Probabilistic Graphical Models Project 1: Belief Propogation

Matthew Caulfield

ECSE-6810

October 25, 2020 (Extension)

1 Introduction

There are many babies born with congenital heart disease and it is critical that they are diagnosed with this disease as soon as possible. With the use of approximate inference on the CHILD Bayesian network shown in figure one Doctors can receive a more accurate diagnosis of congenital heart disease in newborns. The CHILD network combines both clinical expertise and available data to improve the accuracy of the diagnoses. In this project we try to aid a doctor find the probability a baby has birth asphyxia given that the baby was reported to have CO_2 less then 7.5, left ventricular hypertrophy, and a lung X-ray that was reported to be plethoric. We also try to aid a doctor in the diagnosis of a disease given the same parameters.

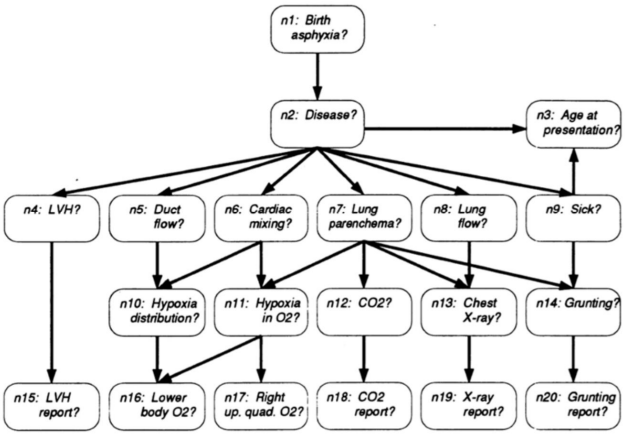

Figure 1: CHILD Network

The CHILD network is a large and complex network. There are 20 nodes in the network connected by 25 arcs. Each node can have either 2, 3, 4, 5, or 6 states. Due to the complexity of the CHILD network variable elimination inference and other exact inferences will not produce results on this network. Instead approximate inference must be used. In this project three forms of approximate inference were used: likelihood weighted sampling, Gibbs sampling, and the mean field method. These methods each have their own advantages and disadvantages, which will be elaborated on in the theory section. With the use of three different approximate inference algorithms, we can get a better understanding of the likely disease and true probability of birth asphyxia.

2 Theory

2.1 Likelihood Weighted Sampling

Likelihood weighted sampling performs approximate inference on a Bayesian network by creating samples that are weighted. The weighted samples are then used to approximate the distribution of the Bayesian network for the given evidence. Figure 2 shows the algorithm for likelihood weighted sampling from Professor Ji's text book *Probabilistic Graphical Models for Computer Vision*[1].

▷ **E**: evidence nodes
Order BN variables $X_1, X_2, \ldots, X_N$ according to their topological order from the root nodes until leaf nodes
Initialize weights $w_1, w_2, \ldots, w_T$ to 1
for t=1 to T **do** t: index to the number of samples
  for n=1 to N **do** n: index to the node number
    if $X_n^t \notin$ **E then**
      sample x_n^t from $p(X_n^t|\pi(X_n^t))$
    else
      $x_n^t = e_n$
      $w_t = w_t * p(X_n^t = e_n|\pi(X_n^t))$
    end if
  end for
  Form sample $\mathbf{x}^t = \{x_1^t, x_2^t, \ldots, x_N^t\}$ and compute its weight w_t
end for
Return $(\mathbf{x}^1, w_1), (\mathbf{x}^2, w_2), \ldots, (\mathbf{x}^T, w_T)$

