Tutorial Alireza Shafaei

March 14, 2016 1:05 PM

We will cover 3 topics ① 3 common tasks ② Bayesian Networks ③ Markov Networks.

Given a probability function $p(x_1, x_2, \dots, x_d)$, the common 3 tasks that interest us are

① Decoding $\arg\max_x p(x)$ or the most likely $x$.

If I give you a black-box $P$ function, how would you do this task?

You need to enumerate over all possible $X$, and keep track of the max.

If $x_i \in \{1, \dots, k\}$, the runtime is $\Theta(d^k)$! note the $\Theta$!

Can we do better? In general, the answer is no!

② Inference

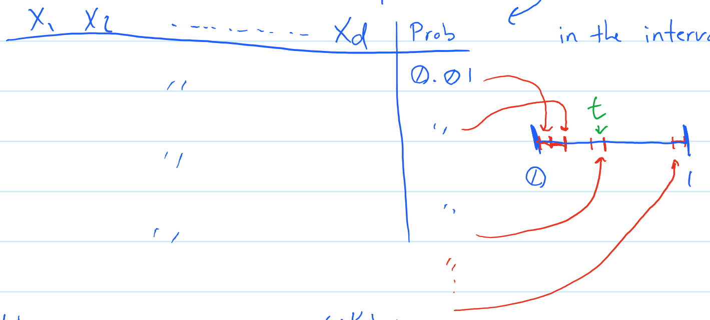
→ deriving the marginal $P(X_j = s)$ ⇒ need to enumerate and sum, Can't do better!

→ finding the normalization constant $Z = \sum_x P(x)$ ⇒ In this case $Z=1$ by definition.

③ Sampling

This $p(X)$ is characterizing a distribution from which we would like to take a sample!

A naive way is to make a prob. table, Each entry is then assigned to a segment in the interval $[0, 1]$. We pick a $t \sim \mathrm{Unif}([0\ 1])$ and return the corresponding $x$ from the table.




this again requires $O(d^k)$!

We can also look at these tasks with conditioning, for instance, what's the most likely sequence assuming $X_5 = 2$?

Decoding

So what do we do with these intractable tasks?

one solution is to decompose $P(X)$ into smaller functions that we can handle better!

☆ Does $p(X)$ necessarily have a decomposition? NO!

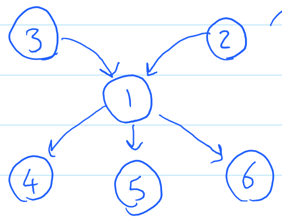
By decomposing $P(X)$ into smaller functions we're {enforcing / assuming} relationship between the variables.

So how do we decompose $p(x)$?

The two approaches that we cover are called ① Bayesian Networks, ② Markov Networks.

Bayesian Networks (DAG models) → Directed Acyclic Graph

Here we assume $P(x) = \prod_{j=1}^{d} P(X_j \mid \underbrace{X_{pa}(j)}_{\text{parents of } X_j})$ Assume $X_j \in \{1,2,3\}$



$\rightarrow P(x) = P(X_2)\,P(X_3)\,\underbrace{P(X_1 \mid X_2, X_3)}\,P(X_4 \mid X_1)\,P(X_5 \mid X_1)\,P(X_6 \mid X_1)$

| $X_1$ | $X_2$ | $X_3$ | P |
|---|---|---|---|
| 1 | 1 | 1 | $P_1$ |
| 1 | 1 | 2 | $P_2$ |
| ⋮ | | ⋮ | ⋮ |
| 3 | 3 | 3 | $P_{27}$ |

$\rightarrow$ How many entries? $\Theta(n^K)$ ($n$ → number of states, $K$ → number of variables)

$\rightarrow \sum P_i = 1$ by definition

# Sampling?

is straightforward, start from the parent and sample wrt to their tables and then proceed to the children!

# Inference?

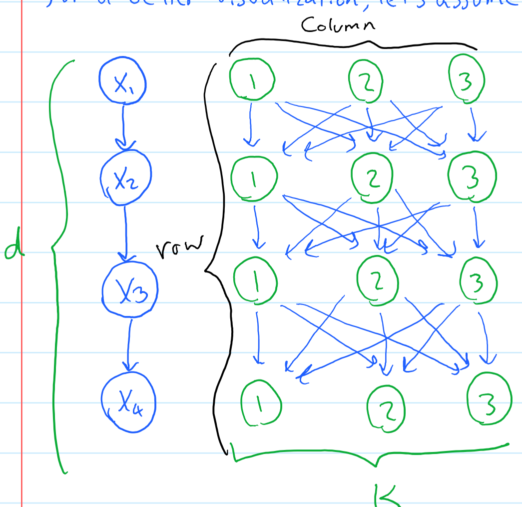
The marginal distribution is calculated by factorizing the $p(x)$ and summing over the other variables. (one way)

It's important to understand the idea, its variations show up often.

# Decoding?

There's a dynamic programming method called Viterbi decoding.

