Probabilistic Graphical Models Project 1: Belief Propogation

Matthew Caulfield

ECSE-6810

October 9, 2020

1 Introduction

The purpose of this project was to implement Belief Propagation on a Bayesian network. Inference allows someone to find the probability of an event occurring given the individual knows a related event either occurred or did not occur. A relevant example would be to figure out whether a person has a disease like Covid-19 or just a common cold. For instance both the common cold and Covid-19 have symptoms like a cough and congestion, but only Covid-19 has the symptom of loss of taste and smell. So given a doctor knows their patient lost their sense of smell, inference on a Bayesian network would tell the doctor that their patient had a higher chance of having Covid-19 than if the doctor knew nothing about the patient. Figure 1 is the Bayesian network that inference is done on for this project.

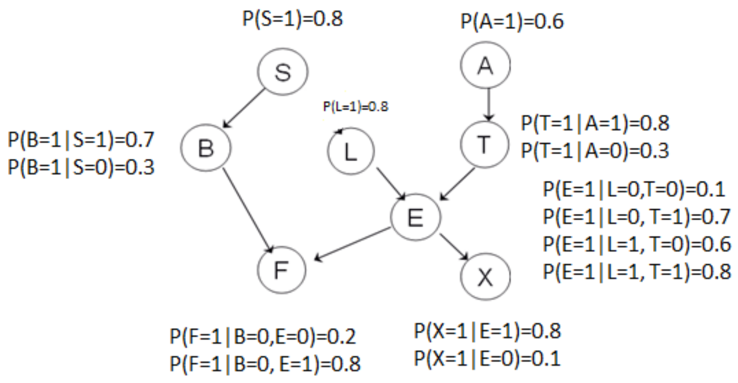
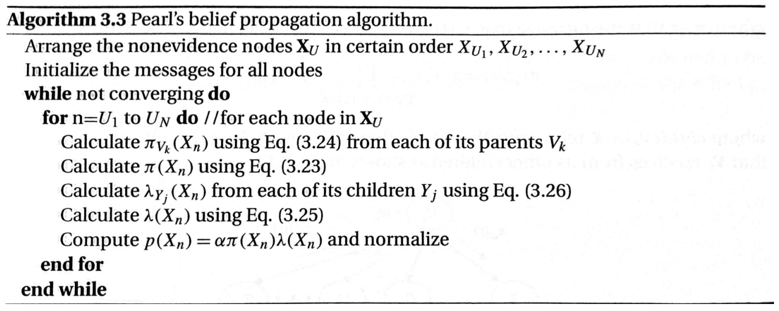


Figure 1: Bayesian Network

There are many types of inference that can be done on a Bayesian network each using different information about the probability distribution of the nodes. Belief propagation is a form of inference on a Bayesian network where the unknowns nodes of the Bayesian network update what they believe their probabilities are based on their relationships with their surrounding nodes. The parent nodes send a message containing the parents node probability and their other children's probability. These messages are weighted by the posterior probability of the node. The children send a message containing their probability and the probability of its other parents. These messages are weighted by the posterior probability of the child node. Three types of inference that will be discussed are the posterior probability inference, the maximum a posteriori (MAP) inference, and the most probable explanation (MPE) inference. The posterior probability inference is used to calculate the posterior probability of unknown nodes in the Bayesian network given observed variables. The sum-product inference algorithm implements the posterior probability inference as a belief propagation algorithm. This is in contrast to both the MAP and MPE inferences which give the most likely configuration of the unknown variables. MPE is the belief propagation method of MAP inference and is implemented computationally with the max-product inference algorithm.

2 Theory

The belief propagation algorithm for the posterior probability inference is the sum-product implementation of Pearl's belief propagation. The general Peal's belief sum-product propagation is shown in figure 2[1].



Algorithm 3.3 Pearl's belief propagation algorithm.

```
Arrange the nonevidence nodes $\mathbf{X}_U$ in certain order $X_{U_1}, X_{U_2}, \ldots, X_{U_N}$
Initialize the messages for all nodes
while not converging do
  for n=$U_1$ to $U_N$ do //for each node in $\mathbf{X}_U$
    Calculate $\pi_{V_k}(X_n)$ using Eq. (3.24) from each of its parents $V_k$
    Calculate $\pi(X_n)$ using Eq. (3.23)
    Calculate $\lambda_{Y_j}(X_n)$ from each of its children $Y_j$ using Eq. (3.26)
    Calculate $\lambda(X_n)$ using Eq. (3.25)
    Compute $p(X_n) = \alpha\pi(X_n)\lambda(X_n)$ and normalize
  end for
end while
```

