Lecture 28 - BNet Inference

`https://www.cs.ubc.ca/~jordon/teaching/cpsc322/2019w2/lectures/lecture28.pdf`

Goals

- Define factors. Derive new factors from existing factors. Apply operations to factors, including assigning, summing out and multiplying factors.
- Carry out variable elimination by using factor representation and using the factor operations. Use techniques to simplify variable elimination.

BNet Inference

Our goal in BNet inference is to compute the probabilities of the variables in a belief network:

- What is the posterior distribution over one or more variables, conditioned on one or more observed variables?

BNets in General

Suppose we have that the variables in the belief network are $\{X_1, \ldots, X_n\}$. We also have a variable Z which we call the "query" variable.

We have variables $Y_1 = v_1, \ldots, Y_j = v_j$ which are the **observed varg1ariables**, and their corresponding observed values. $Z_1, \ldots, Z_k$ are the remaining variables.

We wish to compute:

$$P(Z|Y_1 = v_1, \ldots, Y_j = v_j)$$

We bring back the example of the fire alarm, where we had the following belief network:

In this case we may wish to compute: $P(L|S = t, R = f)$. From marginalization we know that:

$$P(L|S = t, R = f) = \frac{P(L, S = t, R = f)}{P(S = t, R = f)}$$

An we are given the necessary data to do so:

L	*S*	*R*	*P(L, S=t, R=f)*
t	t	f	.3
f	t	f	.2

.3 + .2 = .5

L	*S*	*R*	*P(L | S=t, R=f)*
t	t	f	.6
f	t	f	.4

In general, we have

$$\begin{aligned} P(Z|Y_1 = v_1, \ldots, Y_j = v_j) &= \frac{P(Z, Y_1 = v_1, \ldots, Y_j = v_j)}{P(Y_1 = v_1, \ldots, Y_j = v_j)} \\ &= \frac{P(Z, Y_1 = v_1, \ldots, Y_j = v_j)}{\sum_Z P(Z, Y_1 = v_1, \ldots, Y_j = v_j)} \end{aligned}$$

However we only need to **compute the numerator**, and then **normalize**. This can be framed in terms of **operations between factors** (that satisfy the semantics of probability)

Factors

Def: A ***Factor*** is a representation of a function from a tuple of random variables into a number:

- We will denote the factor f of random variables $X_1, \ldots, X_j$ as $f(X_1, \ldots, X_j)$

A Factors can denote:

$\rightarrow$ One distribution

$\rightarrow$ One partial distribution

$\rightarrow$ Several distributions

$\rightarrow$ Several partial distributions

over the given tuple of variables.

Examples

- Distribution
  $P(X_1, X_2)$ is a factor $f(X_1, X_2)$

X_1	X_2	$f(X_1, X_2)$
T	T	0.12
T	F	0.08
F	T	0.08
F	F	0.72

- Partial distribution
  $P(X_1, X_2 = F)$ is a factor $f(X_1)_{X_2=F}$

X_1	X_2	$f(X_1)_{X_2=F}$
F	T	0.08
F	F	0.72

- Set of Distributions
  $P(X|Z,Y)$ is a factor $f(X,Z,Y)$
- Set of partial Distributions
  $P(X_1, X_3 = v_3|X_2)$ is a factor $f(X_1, X_2)_{X_3=v_3}$

Operations on factors

We can make new factors based on existing factors.

Variable assignment

To begin with, for instance it is possible to assign some, if not all of the variables to a factor. This assignment then reduces the factor dimension (i.e.: the number of variables in a factor).

For instance if we have a factor $f(X,Y,Z)$, what would be the result of assigning $X = t$? TFAE:

$$f(X = t, Y, Z) = f(X, Y, Z)_{X=t} = f(Y, Z)$$

X	Y	Z	val
t	t	t	0.1
t	t	f	0.9
t	f	t	0.2
t	f	f	0.8
~~f~~	~~t~~	~~t~~	~~0.4~~
~~f~~	~~t~~	~~f~~	~~0.6~~
~~f~~	~~f~~	~~t~~	~~0.3~~
~~f~~	~~f~~	~~f~~	~~0.7~~