For a better visualization, let's assume we have a chain, and $X_j \in \{1,2,3\}$




$\rightarrow$ For each node in our original chain, we add a row of $\overset{3}{K}$ nodes and we connect them to all the the nodes of the next row.

$\rightarrow$ On each edge $X_i^{(u)} \rightarrow X_{i+1}^{(v)}$ we put a weight $P(X_{i+1} = v \mid X_i = u)$

$\rightarrow$ Each path from the top row to the bottom row denotes a possible configuration of the X vector! Furthermore if we multiply the weights on the edges we'd get the corresponding probability!

$\rightarrow$ Let's define the weight of a path to be the multiplication of its edges. $W(P) = \prod_{v_i, v_j \in P} w(v_i, v_j)$ $\quad W(\Phi) = 1$

$\rightarrow$ In this setting, decoding corresponds to finding the maximum weight path.

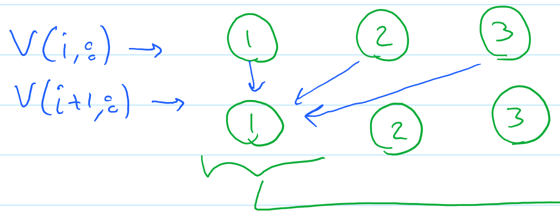
A dynamic programming solution:

Let's define $V(i,j)$ to be the weight of the maximum-weight-path ending at $X_i^{(j)}$ (row $i$ column $j$)

$\rightarrow$ $V(1,:)$ is $\underline{1}$ for all the entries since there are no edges involved.

(matlab notation for all columns of row 1)

$\rightarrow$ Given $V(i,:)$ how can we calculate $V(i+1,:)$?




$$V(i+1,1) = \max \begin{cases} V(i,1) \cdot P(X_{i+1}=1 \mid X_i = 1) \\ V(i,2) \cdot P(X_{i+1}=1 \mid X_i = 2) \\ V(i,3) \cdot P(X_{i+1}=1 \mid X_i = 3) \end{cases}$$

In order to end up at $X_{i+1}^{(1)}$ we must've inevitably came from $X_i^{(1)}$ or $X_i^{(2)}$ or $X_i^{(3)}$, and we know what's the max-weight-path that ends at those vertices ($V(i,:)$), therefore $V(i+1,1)$ is going to be the case where the corresponding $V(i,-)$ times the weight of the new edge is maximized.

Similarly we do this for $X_{i+1}^{(2)}$ and $X_{i+1}^{(3)}$. Each row requires $\Theta(k^2)$ operations.

OK, So we have $V(1,:)$ and given $V(i,:)$ we know how to calculate $V(i+1,:)$.

Therefore we can figure out the entire table

$V(1,:) \to V(2,:) \to V(3,:) \to V(4,:)$

A decoding will be the assignment that gives us $\max(V(\text{end},:))$

→ but we have only calculated the weight of the maximum-weight-path. How do we reconstruct the path itself?

→ When we were deciding between the possible decisions (max), we also save the decision, the arg max, into a separate table $V'$.

let's say $V(4,2)$ is the final max, then we know $X_4 = 2$

now we look at $V'(4,2)$ which gives us the index for $X_3$

$\Rightarrow X_3 = V'(4,2) \quad X_2 = V'(3, V'(4,2)) \quad X_1 = V'(2, V'(3, V'(4,2)))$

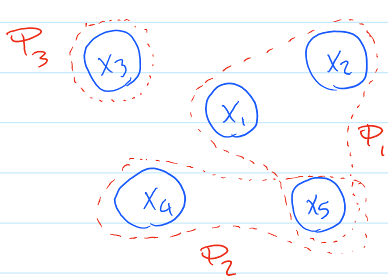
→ What if it's not a chain?

↳ This analogy doesn't work, but the intuition is the same. (will see later)

---

Markov Networks (UGM) ↳ Undirected Graphical Models

Here we assume $P(X) = \frac{1}{Z} \prod_{c \in C} \Phi_c(X_c)$ (a set of functions that depend on subsets of X)




$P(X) = \frac{1}{Z} \Phi_1(X_1, X_2, X_5) \Phi_2(X_4, X_5) \Phi_3(X_3)$

$\Phi_i \geq 0$ are called potential functions

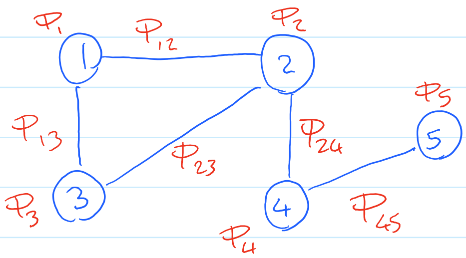
$Z = \sum_X \prod_{c \in C} \Phi_c(X_c)$ ensures $P(X)$ sums to one!

These UGMs are not necessarily easier to handle, so we further restrict ourselves to Pairwise UGMs

$$P(X) = \frac{1}{Z} \prod_j \Phi_j(X_j) \prod_{(i,j) \in E} \Phi_{ij}(X_i, X_j)$$

$\Phi$'s are functions of two variables at most!

