Generative models

- Bayesian networks, factors, Markov networks
- Applications / examples of Bayesian networks
- Bayesian inference

Outline

- **Bayesian networks, factors, Markov networks**
- Applications / examples of Bayesian networks
- Bayesian inference

Factors?

Markov networks:

$$\mathbb{P}(X_1 = x_1, \dots, X_n = x_n) = \frac{\text{Weight}(x)}{\sum_{x'} \text{Weight}(x')}$$

$$\text{Weight}(x) = \prod_{j=1}^{m} f_j(x)$$

What are these factors f_j's and how does one construct them?

- Hard constraints (CSPs): $f(x_1, x_2) = [x_1 \neq x_2]$
- Soft preferences (weighted CSPs, Markov networks): $f(x_1, x_2) = \{(\text{R}, \text{R}) : 1, (\text{R}, \text{B}) : 10, (\text{B}, \text{R}) : 10, (\text{B}, \text{B}) : 1\}$

Can we find a more natural/interpretable way to create factors...

Generative story: describe how the world works...

Generative story

Once upon a time, there was a girl who lived at $(0, 0)$.

One day, she wanted to to go visit her grandma's house at $(10, 10)$.

At every moment, she just chose a random direction that got her closer.

Once he reached her grandma's house, she stayed there and lived happily ever after.

Generative process

Generative process: Going to grandma's

$X_1 = (0, 0)$

for each time step $t = 2, \dots, T$:

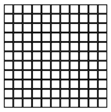
  Generate position $X_t \sim p(X_t \mid X_{t-1})$

Factors:

- $p(x_t \mid x_{t-1})$: uniform distribution over x_t which are (i) at most distance one away from x_{t-1} and (ii) at least as close to the goal

  Example:
  $p((0, 0) \mid (0, 0)) = p((1, 0) \mid (0, 0)) = p((0, 1) \mid (0, 0)) = \frac{1}{3}$

Running the generative process



Run

Generative model defines a distribution over complete paths.

Notational aside

"is proportional to":

- $p(x) \propto w(x) \Leftrightarrow p(x) = \frac{w(x)}{\sum_{x'} w(x')}$
- Example: $p(x, y) \propto [x < y]$ places uniform distribution over pairs (x, y) such that $x < y$

"is drawn/sampled/generated from":

- $X \sim f(X) \quad \Leftrightarrow \quad \mathbb{P}(X = x) = f(x)$

Markov models

Notes

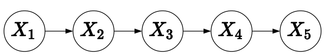
Generative process: Going to grandma's

$X_1 = (0, 0)$

for each time step $t = 2, \dots, T$:

  Generate position $X_t \sim p(X_t \mid X_{t-1})$

Bayesian network representation:



Variables are nodes, add directed edge to each variable from variables it depends on.

Factor graph representation

Generative process: Going to grandma's

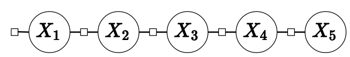
$X_1 = (0, 0)$

for each time step $t = 2, \dots, T$:

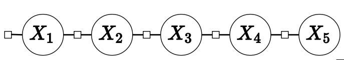
  Generate position $X_t \sim p(X_t \mid X_{t-1})$

Weight of assignment is product of factors:

$$\text{Weight}(x) = \prod_{t=1}^{T} p(x_t \mid x_{t-1})$$



Factor graph representation



$$\mathbb{P}(X = x) = \frac{\text{Weight}(x)}{\sum_{x'} \text{Weight}(x')} \qquad \text{Weight}(x) = \prod_{t=1}^{T} p(x_t \mid x_{t-1})$$

What is the normalization constant? Do variable elimination:

$f_4(x_4) = \sum_{x_5} p(x_5 \mid x_4) = 1$

$f_3(x_3) = \sum_{x_4} p(x_4 \mid x_3) f_4(x_4) = 1$

...

Important: normalization constant $\sum_{x'} \text{Weight}(x') = 1$

$$\mathbb{P}(X = x) = \text{Weight}(x) = \prod_{t=1}^{T} p(x_t \mid x_{t-1})$$

This holds for all generative models.

Generative models summary

Described by a generative process (randomized program): convenient representation for **modeling**

Example:

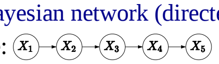
Generative process: Going to grandma's

$X_1 = (0, 0)$

for each time step $t = 2, \dots, T$:

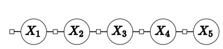
  Generate position $X_t \sim p(X_t \mid X_{t-1})$

Defines a Bayesian network (directed graphical model):

Example: 

Defines a Markov network (undirected graphical model / factor graph): convenient representation for **inference**

Example: $\mathbb{P}(X = x) = \text{Weight}(x) = \prod_{t=1}^{T} p(x_t \mid x_{t-1})$



Outline

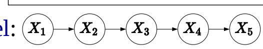
- Bayesian networks, factors, Markov networks
- **Applications / examples of Bayesian networks**
- Bayesian inference

General approach: model the world via a set of variables and how they were generated (don't solve the problem directly)...

Notes

Application: language modeling

Generative process: n-gram model
For each word position $t = 1, 2, \dots, T$:
Generate word $X_t \sim p(X_t \mid X_{t-n+1:t-1})$

Bigram model: 

$$\mathbb{P}(X = x) = \prod_{t=1}^{T} p(x_t \mid x_{t-n+1:t-1})$$: fluency of sentence x.