Summing out a variable example

We can ***sum out a variable***. WLOG let that variable be X_1, with domain $\{v_1, \ldots, v_k\}$. We can sum it out from factor $f(X_1, \ldots, X_j)$ resulting in a factor on $X_2, \ldots, X_j$ defined by:

$$\left(\sum_{X_1} f\right)(X_2, \ldots, X_j) = f(X_1 = v_1, X_2, \ldots, X_j) + \cdots + f(X_1 = v_k, X_2, \ldots, X_j)$$

$f_3(A,B,C)$:

B	A	C	val
t	t	t	0.03
t	t	f	0.07
f	t	t	0.54
f	t	f	0.36
t	f	t	0.06
t	f	f	0.14
f	f	t	0.48
f	f	f	0.32

$\sum_B f_3(A,B,C)$:

A	C	val
t	t	0.57
t	f	0.43
f	t	
f	f	

Multiplying factors

Factors can also be multiplied together. The ***product*** of a factor $f_1(A,B)$ and $f_2(B,C)$ where B is the variable in common is the factor $f_1 \times f_2(A,B,C)$ defined by:

$$f_1(A,B)f_2(B,C) = (f_1 \times f_2)(A,B,C)$$

Notes:

- It is defined on all A, B, C triples, obtained by multiplying together the appropriate pair of entries from f_1 and f_2.
- A, B, C can be sets of variables.

Intro to Variable Elimination

Suppose we again have the scenario from above:

The variables in the belief network are $\{X_1, \dots, X_n\}$. Z is the query variable.

We have variables $Y_1 = v_1, \dots, Y_j = v_j$ which are the **observed variables**, and their corresponding observed values. $Z_1, \dots, Z_k$ are the remaining variables.

We want to compute:

$$P(Z|Y_1 = v_1, \dots, Y_j = v_j)$$

We have then showed that what really need to be computed is:

$$P(Z, Y_1 = v_1, \dots, Y_j = v_j)$$

And this expression can be computed in terms of operations between the factors (that satisfy the semantics of probability).

If we express the joint distribution as a single factor:

$$f(Z, Y_1, \dots, Y_j, Z_1, \dots, Z_k)$$

Then we can compute $P(Z, Y_1 = v_1, \dots, Y_j = v_j)$ by:

- **Assigning** $Y_1 = v_1, \dots, Y_j = v_j$

- **Summing out** the variables $Z_1, \ldots, Z_k$

$$P(Z, Y_1 = v_1, \ldots, Y_j = v_j) = \sum_{Z_k} \ldots \sum_{Z_1} f(Z, Y_1, \ldots, Y_j, Z_1, \ldots, Z_k)_{Y_1=v_1,\ldots,Y_j=v_j}$$

However, the joint distribution is too big to sum this for large problems. Thus we need to employ other strategies.

Lecture 29 - Variable Elimination

`https://www.cs.ubc.ca/~jordon/teaching/cpsc322/2019w2/lectures/lecture29.pdf` From the previous section we derived that:

$$P(Z, Y_1 = v_1, \dots, Y_j = v_j) = \sum_{Z_k} \dots \sum_{Z_1} f(Z, Y_1, \dots, Y_j, Z_1, \dots, Z_k)_{Y_1=v_1,\dots,Y_j=v_j}$$

But the whole point of BNets was to get rid of the JPD. Using the chain rule and the definition of a BNet we can write $P(X_1, \dots, X_n)$ as:

$$\prod_{i=1}^{n} P(X_1| \underbrace{\text{par}(X_1)}_{\text{parents of } X_i})$$

Using the chain rule we can also express the joint factor as a product of factors:

$$f(Z, {\color{green}Y_1, \dots, Y_j}, {\color{blue}Z_1, \dots, Z_k}) = \prod_{i=1}^{n} f(X_i|\text{par}(X_i))$$

$$P(Z, Y_1 = v_1, \dots, Y_j = v_j) = \sum_{Z_k} \dots \sum_{Z_1} \prod_{i=1}^{n} f(X_i|\text{par}(X_i))_{Y_1=v_1,\dots,Y_j=v_j}$$

We therefore observe that inference in belief networks reduces to computing the **sum of products**. The algorithm consequently is:

1. Construct a factor for each conditional probability
2. In each factor assign the observed variables to their observed values.
3. Multiply the factors.
4. for each of the other variables $Z_i \in \{Z_1, \dots, Z_k\}$, **sum out** Z_i

How to simplify the Computation?

