Markov networks

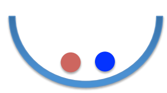
CS B553
Spring 2013

Announcements

- Assignment 2 posted!
  - Implement a Part-of-Speech tagger
  - With Bayes nets and variable elimination

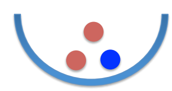
Problem 5, from Homework 1

- This question was really about a very interesting probability construct, the *Polya Urn*
  - Start with an urn with R red marbles and B blue marbles
  - In every time step, draw a marble at random
  - Replace the marble, and then also add a second marble of the same color



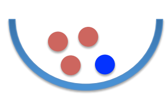
Problem 5, from Homework 1

- This question was really about a very interesting probability construct, the *Polya Urn*
  - Start with an urn with R red marbles and B blue marbles
  - In every time step, draw a marble at random
  - Replace the marble, and then also add a second marble of the same color



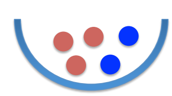
Problem 5, from Homework 1

- This question was really about a very interesting probability construct, the *Polya Urn*
  - Start with an urn with R red marbles and B blue marbles
  - In every time step, draw a marble at random
  - Replace the marble, and then also add a second marble of the same color



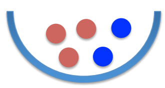
Problem 5, from Homework 1

- This question was really about a very interesting probability construct, the *Polya Urn*
  - Start with an urn with R red marbles and B blue marbles
  - In every time step, draw a marble at random
  - Replace the marble, and then also add a second marble of the same color



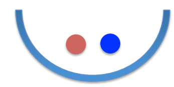
Problem 5, from Homework 1

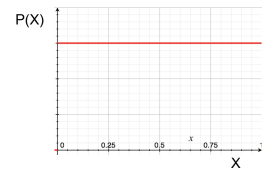
- This question was really about a very interesting probability construct, the *Polya Urn*
  - Start with an urn with R red marbles and B blue marbles
  - In every time step, draw a marble at random
  - Replace the marble, and then also add a second marble of the same color
- Polya Urns can model real-world *preferential attachment* phenomena



Polya urn distributions at the limit

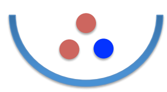
- Suppose you start with 1 red marble and 1 blue marble
  - Let X denote the fraction of red marbles after running this experiment for a very long time
  - How is X distributed?

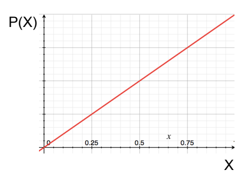




Polya urn distributions at the limit

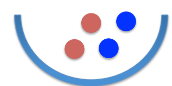
- Suppose you start with 2 red marbles and 1 blue marble
  - Let X denote the fraction of red marbles after running this experiment for a very long time
  - How is X distributed?

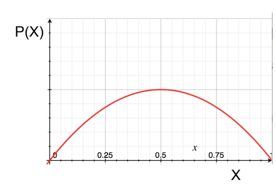




Polya urn distributions at the limit

- Suppose you start with 2 red marbles and 2 blue marble
  - Let X denote the fraction of red marbles after running this experiment for a very long time
  - How is X distributed?

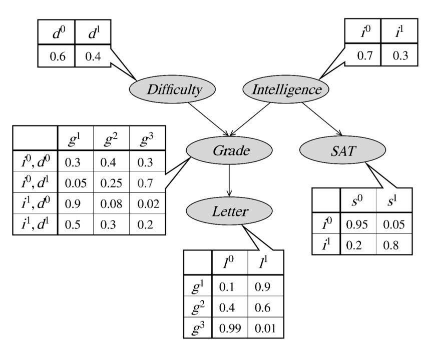




Special cases: Chains and polytrees

- For chains, we can always find an elimination ordering that takes time linear in the number of nodes
  - Start at the beginning of the chain and eliminate variables in node order
- A *polytree* is a dag such that there is at most one trail between every pair of nodes
  - In a polytree, it is always possible to find an elimination ordering that takes time linear in the *size of the conditional probability distributions*
  - E.g. start at leaves of tree and work upwards towards root(s)

Conditional probability distributions



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

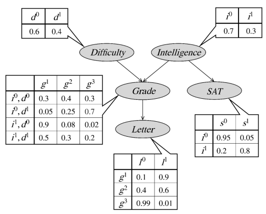
	g^1	g^2	g^3
i^0, d^0	0.3	0.4	0.3
i^0, d^1	0.05	0.25	0.7
i^1, d^0	0.9	0.08	0.02
i^1, d^1	0.5	0.3	0.2

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^3	0.99	0.01

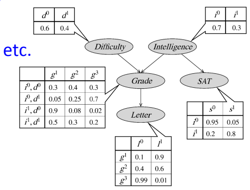
Conditional probability distributions

- There are various options for storing the CPDs in memory, but easiest is just a multi-dimensional array
  - We'll see others later on
- Where do the CPDs come from?
  - Set by hand, by intuition
  - Learned from data