Figure 2: Algorithim for likelihood weighted sampling

As the algorithm describes, the nodes of the network are topographically ordered from the root nodes to the leaf nodes. Once the nodes are ordered each sample is created by traversing the ordered nodes. The state of each node that is not in the evidence is sampled from its probability distribution given its parents current samples' state. If the node is an evidence node then for this sample the state of the node is set to the state from the evidence and the weight for this node is multiplied by the probability of the evidence given the configuration of its parents current sample state. This is done for a predetermined amount of samples. To find the probability distribution function of a node given evidence from the weighted sample equation one is used. Where y_k is the kth state of the node for which the probability distribution is desired, y_m is the state of node y in the mth sample, and w_m is the mth samples weight

$$P_D(y_k|\mathbf{e}) = \frac{\sum_{m=1}^{M} w_m I[y_m = y_k]}{\sum_{m=1}^{M} w_m} \tag{1}$$

Because the network traverses the nodes and creates the current nodes sample based on its parents configuration in the same sample the order of the network matters. For each sample parents must be sampled before their children. The strengths of this algorithm are that it is simple to implement, as the number of samples taken increases the distribution of the samples asymptotically approaches the true distribution, and is more efficient then likelihood sampling because samples do not need to be thrown out. However it also has some drawbacks like it is limited to discrete Bayesian networks, is inefficient for inference with evidence far from the root nodes, and has bias due to the weights of each sample. Because the CHILD network has evidence on its leaves this algorithm is not efficient for this network. Bias is also observed when implementing on the CHILD network because of the weights assigned to each sample.

2.2 Gibbs sampling

Gibbs sampling is a form of Markov Chain Monte Carlo sampling implemented to do inference on a Bayesian network. The samples created follow a Markov chain where the Bayesian network is a an ergodic Markov chain. After a burn-in period the samples will follow the underlying distribution of the chain. Gibbs sampling can be thought of as a random walk through the Bayesian Network and constructs a Markov chain of samples where each sample differs from the previous sample by one node changing state. To find the state of the changed node the sample is taken from the probability distribution of the node given its Markov blanket from the previous sample. Equation two is used to find the probability distribution of a node given its Markov blanket. Where Y_k is the kth child of X_n.

$$p(X_n|MB(X_n)) = \frac{p(X_n|\pi(X_n)) \prod_{k=1}^{K} p(Y_k|\pi(Y_k))}{\sum_{x_n} p(x_n|\pi(X_n)) \prod_{k=1}^{K} p(Y_k|\pi(Y_k))} \tag{2}$$

To avoid correlation between the samples, a number of samples are skipped between each recorded sample. Figure 3 contains the algorithm for Gibbs sampling[1].

▷ **E**: evidence nodes
Order BN variables $X_1, X_2, \ldots, X_N$ according to their topological order from the root nodes until leaf nodes
Initialize weights $w_1, w_2, \ldots, w_T$ to 1
for t=1 to T **do** t: index to the number of samples
  for n=1 to N **do** n: index to the node number
    if $X_n^t \notin$ **E then**
      sample x_n^t from $p(X_n^t | \pi(X_n^t))$
    else
      $x_n^t = e_n$
      $w_t = w_t * p(X_n^t = e_n | \pi(X_n^t))$
    end if
  end for
  Form sample $\mathbf{x}^t = \{x_1^t, x_2^t, \ldots, x_N^t\}$ and compute its weight w_t
end for
Return $(\mathbf{x}^1, w_1), (\mathbf{x}^2, w_2), \ldots, (\mathbf{x}^T, w_T)$

Figure 3: Algorithm for Gibbs sampling

Three parameters must be tuned before using Gibbs sampling for approximate inference: the length of the burn-in period, the initial state of each node, and the number of samples to skip between recorded samples. To tune the length of the burn-in period Gibbs sampling was run with 150 different burn-in rates and a burn-in rate was decided based on when the probability of the baby having birth asphyxia converged. Figure 4 is a graph of the probability of birth asphyxia given the evidence for each burn in rate.