Figure 2: Pearl's belief sum-product propogation algorithm

Equation 3.23, 3.24, 3.25, and 3.26 are as follows[1]:

$$\pi(X) = \sum_{v_1,\ldots,v_k} p(X|v_1,...,v_k) \prod_{k=1}^{K} \pi_{v_K}(X) \tag{3.23}$$

$$\pi_{v_k} X = \pi(v_k) \prod_{C \in child(V_k) \setminus X} \lambda_C(V_k) \tag{3.24}$$

$$\lambda(X) = \prod_{j=1}^{J} \lambda_{Y_j}(X) \tag{3.25}$$

$$\lambda_{Y_j}(X) = \sum_{y_j} \lambda(y_j) \sum_{u_1,u_2,\ldots u_p} p(y_j|X, u_1, u_2, .., u_p) \prod_{k=1}^{p} \pi_{u_k}(y_j) \tag{3.26}$$

Belief propagation works by propagating the effects of the evidence E throughout the network and how the evidence influences each node from their parents and from their children separately. The separated influences are then multiplied together and the result is normalized to make a probability distribution $p(X|E)$ where X is the unknown node being influenced by E.

While the sum-product inference algorithm finds posterior probability, the max-product algorithm finds the most probable configuration of the unknown variables. The max-product method uses the same pearl's belief algorithm in figure 2 with different equations for λ and π. Instead of the summation of the posterior probabilities of the unknown node the maximum posterior probability is used. The following equations are used to find π and λ.

$$\pi(X) = \max_{V_1,..,V_k} P(X|V_1, V_2, ...V_K)) \prod_{k=1}^{K} \pi_{v_K}(X) \tag{1}$$

$$\lambda_{Y_j}(X) = \max_{y_j} \lambda(y_j) \max_{u_1,u_2,\ldots u_p} p(y_j|X, u_1, u_2, .., u_p) \prod_{k=1}^{p} \pi_{u_k}(y_j) \tag{3.26}$$

Once the probabilities $p(X|E)$ converge the most likely configuration of each x based on this prior probability distribution is chosen as the configuration for each unknown variable. By using the maximum of each posterior probability instead of summing them the most likely configuration is found.

To check the accuracy of each implemented function variable elimination inference was used. For the sum-product algorithm, variable elimination for posterior inference was used to find a theoretical posterior probability of each unknown node. Variable elimination uses a normalized joint distribution of the joint distribution of the known variables and unknown variable. Where undesired variable are removed from the joint distribution by summation and the chain rule is used to find the joint distribution. For example if $p(A|E = e)$ is desired to be calculated the following formula[1] is used.

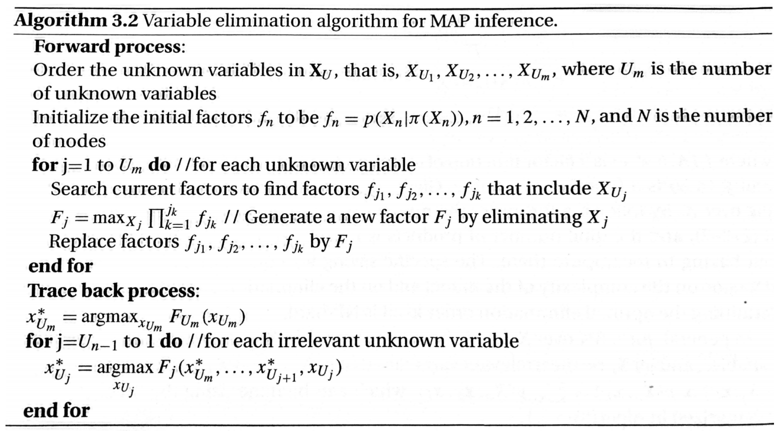
$$\begin{aligned} p(A|E = e) &= \alpha p(A, E = e) && (2) \\ &= \alpha \Sigma_{b,c,d} p(A, b, c, d, e) && (3) \\ &= \alpha \Sigma_{b,c,d} p(A) p(b, c, d, e|A) && (4) \end{aligned}$$

The variable alpha is the used to normalize the probability. The goal of this project was to compute $p(X_q|F = 1, X = 0)$, $X_q \in \{S, A, B, L, T, E\}$. The theoretical posterior probability of each unknown where found with the method described in equation X and were found to be as follows.

$$\begin{aligned} p(B|F = 1, X = 0) &= (0.3846, 0.6154) \\ p(S|F = 1, X = 0) &= (0.2507, 0.7493) \\ p(E|F = 1, X = 0) &= (0.6609, 0.3391) \\ p(L|F = 1, X = 0) &= (0.2443, 0.7557) \\ p(T|F = 1, X = 0) &= (0.4642, 0.5358) \\ p(A|F = 1, X = 0) &= (0.4424, 0.5576) \end{aligned}$$

Where each prior probability is of the form $p(x_q = 0, x_q = 1)$.