Let's assumes we have turned the CPTs into factors, and performed the assignments. We can focus on a basic case, to illustrate the decomposition. This case takes form when $k = 1$, and the distribution function is:

$$\sum_{Z_1} \prod_{i=1}^{n} f(\text{vars}(X_i))$$

We have that in order to compute this efficiently, we first need to factor out the terms that do not involve Z_1. The above statement then becomes:

$$\left(\prod_{i:Z_1 \notin \text{vars}(X_i)} f(\text{vars}(X_i)) \right) \cdot \left(\sum_{Z_1} \prod_{i:Z_1 \in \text{vars}(X_i)} f(\text{vars}(X_i)) \right)$$

This is illustrated by the following specific case:

$$\sum_{A} f(C, D) \times f(A, B, D) \times f(E, A) \times f(D) =$$
$$f(C, D) \times f(D) \times \sum_{A} f(A, B, D) \times f(E, A)$$

This simplification is similar to what you can do in basic algebra with multiplication and addition:

- It takes 14 multiplications or additions to evaluate the expression:

$$ab + ac + ad + aeh + afh + agh$$

- This expression be evaluated more efficiently (only 7 operations)

$$a \times (b + c + d + h \times (e + f + g))$$

There also is numerous ways in which it is possible to simplify the expression.

Variable elimination algorithm

Given

$$f(Z, Y_1, \ldots, Y_j, Z_1, \ldots, Z_k)$$

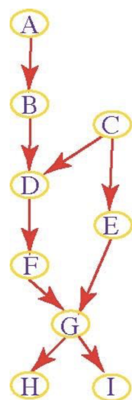
To compute:

$$P(Z|Y_1 = v_1, \ldots, Y_j = v_j)$$

1. Construct a factor for each conditional probability.
2. Set the observed variables to their observed values
3. Given an elimination ordering, simplify/decompose sum of products
4. Perform products and sum out Z_i
5. Multiply the remaining factors (All the remaining factors are in Z as we would've assigned all the evidence variables, and summed out all of the others).
6. Normalize: divide the resulting factor $f(Z)$ by $\sum_Z f(Z)$

Example

Consider the following BNet:



Suppose we want to compute $P(G|H = h_1)$. We know that:

$$P(G, H) = \sum_{A,B,C,D,E,F,I} P(A, B, C, D, E, F, G, H, I)$$

Using the chain rule, and conditional probability, we obrain the following decomposition:

$$= \sum_{A,B,C,D,E,F,I} P(A)P(B|A)P(C)P(D|B,C)P(E|C)P(F|D)P(G|E,F)P(H|G)P(I|G)$$

We use factors now to represent the probabilities:

$$= \sum_{A,B,C,D,E,F,I} f_0(A)f_1(A,B)f_2(C)f_3(D,B,C)f_4(E,C)f_5(F,D)f_6(G,F,E)f_7(H,G)f_8(I,G)$$

We have ovserved $H = h_1$, thus the expression simplifies to:

$$= \sum_{A,B,C,D,E,F,I} f_0(A)f_1(A,B)f_2(C)f_3(D,B,C)f_4(E,C)f_5(F,D)f_6(G,F,E)f_9(G)f_8(I,G)$$

We employ an elimination ordering of A, C, E, I, B, D, F:

$$\begin{aligned}
&= f_9(G)\sum_F\sum_D f_5(F,D)\sum_B\sum_I f_8(I,G)\sum_E f_6(G,F,E)\sum_C f_2(C)f_3(D,B,C)f_4(E,C)\cdot \\
&\qquad \cdot \underbrace{\sum_A f_0(A)f_1(A,B)}_{=f_{10}(B)} \\
&= f_9(G)\sum_F\sum_D f_5(F,D)\sum_B f_{10}(B)\sum_I f_8(I,G)\sum_E f_6(G,F,E)\underbrace{\sum_C f_2(C)f_3(D,B,C)f_4(E,C)}_{=f_{12}(B,D,E)} \\
&= f_9(G)\sum_F\sum_D f_5(F,D)\sum_B f_{10}(B)\sum_I f_8(I,G)\underbrace{\sum_E f_6(G,F,E)f_{12}(B,D,E)}_{=f_{13}(B,D,F,G)} \\
&= f_9(G)\sum_F\sum_D f_5(F,D)\sum_B f_{10}(B)f_{13}(B,D,F,G)\underbrace{\sum_I f_8(I,G)}_{=f_{14}(G)} \\
&= f_9(G)f_{14}(G)\sum_F\sum_D f_5(F,D)\underbrace{\sum_B f_{10}(B)f_{13}(B,D,F,G)}_{=f_{15}(D,F,G)} \\
&= f_9(G)f_{14}(G)\sum_F\underbrace{\sum_D f_5(F,D)f_{15}(D,F,G)}_{=f_{16}(F,G)} \\
&= f_9(G)f_{14}(G)\underbrace{\sum_F f_{16}(F,G)}_{=f_{17}(G)} \\
&= f_9(G)f_{14}(G)f_{17}(G)
\end{aligned}$$

We multiply the remaining factors to obtain:

$$P(G,H) \propto f_{18}(G)$$

Which we need to normalize to get:

$$P(G, H) = \frac{f_{18}(G)}{\sum_{g \in \mathrm{dom}(G)} f_{18}(G)}$$

Variable Elimination: Independence

Variable elimination looks incredibly complicated for large graphs. We used conditional independence:

$$\prod_{i=0}^{n} P(X_i | \mathrm{par}(X_1))$$

We inquire whether or not we can use this conditional independence to make variable elimination simpler. The answer is **YES**, because all of the variables for which the query is conditionally independent given the observations can be pruned from the BNet (as can the **unobserved leaf nodes**).

Lecture 30 - Markov Models

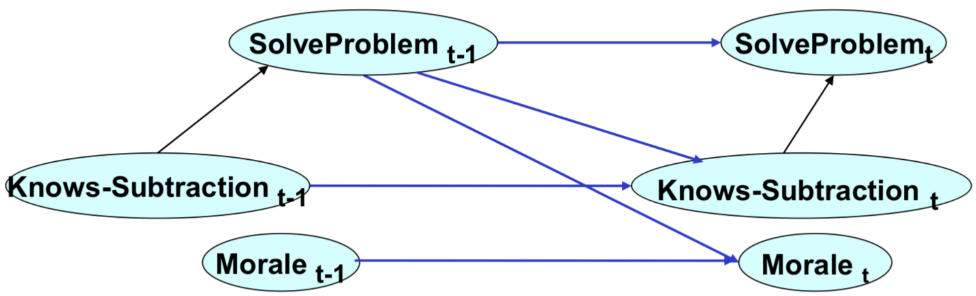
https://www.cs.ubc.ca/~jordon/teaching/cpsc322/2019w2/lectures/lecture30.pdf

Modeling Static Environments

- So far we have used Bnets to perform inference in static environments
- For instance, the system keeps collecting evidence to diagnose the cause of a fault in a system (e.g., a car).
- The environment (values of the evidence, the true causes of the fault) does not change as I gather new evidence

Modeling Evolving Environments

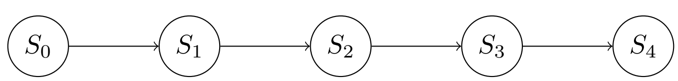
- Often we need to make inferences about evolving environments
- Represent the state of the world at each specific point in time via a series of snapshots, or ***time slices***



Markov Models

What is the simplest possible Dynamic Belief/Bayesian Network (DBN)?

We would model this by using one variable per time slice. Let's assume S_t represents each state at time t with domain $\{v_1, \dots, v_n\}$



A stationary *Markov* chain is described as follows for all $t > 0$:

- $P(S_{t+1}|S_0, \dots, S_t) = P(S_{t+1}|S_t)$ (Markov Assumption)
- $P(S_{t+1}|S_t) = P(S_{t'+1}|S_{t'}), \forall t, t'$ (stationary assumption)

Consequently, all that needs to be specified is $P(S_0)$ and $P(S_{t+1}|S_t)$:

- It is a simple Model, easy to specify
- It is often the natural model
- The network can extend indefinitely
- Variations of SMC are at the core of many Natural Language Processing (NLP) applications!