$(i,j)\in E$ at most!

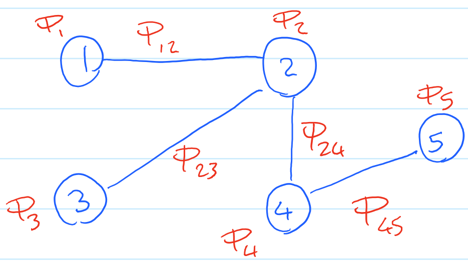



$$P(x) = \frac{1}{Z}\phi_1(x_1)\phi_2(x_2)\phi_3(x_3)\phi_4(x_4)\phi_5(x_5)\,\phi_{12}(x_1,x_2)\phi_{13}(x_1,x_3)\phi_{23}(x_2,x_3)\phi_{24}(x_2,x_4)\,\phi_{45}(x_4,x_5)$$

- Decoding
- Inference
- Sampling

even now not necessarily easy to solve → let's further restrict for now. Mark will probably talk about it later

Tree structured Pairwise UGMs! (no cycles allowed)




$$P(x) = \frac{1}{Z}\phi_1(x_1)\phi_2(x_2)\phi_3(x_3)\phi_4(x_4)\phi_5(x_5)\,\phi_{45}(x_4,x_5)\phi_{12}(x_1,x_2)\phi_{23}(x_2,x_3)\phi_{24}(x_2,x_4)$$

→ Decoding, in the chain case it's similar as before, except a minor change

$$V(i+1,j) = \max\begin{cases} V(i,1)\cdot\phi_{i,i+1}(1,j)\cdot\phi_{i+1}(j) \\ V(i,2)\cdot\phi_{i,i+1}(2,j)\cdot\phi_{i+1}(j) \\ V(i,3)\cdot\phi_{i,i+1}(3,j)\cdot\phi_{i+1}(j) \end{cases} \qquad V(1,:) = [\phi_1(1), \phi_1(2), \phi_1(3)]$$

→ Inference

→ Partition $Z$, is now the sum of weights of all paths! the same DP approach works, but now we do

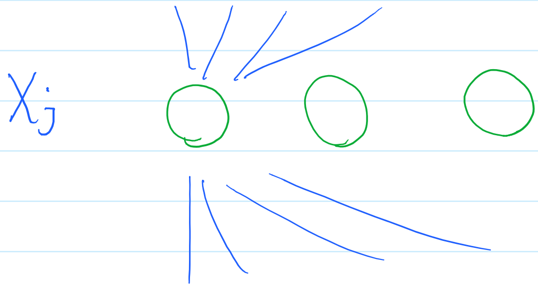
$$V(i+1,j) = \sum\begin{cases} V(i,1)\cdot\phi_{i,i+1}(1,j)\cdot\phi_{i+1}(j) \\ V(i,2)\cdot\phi_{i,i+1}(2,j)\cdot\phi_{i+1}(j) \\ V(i,3)\cdot\phi_{i,i+1}(3,j)\cdot\phi_{i+1}(j) \end{cases}$$

$Z = \text{Sum}(V(\text{end},:))$ We need $Z$ to get the probability!

→ Marginal

When I ask for $P(X_j = s)$, it's like asking what's the sum of weights of all paths that go through $X_j^{(s)}$

of all paths that go through $X_j^{(s)}$




When we were calculating $Z$ in the previous Section, $V(i,j)$ was the sum of all paths that end up at $X_i^{(j)}$ from the top!

let's repeat the same procedure from the bottom to get $\bar{V}(i,j)$ the sum of all paths that end up at $X_i^{(j)}$ from the bottom!

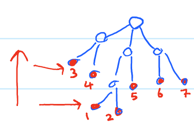
The marginal will be $\dfrac{V(i,j) \cdot \bar{V}(i,j)}{Z}$

Each path that ends at $X_i^j$ with weight $V(i,j)$ can continue from $X_i^j$ with any path in $\bar{V}(i,j) \rightarrow$ the total is $V(i,j) \cdot \bar{V}(i,j)$

↳ Sampling

↳ Calculate $Z$, and then sample from the end $S_d \sim V(d, :)/Z$
then sample $S_{d-1}$ given $S_d$ i.e. $S_{d-1} \sim V(d-1, :) \cdot \Phi_{d-1,d}(:, S_d) \cdot \Phi_d(S_d) / Z$
and go all the way back to the first node.

When it's not a chain, the dp solution starts at the leaves of the graph and builds the answer going up using the same ideas.




You can find $Z$ when you reach the top and with a backward pass you can get the marginals and Sampling is like before.

---

How do we learn $\Phi_s$ from a dataset $\{(X_i)\}_{i=1}^{n}$?

What if we want to learn a model for $P(Y_i | X_i)$?

What if our UGM is not a tree?

How do we convert Bayesian Nets to Markov nets?