Applications:

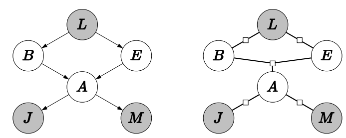
- Generate random stories
- Machine translation: make sure output looks like English
- Speech recognition: "I recognize speech" versus "I wreck a nice beach"

Notes

Application: reasoning under uncertainty

Question: At the office, you get a call from your neighbor John. What's the probability that there was a burglary?

L: location
B: burglary happened?
E: there was an earthquake?
A: alarm went off?
J: John called?
M: Mary called?



$$\mathbb{P}(L = l, B = b, E = e, A = a, J = j, M = m) = p(l)p(b \mid l)p(e \mid l)p(a \mid b, e)p(j \mid a)p(m \mid a)$$

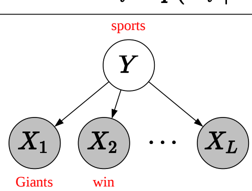
[Demo of simplified network]

- Allows principled incorporation of diverse long-range evidence
- Account for natural reasoning patterns such as explaining away

Application: document classification

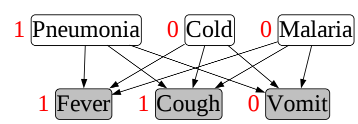
Question: given a text document, what is it about?

Generative process: Naive Bayes
Generate the document category $Y \sim p(Y)$
For each word index $i = 1, \dots, L$:
Generate word $X_i \sim p(X_i \mid Y)$



Application: medical diagostics

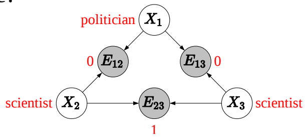
Question: If patient has has a cough and fever, what disease(s) does he/she have?



Generative process: Diseases and symptoms
For each disease $i = 1, \dots, m$:
Generate whether patient has that disease $X_i \sim p(X_i)$
For each symptom $j = 1, \dots, n$:
Generate whether patient has that symptom $Y_j \sim p(Y_j \mid X_{1:m})$

Application: social network analysis

Question: Given a social network (graph over n people), what types of people are there?



Generative process: Stochastic block model

For each person $i = 1, \dots, n$:

Generate the type of that person $X_i \sim p(X_i)$

For each pair of people $i \neq j$:

Generate whether i and j are linked $E_{ij} \sim p(E_{ij} \mid X_i, X_j)$

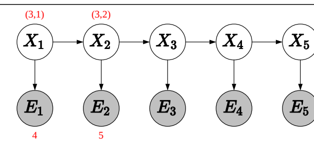
Application: object tracking

Generative process: Hidden Markov model

For each time step $t = 1, \dots, T$:

Generate object location $X_t \sim p(X_t \mid X_{t-1})$

Generate sensor reading $E_t \sim p(E_t \mid X_t)$



Other applications:

- Computational biology: gene finding
- Natural language: speech recognition, information extraction

Application: tracking multiple objects

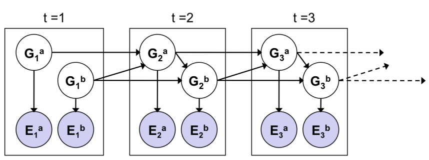
Generative process: Dynamic Bayesian network

For each time step $t = 1, \dots, T$:

For each object $o \in \{a, b\}$:

Generate location $G_t^o \sim p(G_t^o \mid G_{t-1})$

Generate sensor reading $E_t^o \sim p(E_t^o \mid G_t^o)$

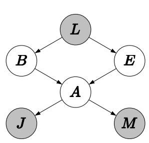


Outline

- Bayesian networks, factors, Markov networks
- Applications / examples of Bayesian networks
- **Bayesian inference**

General framework: Bayesian networks

Notes



- Coined by Judea Pearl in 1985, popularized probability in AI
- Variables $X_1, \dots, X_n$
- For each variable X_i, have a local conditional probability distribution $p(x_i \mid x_{\text{Parents}(i)})$:

$$\mathbb{P}(X = x) = \prod_{i=1}^{n} p(x_i \mid x_{\text{Parents}(i)})$$

Bayesian inference

Generative models so far have two types of variables:

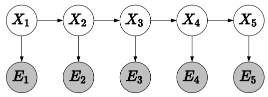
X: hidden variables describing the world (e.g., actual positions)
E: variables you observe about the world (e.g., sensor readings)

General approach: Define a generative process over both hidden and observed variables ("how the observed values came to be").

$$\mathbb{P}(X = x, E = e) = p_{\text{prior}}(x) \cdot p_{\text{likelihood}}(e \mid x)$$

Objective: compute **posterior** $\mathbb{P}(X_i = x_i \mid E = e)$ for some query variable $X_i \subset X$?

Bayesian inference



Objective: compute is the **posterior** $\mathbb{P}(X_i = x_i \mid E = e)$?

Algorithm:

- Condition on $E = e$ and produce a factor graph.
- Run inference algorithm (sum variable elimination, Gibbs sampling, or particle filtering).

Summary

- Markov networks: factors specify **local** dependencies, $\textbf{Weight}(x)$ induces **global** dependencies (analogy with rewards and utilities); normalize to get $\mathbb{P}(x = x)$.
- Inference algorithm: compute distribution over query variables taking into account all global dependencies
- Bayesian networks / generative processes: randomized program that assigns values to variables (think of a story of how the observed variables got there via hidden dynamics in the world); defines factor graph where factors are local probability distributions
- Bayesian inference: observe $E = e$, play detective to get $\mathbb{P}(X = x \mid E = e)$