iPMR

Our Journey

Graphical Models	Decision Theory	Learning Probabilistic Models	Mixture and Factor Models	Markov Models	Approximate Inference

- Informally Introduce Belief Networks
- Formalise
  - Graph Theory
  - Probabilistic Graphical Models
    - Belief Networks (Bayesian Networks)
    - Markov Networks
    - Factor Graphs

Graphical Models

- Belief networks are examples of probabilistic graphical models.
- Graphical models encode structural aspects of probability distributions.
- There are other forms of graphical models
  - Factor graphs
  - Markov networks (undirected graphical models)
  - Others (structural equation models, causal models, etc. – we will not spend much time on these on this course).

Factor Graphs

- Factor Graphs
  - We noted we could write $$P(\mathbf{x}) = P(x_1)\prod_{i=2}^{M} P(x_i|x_{<i})$$
  - However we can also write $$P(\mathbf{x}) = \frac{1}{Z}\prod_{i=1}^{N_f}\psi_i(\mathbf{x})$$

Z is a normalisation const., ψ terms are 'factors'. *Nf* is number of factors.

- Yes. But. Is this a useful way to split up *P(x)?*
- Those who did MLPR: remember exponential family distributions?

$$P(\mathbf{x}) = \frac{1}{Z}\exp\left(f(\mathbf{x}) + \sum_{i=1}^{N_f} w_i\phi_i(\mathbf{x})\right)$$

- Note that this can be written in factor representation.
- Sometimes not all the variables are expressed in every factor. Often each factor only contains a few variables.
- E.g. $\psi_i(x_1,x_2,x_3,x_4,x_5) = \psi_i(x_1,x_2)$

Building a Factor Graph

- Let C_i denote the set of indices of the variables that occur in the ith factor.
- Let χ_i collect the variables indexed by C_i. E.g. $\chi_1 = (x_1, x_3, x_7, x_8)$. Then we can write

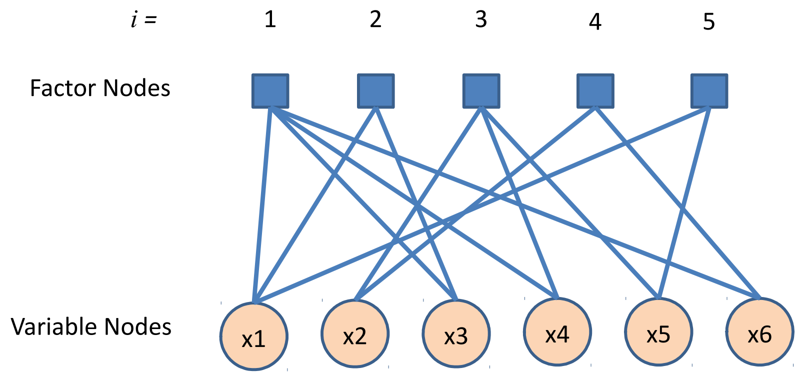
$$P(\mathbf{x}) = \frac{1}{Z}\prod_{i=1}^{N_f}\psi_i(\chi_i)$$

- We can build a graphical representation to capture this. It is called a factor graph.

Building a Factor Graph

$$P(\mathbf{x}) = \frac{1}{Z}\prod_{i=1}^{N_f}\psi_i(\chi_i)$$

$$\begin{aligned}\chi_1 &= (x_1, x_3, x_4, x_6)\\ \chi_2 &= (x_1, x_3)\\ \chi_3 &= (x_2, x_4, x_5)\\ \chi_4 &= (x_2, x_6)\\ \chi_5 &= (x_5, x_1)\end{aligned}$$



Factor Graphs

- Factor graphs are the natural representation for factorial decompositions of probability distributions.
- These sorts of representations are used in many settings we will encounter later.
- Often making factorising assumptions make parameterising a model feasible.

Markov Network

- A Markov network is an undirected graphical model.
- It is built from a factor graph by linking together all the nodes in each factor.
- A link in a factor graph encodes a direct dependence:
  - An edge i -- j is missing implies $I(x_i, x_j | \mathbf{x}_{\setminus\{i,j\}})$
  - The second term denotes all the *x* variables apart from the *i*th and *j*th term.

Terminology: a Markov Random Field is a Markov Network.