Learning the CPDs

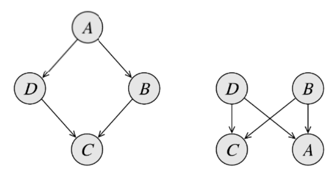
- Easy if we have a large amount of labeled training data
  - *Fully-supervised learning:* We have ground truth (correct) labels for all variables for all of our training exemplars
  - To learn, for example, P(Letter | Grade), we need to estimate the 6 entries in the CPD table
  - E.g. Simply look at all students for which Grade=A, calculate % of students where Letter is strong, etc.
- Harder case:
  - Weakly-supervised learning
  - We have labels for some but not all variables; we'll see this later!



Another example

- We have 4 people, Alice, Bob, Charles, and Dan
  - Alice and Bob, Bob and Charles, Charles and Dan, and Dan and Alice are friends
- Each person belongs to one of 2 political parties, given by random variables A, B, C, D
  - Friends are likely to belong to the same party
- We'd like to answer questions like,
  - E.g. "Supposing A and B are democrats, what's the probability that C is a republican?"
- How to model these variables as a Bayes Net?
  - What independence assumptions would we like?

Some possibilities…



- Independencies implied by left Bayes net:

$$A \perp C \mid D, B \qquad B \perp D \mid A \qquad B \not\perp D \mid A, C$$

- Independencies implied by right Bayes net:

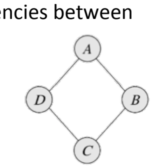
$$A \perp C \mid D, B \qquad B \perp D$$

Limitations of Bayes nets

- Bayes nets are useful for many problems, but simply cannot model certain sets of independence relations
- Also, Bayes nets require directionality of influences (e.g. causality)

Markov networks

- Markov networks model dependencies between variables as <u>undirected</u> graphs
  - Nodes represent random variables
  - Edges represent direct correlation between variables



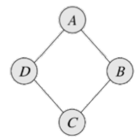
- Since dependencies are not directional, conditional probability distributions no longer make sense
  - Instead, we might want to model them using joint distributions, e.g. P(A,B)
  - For generality, we need not require these dependencies to even be probability distributions

Factors

- Markov networks model dependencies using *factors*
  - A generalization of probability distributions
  - A factor $\phi(\mathbf{X})$ for set of random variables **X** is just a function $\phi : \text{Val}(\mathbf{X}) \rightarrow \mathbb{R}$
  - The *scope* of $\phi(\mathbf{X})$ is the set of variables in **X**
- Probability distributions are a special case of factors
  - Factors can encode either joint and marginal probability distributions, or relationships that aren't probabilities at all
  - Factor values need not be in the range [0,1]

Factors as affinities

- Can view factors as "affinity scores," measuring the degree of compatibility between variable values



$\phi_1(A,B)$		
a^0	b^0	30
a^0	b^1	5
a^1	b^0	1
a^1	b^1	10

$\phi_2(B,C)$		
b^0	c^0	100
b^0	c^1	1
b^1	c^0	1
b^1	c^1	100

$\phi_3(C,D)$		
c^0	d^0	1
c^0	d^1	100
c^1	d^0	100
c^1	d^1	1

$\phi_4(D,A)$		
d^0	a^0	100
d^0	a^1	1
d^1	a^0	1
d^1	a^1	100

- We can write a joint probability distribution as,

$$P(a,b,c,d) = \frac{1}{Z}\phi_1(a,b)\cdot\phi_2(b,c)\cdot\phi_3(c,d)\cdot\phi_4(d,a)$$

  - Where Z is a normalizing constant,

$$Z = \sum_A\sum_B\sum_C\sum_D \phi_1(a,b)\cdot\phi_2(b,c)\cdot\phi_3(c,d)\cdot\phi_4(d,a)$$

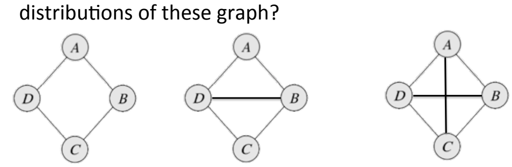
Independence of variables

- In a Markov network, for sets of random variables **X**, **Y**, and **Z**, $\mathbf{X} \perp \mathbf{Y} | \mathbf{Z}$ iff we can factor the joint probability distribution into a form like:

$$P(\mathbf{X},\mathbf{Y},\mathbf{Z}) = \phi_1(\mathbf{X},\mathbf{Z})\cdot\phi_2(\mathbf{Y},\mathbf{Z})$$

Factoring Markov networks

- How might we factor the joint probability distributions of these graph?