To calculate the theoretical assignments of the most probable explanation algorithm the variable elimination algorithm for MAP inference was used. This algorithm is from professor Ji's textbook Probabilistic Graphical Models for Computer Vision[1].



Algorithm 3.2 Variable elimination algorithm for MAP inference.

```
  Forward process:
  Order the unknown variables in $\mathbf{X}_U$, that is, $X_{U_1}, X_{U_2}, \ldots, X_{U_m}$, where $U_m$ is the number of unknown variables
  Initialize the initial factors $f_n$ to be $f_n = p(X_n|\pi(X_n)), n = 1, 2, \ldots, N$, and $N$ is the number of nodes
  for j=1 to $U_m$ do //for each unknown variable
    Search current factors to find factors $f_{j_1}, f_{j_2}, \ldots, f_{j_k}$ that include $X_{U_j}$
    $F_j = \max_{X_j} \prod_{k=1}^{j_k} f_{j_k}$ // Generate a new factor $F_j$ by eliminating $X_j$
    Replace factors $f_{j_1}, f_{j_2}, \ldots, f_{j_k}$ by $F_j$
  end for
  Trace back process:
  $x^*_{U_m} = \text{argmax}_{x_{U_m}} F_{U_m}(x_{U_m})$
  for j=$U_{n-1}$ to 1 do //for each irrelevant unknown variable
    $x^*_{U_j} = \underset{x_{U_j}}{\text{argmax}} F_j(x^*_{U_m}, \ldots, x^*_{U_{j+1}}, x_{U_j})$
  end for
```

Figure 3: Variable elimination algorithm for MAP inference

The variable elimination for MAP algorithm in figure 3 found the most likely configuration of the nodes to be $B = 1, S = 1, E = 0, L = 1, T = 1, A = 1$.

3 Results

The variable elimination inference method was used to find the theoretical prior probability distribution of each unknown variable given $F = 1$ and $X = 0$.

$$\begin{gathered}
p(B|F = 1, X = 0) = (0.15396060227340358, 0.8460393977265964))\\
p(S|F = 1, X = 0) = (0.18057830750594686, 0.8194216924940531))\\
p(E|F = 1, X = 0) = (0.6780650542118433, 0.32193494578815685)\\
p(L|F = 1, X = 0) = (0.2638309702529885, 0.7361690297470114)\\
p(T|F = 1, X = 0) = (0.45871559633027514, 0.5412844036697247)\\
p(A|F = 1, X = 0) = (0.45439407955596667, 0.5456059204440333)
\end{gathered}$$

This took 5 iterations to converge. The two probability distribution most different from the theoretical results were $p(B|F = 1, X = 0)$ and $p(S|F = 1, X = 0)$. The percent difference between the theoretical and experimental results for $p(B|F = 1, X = 0)$ was 37% and the difference be the theoretical and experimental results for $p(S|F = 1, X = 0)$ was 10%. The experimental result that was the least different was $p(A|F = 1, X = 0)$ with a difference between the theoretical and experimental result 2.15%. Four of the experimental posterior probabilities were within 5% of the the theoretical result and can be explained by rounding in the experimental result. The two that were not were very different from the theoretical solution. A possible reason for this is a miscalculation of the theoretical results. The experiment also converged fairly quickly with five iterations which is expected because it is a small Bayesian network.

The MPE experiment took 4 iterations and found that the unknown variables converge as follows $B = 1, S = 1, E = 0, L = 1, T = 1, A = 1$. This matched the theoretical results for MAP variable elimination. This experiment was successful it as every unknown variable configured as expected for the most probable explanation. It also took one less iteration to converge then the sum-product experiment.

4 Conclusion

In this project Pearl's Belief propagation was implemented for both sum-product and max-product. The sum-product finds variables their posterior probability given some variables in the Bayesian network. The max-product implementation is a form of most probable explanation inference which assigns unknown variables their most likely configuration. MPE is an a belief propagation implementation of the maximum aposteriori inference of a Bayesian network. Some issues encountered in this project was to make sure that the correct instance of each probability were being multiplied or added together. For example each probability distribution had a probability for the variable configured to 0 or 1. The trouble faced was to make sure that the correct probability was being used so if the posterior probability was for the configuration where its first parent was configured as 0 then that probability needed to be multiplied by the probability the first parent was 0. This may seem easy but it is tricky when it needs to work for each parent being configured differently. This project deepened my understanding of Bayesian networks in general as well as how MAP inference and posterior probability inference work.

5 Works Cited

[1] Ji, Q. (2020). Chapter 3. In Probabilistic graphical models for computer vision (pp. 42-52). London: Academic Press.