Examples

- Can you think of some situations when factor graphs or Markov networks are a natural representation?

Stop Point

- Questions?

Separation

- Conditional independence in Markov networks is much simpler.
- Separation Method:
  - Consider I(A,B|C).
  - Remove all the nodes C from the graph, and all links connecting node C.
  - If there is now no path from A to B, the independence relationship holds.
- Can you verify this using the probability distribution?
- The same approach applies to factor graphs.

Conversion

- Converting between network types.
- We have covered Factor Graph →Markov Network by construction.
- But we can convert between other graph types.
- Markov Network →Factor Graph
  - Find all maximal cliques in network. Make one factor per maximal clique.
- Note this is not necessarily minimal.
- Can you think of a Factor Graph s.t.

FG → MN → FG

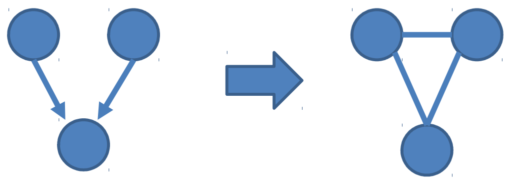
returns something different from what you started with?

Converting Belief Networks

- Question:
- How would you convert a belief network to a factor graph?

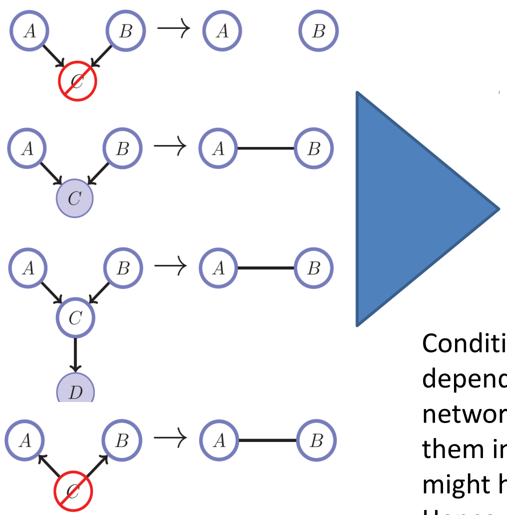
Converting Belief Networks

- Markov Networks to Belief Networks is hard. We will not cover that here. (In fact to do this solves inference? Can you see why?)
- Belief Networks → Markov Networks?
  - Marry Parents…
  - Why? → Next slide.



Conditional Independence Revisited

- From Barber (red circle implies marginalising – summing out, blue filled is conditioning). We only consider the link between A and B.



Sometimes A and B are directly dependent under conditioning.

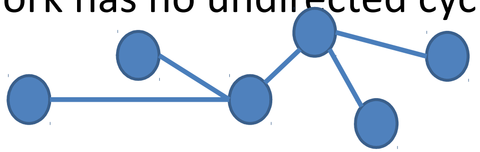
Conditioning cannot create dependencies in a Markov network. So we have to put them in explicitly where this might happen.
Hence marrying parents.

Stop Point

- Questions?

Tree Structured Networks

- An undirected network is a *tree* if the network has no undirected cycles. (*)



- The simplest type of tree is a chain

- Trees are special. Trees are easy to do inference with…

* Strictly it is a polytree – there could be many isolated trees. WLOG any method that applies to trees also applies to polytrees by applying it to each tree in the polytree. We simply refer to trees henceforth without further worry about this matter.

Where are we going?

- We have talked about
  - Probability distributions
  - Decomposing distributions into structure and probability values
  - Representing structure graphically is natural
  - Specifying structure from prior causal knowledge
  - The efficiency of specifying structure
  - Converting between graphical structures
  - Understanding independence from graphical structures
- We have said next to nothing about
  - What the graphical structure enables us to do
  - What sorts of problems we need to address with probabilistic models
    - Inference
    - Learning
  - How we can get the numbers in our models.

What is Inference?

- If we *know* a probabilistic model for something, how do we use it?
- Usually we ask questions we care about.
- They take the form
  - If THIS and THAT happen then what might happen to THOSE?
  - This is a conditional probability

$$P(\mathrm{THOSE}|\mathrm{THIS},\mathrm{THAT}) = \sum_{\mathrm{OTHER}} P(\mathrm{THOSE},\mathrm{OTHER}|\mathrm{THIS},\mathrm{THAT})$$

- Given a probabilistic model (which represents a joint distribution), what we want out are marginalised conditional distributions.
- Finding these is called inference.