- Not simply a product over pairwise factors
- Instead, Markov networks factor over the *cliques* of the graph

Gibbs Distribution

- The joint distribution of a Markov network is given by a *Gibbs Distribution*,

$$P(\mathbf{X}) = P(X_1, ..., X_N) = \frac{1}{Z}\phi_1(\mathbf{A_1})\cdot\phi_2(\mathbf{A_2})\cdot ... \cdot\phi_N(\mathbf{A_N})$$

  - Where $\mathbf{A_1}, ..., \mathbf{A_n} \subseteq \mathbf{X}$

- A Gibbs distribution factors over a given Markov network *G* if each $\boldsymbol{A_i}$ is a clique of *G*

Markov network independence assumptions

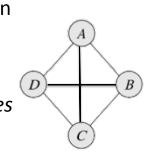
- Two variables that are *directly connected* are (potentially) *directly correlated* with one another
- Two variables X and Y that do not have an edge between them are independent conditioned on all other nodes in the graph, $X \perp Y | G - \{X, Y\}$
- A variable X is independent from all of its non-neighbors in the graph, conditioned on its neighbors

Markov network independence assumptions

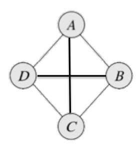
- In a Markov network, a path between variables X and Y given observed variables **Z** is *active* if the path does not traverse any node in **Z**
- A set of variables **Z** *separates* sets of variables **X** and **Y** iff there are no active paths between any variables in **X** and any variables in **Y**
- Then **X** and **Y** are independent conditioned on **Z**, $\mathbf{X} \perp \mathbf{Y} | \mathbf{Z}$, if and (almost) only if **Z** separates **X** and **Y**

Factorizations

- In general, what is the joint distribution for the graph at right?
- In some cases, e.g. social network, it's possible that the joint distribution *does* factor over e.g. edges of the graph
- We use a *factor graph* to explicitly encode the factorization



Factor graphs



- Two kinds of nodes
  - Random variable nodes (circles)
  - Factor graphs (squares)
  - Edges connect factor nodes and variable nodes
- Draw the factor graph for:

$$P(a,b,c,d) = \frac{1}{Z}\phi_1(a,b)\cdot\phi_2(b,c)\cdot\phi_3(c,d)\cdot\phi_4(d,a)\cdot\phi_5(a,c)\cdot\phi_6(b,d)$$

$$P(a,b,c,d) = \frac{1}{Z}\phi(a,b,c,d)$$

Log-linear models

- From the Gibbs distribution,

$$P(\mathbf{X}) = P(X_1,...,X_N) = \frac{1}{Z}\phi_1(\mathbf{A_1})\cdot\phi_2(\mathbf{A_2})\cdot ... \cdot\phi_N(\mathbf{A_N})$$

  - We can take logarithms,

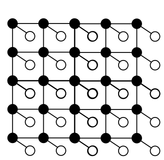
$$P(X_1,...,X_N) = \frac{1}{Z}\exp\left(\log\phi_1(\mathbf{A_1}) + \log\phi_2(\mathbf{A_2}) + ... + \log\phi_N(\mathbf{A_N})\right)$$

$$= \frac{1}{Z}\exp\left(-\sum_1^N f_i(\mathbf{A_i})\right)$$

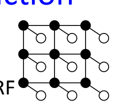
where $f_i(\mathbf{A_i}) = -\log\phi_i(\mathbf{A_i})$ is called an *energy function*

Pairwise Markov networks

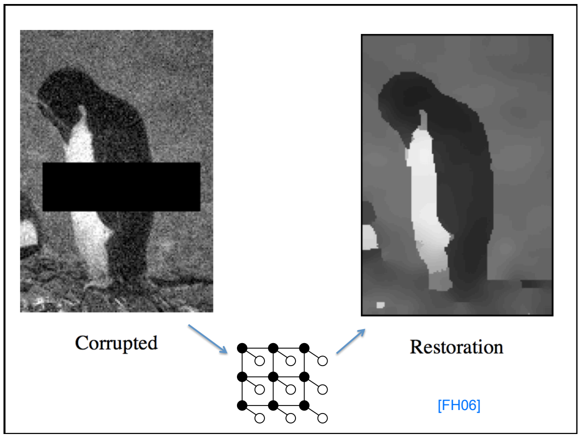
- In a pairwise Markov network (aka pairwise Markov Random Field or MRF), the max clique size is 2
  - Grid graphs are an especially popular special case



Application: Image reconstruction



- Given a noisy image, infer original image
- Express problem naturally in terms of an MRF
  - Image is stored as a sampled function on a grid
  - We can observe noisy pixel values, and we'd like to estimate the original, clean pixel values
  - Important constraint: In images of real-world scenes, one pixel's color is correlated with that of its neighbors
  - The pairwise factors model this constraint
- Problem can be solved by doing inference on the Markov network


Corrupted
Restoration
[FH06]