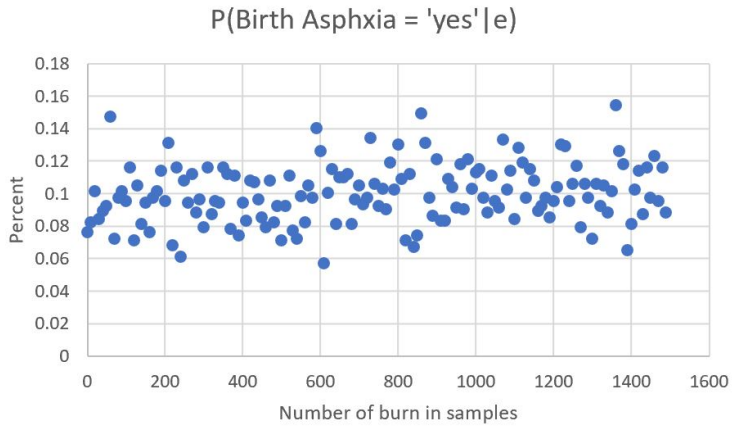

Figure 4: Probability of Birth Asphyxia given evidence for different burn-in rates

Based on this graph a burn-in rate of 1000 was decided. Once a burn-rate was decided the number of samples to skip was tuned. Another 150 chains were created each with a different skip rate. The skip rate was also decided based on when the probability of the baby having birth asphyxia converged. A graph of each trial is shown in figure 5.

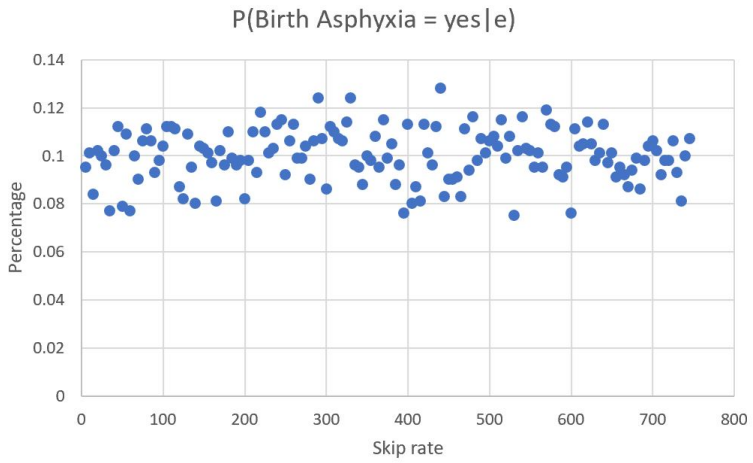

Figure 5: Probability of Birth Asphyxia given evidence for different skip rates

Each sample does not seem correlated to the last because skip rate did not have that much of an effect on the probability, To be conservative a skip rate of 10 was chosen. Because the length of burn-in is sufficiently long any initialization will work.

2.3 Mean Field

The mean field algorithm is a variational inference approach to approximate inference. In this approach a surrogate distribution $q(\mathbf{X}, \beta)$ is found as an approximation of $p(\mathbf{X}|\mathbf{E})$. Beta is found so the Kullback-Leivler divergence of q and p, $KL(q(\mathbf{X}, \beta)||p(\mathbf{X}|\mathbf{E}))$, is minimized. For a discrete Bayesian network $q(x_i, \beta_i)$ can be found using equation 3. $q(\mathbf{X}, \beta)$ is found using equation 4.

$$q(x_i, \beta_i) = \prod_{k=1}^{K-1} \beta_{ik}^{I(x_i=k)} (1 - \sum_{k=1}^{K-1} \beta_{ik})^{I(x_i=K)}, \qquad \sum_{k=1}^{K} \beta_{ik} = 1 \tag{3}$$

$$q(\mathbf{X}, \beta) = \prod_{X_i \in \mathbf{X}} q(x_i, \beta_i) \tag{4}$$

The divergence of q and p can be seen as a function $F(\beta)$. Equation 5 is the equation for the divergence.

$$F(\beta) = \sum_{x_i} q(x_i, \beta_i) log \frac{q(x_i, \beta_i)}{exp[E_{\mathbf{X}/x_i}[logp(\mathbf{x}, \mathbf{e})]]} + K_{\mathbf{X}/x_i} \tag{5}$$