Remember: Marginalising is summing out over unwanted variables

Why is Inference Hard?

- In a general distribution, when we condition or marginalise a variable we have to worry about the effects on every combination of all the other variables.
- We can do inference by *enumeration* of all possibilities
- This becomes infeasible in large systems.
- *Sometimes* graphical models help.

- Aside: computing probabilities over many variables is also problematic, as finding the normalisation constant is costly.
  - We leave this issue for later. Here we assume we want distributions over a small number of variables (we can condition on as many variables as we want).

Inference in Markov Networks

- Consider marginalisation and conditioning operations on a tree.
- Conditioning
  - Look at all neighbours. Replace factors at all neighbours to be conditional factors. This is called *absorbing*.
- Marginalising
  - Find all the factors containing the node to be marginalised. Replace all these factors with one big factor produced by marginalising over those factors only.
  - All other factors stay the same.
- This is the basis of the elimination algorithm.

Written Example

Sum-Products

- Sum distribution in sum-products

$$
\begin{aligned}
P(x_1, x_2, x_4, x_5) &= \sum_{x_3} \frac{1}{Z} \psi_1(x_1, x_4)\psi_2(x_1, x_5)\psi_3(x_2, x_3)\psi_4(x_3, x_5) \\
&= \frac{1}{Z} \psi_1(x_1, x_4)\psi_2(x_1, x_5) \sum_{x_3} \psi_3(x_2, x_3)\psi_4(x_3, x_5) \\
&= \frac{1}{Z} \psi_1(x_1, x_4)\psi_2(x_1, x_5)\psi_*(x_2, x_5)
\end{aligned}
$$

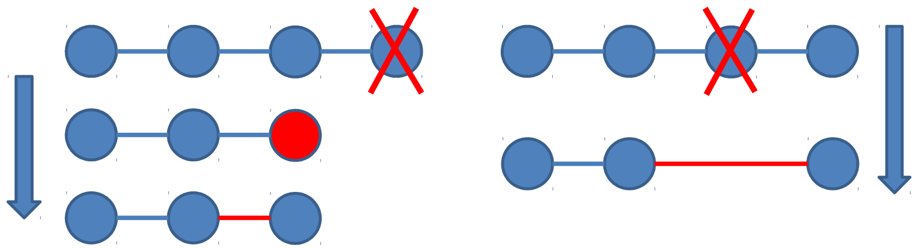
where

$$\psi_*(x_2, x_5) = \sum_{x_3} \psi_3(x_2, x_3)\psi_4(x_3, x_5)$$

- Order matters. Do cheap eliminations first.

Elimination Algorithm in Chains

- Consider Chains
  - If we eliminate from the ends of the chain, then it is cheap: results in a factor over one variable.
  - If we eliminate from the middle of the chain then it is cheap: results in a new link in the chain.



Elimination in Trees

- Suppose we want the marginal distribution at one node. (Conditioned nodes have been absorbed.)
- Any node of an undirected tree can be viewed as the root. Make this the node you care about.
- Use elimination from *leaves* of the tree.
  - Just like the chain
  - Each step produces a subtree.
  - Eventually just left with one node: the root.
  - Have marginal distribution for this node.

The Elimination Algorithm

We give the elimination algorithm for factor graphs:

1. Specify three sets: Evidential nodes (the nodes to be conditioned on), query nodes (the nodes we want a distribution over), latent nodes (the nodes we want to marginalise over).
2. Absorb the conditioned nodes into the existing factors.
3. Choose an elimination ordering for the latent nodes.
4. For each latent node in order above
   - Find all connected factors, and compute the product.
   - Sum out over the latent node from that product to get a new factor.
   - Replace all the previous factor nodes with a new node with that new factor.
5. We are left with factors over the nodes we care about. Enumerate these and normalise to get the probabilities.

Question

- But what if we want to compute more than one distribution at once? E.g. we want the marginal at every node in the network.
- But what if the network is not a tree?
- More next lecture!

Summary

- Belief Networks, Markov Networks, Factor Graphs.
- Read and work through Chapters 4 and 5 of Barber.
- There is much more detail there which is important for understanding.