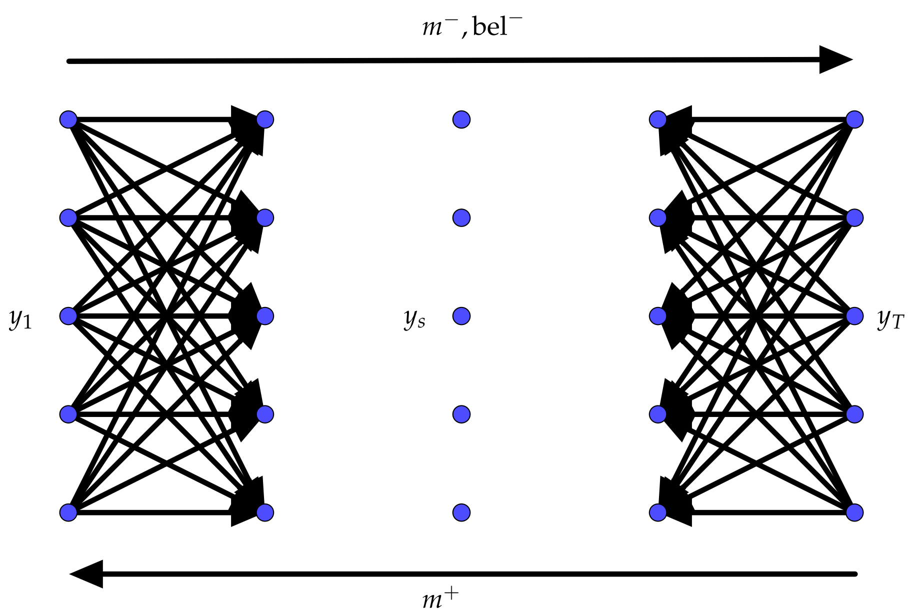
$$m_{t-1\to t}^-(y_t) = \sum_{y_{t-1}} \psi_t(y_{t-1}, y_t)\mathrm{bel}_{t-1}^-(y_{t-1}).$$

Similarly we define the backward propagation:

$$m_{t+1\to t}^+(y_t) = \sum_{y_{t+1}} \psi_{t+1}(y_t, y_{t+1})\psi_{t+1}(y_{t+1})m_{t+2\to t+1}^+(y_{t+1}).$$

Then the marginal probability is given by

$$p(y_t) = \mathrm{bel}_t(y_t) \propto \underbrace{m_{t+1\to t}^+(y_t)}_{\text{backward}}\underbrace{\mathrm{bel}_t^-(y_t)}_{\text{forward}}.$$



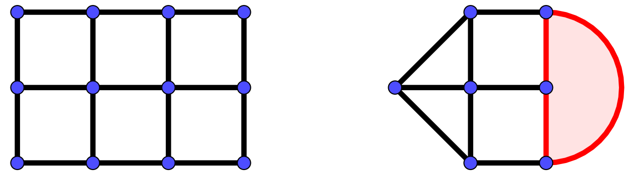
To compute all the marginal probability using the above algorithm, it takes $O(TV^2)$ time and $O(TV)$ memory to store all the forward and backward values, i.e. $m_{t+1\to t}^+(y_t)$ and $\mathrm{bel}_t^-(y_t)$.

11.3 Belief Propagation of General Graphs

We have the belief propagation for time series, it is natural to ask about undirected graph model defined by a general graph,

$$p(y_s = v) = \sum_{\substack{y'_{1:T} \\ y'_s = v}} \prod_C \psi_C(y'_C).$$

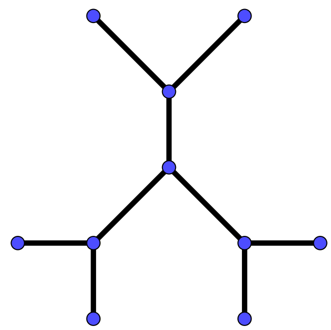
The above summation might be hard. For example, if C runs through cliques of size 5, the sum is over V^5 terms. Even if we start with a graph with maximum clique size ≤ 2, the Ising model on $2d$ lattice, we will end up with larger cliques, i.e. cliques of size 3.



The minimum size of max clique induced -1 is the *treewidth* of a graph. Calculating the treewidth of a graph is NP hard, and can be reduced to 3-SAT.

11.3.1 Belief Propagation on graphs with treewidth $= 1$

We derive the generalization of forward-backward sum product algorithm on trees.



For a tree graph, we can pick any vertex as a root, then for any vertex x_s the parent nodes set $pa(x_s)$ and children nodes set $ch(x_s)$ are well defined. The belief propagation consists of two parts: upward pass and downward pass. The upward pass is defined by

$$\begin{aligned} m^-_{s\to t}(x_t) &= \sum_{x_s} \psi_{s,t}(x_s, x_t)\mathrm{bel}^-_s(x_s), \\ \mathrm{bel}^-_t(x_t) &\propto \psi_t(x_t) \prod_{s\in ch(t)} m^-_{s\to t}(x_t). \end{aligned}$$

The downward pass is defined by

$$\begin{aligned} \mathrm{bel}_s(x_s) &\propto \mathrm{bel}^-_s(x_s) \prod_{t\in pa(s)} m^+_{t\to s}(x_s), \\ m^+_{t\to s}(x_s) &= \sum_{x_t} \psi_{s,t}(x_s, x_t)\psi_t(x_t) \prod_{\substack{c\in ch(t), \\ c\neq s}} m^-_{c\to t}(x_t) \prod_{p\in pa(t)} m^+_{p\to t}(x_t). \end{aligned}$$

To compute all the marginal probability using the above belief propagation, it takes $O(TV^2)$ time and $O(TV)$ memory.

11.4 Some Remarks

1. Gaussian Belief Propagation for Gaussian directed models is the same as Kalman Filter.

2. The above algorithms follow serial protocol for sum-product, i.e. we sequentially compute the forward and backward values. A variant of these algorithms uses parallel protocol, where believes are sent to neighbor nodes at the same time.

$$\text{bel}_s(x_s) \propto \psi_s(x_s) \prod_{t \in \text{nbr}(s)} m_{t \to s}(x_s),$$
$$m_{s \to t}(x_t) = \sum_{x_s} \psi_s(x_s) \psi_{s,t}(x_s, x_t) \prod_{\substack{u \in \text{nbr}(s) \\ u \neq t}} (x_s).$$

3. For the sum-product algorithm, one needs the distributive property of $+, \times$. In general, an abstract structure with distributive property is called commutative semi-ring. The sum-product algorithm gives us the marginal distribution. If we replace $+$ by max, the same algorithm gives us the argmax assignment. If we replace $+$ by $\vee$ and $\times$ by $\wedge$, the same algorithm gives us the satisfying assignment.