$F(\beta)$ is minimized when $q(x_i, \beta_i) = exp[\sum[E_{\mathbf{X}/x_i}[logp(\mathbf{x}, \mathbf{e})]]$, so $Beta_{ik}$ can be calculated using equation 6.

$$\beta_{ik} = \frac{exp[E_{\mathbf{X}/x_i}[logp(\mathbf{x}/x_i, x_i = k, \mathbf{e})]]}{\sum_{j=1}^{K} exp[E_{\mathbf{X}/x_i}[logp(\mathbf{x}/x_i, x_i = j, \mathbf{e})]]} \tag{6}$$

By applying the chain rule we can simplify $E_{\mathbf{X}/x_i}[logp(\mathbf{x}/x_i, x_i = k, \mathbf{e})]$ with equation 7.

$$E_{\mathbf{X}/x_i}[logp(\mathbf{x}/x_i, x_i = k, \mathbf{e})] = \sum_{l=1}^{N} E_{\mathbf{X}/x_i}[logp(x_l)|\pi(x_l)] \tag{7}$$

The mean field algorithm in figure 6 uses gradient descent to find the β that minimizes $F(\beta)$ which happens once the iterations of β converges.

▷ **Input**: a BN with evidence $\boldsymbol{e}$ and unobserved variables $\boldsymbol{X}$
▷ **Output**: mean field parameters $\boldsymbol{\beta} = \{\beta_{nk}\}$ for $k = 1, 2, \ldots, K_n$
Randomly initialize the parameters β_{nk} subject to $\sum_{k=1}^{K_n} \beta_{nk} = 1$
while $\boldsymbol{\beta} = \{\beta_{nk}\}$ not converging **do**
  for n=1 to N **do** //explore each node
    for k=1 to $K_n - 1$ **do** //K_n is the number of states for nth node
      Compute β_{nk} using Eq. (3.49)
    end for
  end for
end while

Figure 6: Mean Field algorithm

Once the optimal β is found it is plugged into equation 3 to find $q(x_i, \beta_i) \approx p(x_i|e)$. The mean field method works best on Bayesian networks where the nodes are almost independent of each other. This is not the case for the CHILD network. The nodes of the CHILD network are highly dependent as a result this method is not as accurate as the other methods for this network. The results of the mean field method are expected to be much worse then the other two algorithms.

3 Results and Conclusion

The results found for each method are in the following table. Where E is the CO_2 report is less then 7.5, left ventricular hypertrophy is reported, and the lung x-ray is reported to be Plethoric.

Method	P(Birtha Asphxia|E)	$Disease^*\|E$
likelihood weighted samples	0.0864144488463717	PAIVS
Gibbs sampling	0.089	TGA
Mean field	0.27605244141193713	TGA

As we can see likelihood weighted samples and Gibbs sampling had similar values,8.64% and 8.9%, for the probability of Birth Asphyxia given the evidence. The probability found with the means field method was 27% which was very different from the other two as expected. The one interesting result that was unexpected was that likelihood weighted samples found the disease to most likely be PAVIS while the other two methods found the disease to most likely be TGA. I think this is because of the bias in likelihood weighted samples. From the results we can conclude that the the probability of Birth Asphyxia given the evidence is most likely around 8.7% and that the disease is most likely to be TGA. The strength of likelihood weighted samples is that its easy to implement and there are no parameters to tune. However it shows bias and is not optimal for the CHILD network because the evidence is in the leaves which are far from the root nodes. Gibbs sampling was the best method for this network. However it required parameter tuning. This was easier to do once we had results from likelihood weighted sampling to compare to. Means field was the hardest to implement and gave the worst results for the CHILD network because the CHILD network is highly dependent. Its strength was that it is fast and provided a lower bound for the desired results. From this project I learned a lot about approximate inference and implementation of the different approximate inference methods. It also improved my ability to understand and implement Bayesian networks in python from the last project.

4 Works Cited

[1] Ji, Q. (2020). Chapter 3. In Probabilistic graphical models for computer vision (pp. 59-68). London: Academic Press.