The Normalization Shortcut

$P(B \mid j, m)$ stands for the probability distribution of $B$ given that $J = j$ and $M = m$

By definition $P(B \mid j, m) = P(B, j, m) / P(j, m)$, so

letting $\alpha = (P(j, m))$ lets us write:

$P(B \mid j, m) = \alpha P(B, j, m)$

Why? Because we don’t have to calculate $P(j, m)$ explicitly!

By the laws of probability $P(b \mid j, m) + P(\neg b \mid j, m) = 1$, so

$\alpha P(b, j, m) + \alpha P(\neg b, j, m) = 1$

$\alpha = 1 / (P(b, j, m) + P(\neg b, j, m))$

In general: $\alpha$ means “make distribution sum to 1”
Markov Chain Monte Carlo

- Fix the values of observed variables
- Set the values of all non-observed variables randomly
- Perform a random walk through the space of complete variable assignments. On each move:
  1. Pick a variable X
  2. Calculate Pr(X|true | all other variables)
  3. Set X to true with that probability
  4. Repeat many times. Frequency with which any variable X is true is its posterior probability.
  5. Converges to true posterior when frequencies stop changing significantly
     - stable distribution, mixing

CSE 592

MCMC with Gibbs Sampling

- Fix the values of observed variables
- Set the values of all non-observed variables randomly
- Perform a random walk through the space of complete variable assignments. On each move:
  1. Pick a variable X
  2. Calculate Pr(X|true | all other variables)
  3. Set X to true with that probability
  4. Repeat many times. Frequency with which any variable X is true is its posterior probability.
  5. Converges to true posterior when frequencies stop changing significantly
     - stable distribution, mixing

CSE 592
Markov Blanket Sampling

How to calculate \( Pr(X=true | \text{all other variables}) \)?
Recall: a variable is independent of all others given it's
Markov Blanket
- parents
- children
- other parents of children
So problem becomes calculating \( Pr(X=true | \text{MB}(X)) \)
- We solve this sub-problem exactly
- Fortunately, it is easy to solve

\[
P(X) = \alpha \prod_{\text{Parents}(X)} \prod_{\text{Children}(X)} Pr(Y | \text{Parents}(Y))
\]

Example

\[
P(X) = \alpha \prod_{\text{Parents}(X)} \prod_{\text{Children}(X)} Pr(Y | \text{Parents}(Y))
\]

\[
P(X | A, B, C) = \frac{P(X, A, B, C)}{P(A, B, C)}
\]

\[
P(A)P(X | A)P(B | X, C) = P(A, B, C)
\]

\[
P(X | A) = \frac{P(X | A)P(B | X, C)}{P(A, B, C)}
\]

Example

\[
\begin{align*}
P(x) &= P(s) \times P(h | s) \times P(b | h, l) \\
&= 0.2 \times 0.6 \times 0.9 = 0.108
\end{align*}
\]

\[
\begin{align*}
P(\neg x) &= P(s) \times P(h | s) \times P(b | h, l) \\
&= 0.2 \times 0.6 \times 0.1 = 0.012
\end{align*}
\]

Example 2

\[
\begin{align*}
P(x) &= P(s) \times P(h | s) \times P(b | h, l) \\
&= 0.2 \times 0.9 \times 0.7 = 0.162
\end{align*}
\]

\[
\begin{align*}
P(\neg x) &= P(s) \times P(h | s) \times P(b | h, l) \\
&= 0.2 \times 0.1 \times 0.3 = 0.006
\end{align*}
\]

Example 3

\[
\begin{align*}
P(x) &= P(s) \times P(h | s) \times P(b | h, l) \\
&= 0.2 \times 0.9 \times 0.7 = 0.147
\end{align*}
\]

\[
\begin{align*}
P(\neg x) &= P(s) \times P(h | s) \times P(b | h, l) \\
&= 0.2 \times 0.1 \times 0.3 = 0.012
\end{align*}
\]

Example 4

\[
\begin{align*}
P(x) &= P(s) \times P(h | s) \times P(b | h, l) \\
&= 0.2 \times 0.9 \times 0.7 = 0.147
\end{align*}
\]

\[
\begin{align*}
P(\neg x) &= P(s) \times P(h | s) \times P(b | h, l) \\
&= 0.2 \times 0.1 \times 0.3 = 0.012
\end{align*}
\]
### Example 5: Different Evidence

Evidence: S=true, B=false

- Smoking: 0.2
- Heart disease: 0.8
- Lung disease: 0.1
- Shortness of breath (S): 0.1 (F), 0.8 (T), 0.7 (F), 0.1 (T)

### Example 6

Evidence: S=true, B=false

- Smoking: 0.2
- Heart disease: 0.8
- Lung disease: 0.1
- Shortness of breath (S): 0.1 (F), 0.8 (T), 0.7 (F), 0.1 (T)

### Example 7

Sample H:

\[ P(h|s, \neg h, \neg b) = \alpha P(h|s)P(\neg b|\neg h) = \alpha (0.6)(0.1) = 0.06 \]

\[ P(\neg h|s, \neg h, \neg b) = \alpha P(\neg h|s)P(\neg b|\neg h) = \alpha (0.4)(0.3) = 0.12 \]

Normalize: 0.06/(0.06+0.12)=0.33

Flip coin: H stays false (maybe)

### Example 8

Sample L:

\[ P(l|s, \neg h, \neg b) = \alpha P(l|s)P(\neg b|\neg h) = \alpha (0.8)(0.3) = 0.24 \]

\[ P(\neg l|s, \neg h, \neg b) = \alpha P(\neg l|s)P(\neg b|\neg h, \neg l) = \alpha (0.2)(0.9) = 0.18 \]

Normalize: 0.24/(0.24+0.18)=0.75

Flip coin: ...

### Summary

Exact inference by variable elimination:
- Polynomial on polymers, NP-hard on general graphs
- Space = time, very sensitive to topology

Approximate inference by LV, MCMC, and rejection sampling:
- LV does poorly when there is lots of (downstream) evidence
- LV, MCMC generally insensitive to topology
- Convergence can be very slow with probabilities close to 1 or 0
- Can handle arbitrary combinations of discrete and continuous variables

### Outline

- Time and uncertainty
- Inference: filtering, prediction, smoothing
- Hidden Markov models
  - Dynamic Bayesian networks
  - Particle filtering
**Time and uncertainty**

The world changes, we need to track and predict it.

Diabetes management vs vehicle diagnosis.

Basic idea: copy state and evidence variables for each time step

\( X_t \) = set of unobservable state variables at time \( t \)

\( E_t \) = set of observable evidence variables at time \( t \)

This assumes discrete time; step size depends on problem.

Notation: \( X_{a:t} = X_a, X_{a+1}, \ldots, X_{t-1}, X_t \)

---

**Markov processes (Markov chains)**

Construct a Bayes net for these variables: parents?

Markov assumption: \( X_t \) depends on bounded subset of \( X_{t-1} \)

First-order Markov process:

\[
P(X_t | X_{t-1}) = P(X_t | X_{t-2}, X_{t-1})
\]

Second-order Markov process:

\[
P(X_t | X_{t-1}, X_{t-2}) = P(X_t | X_{t-2}, X_{t-1})
\]

Sensor Markov assumption:

\[
P(E_t | X_{t-1}) = P(E_t | X_t)
\]

Stationary process: transition model \( P(X_t | X_{t-1}) \) and sensor model \( P(E_t | X_t) \) fixed for all \( t \)

---

**Example**

First-order Markov assumption not exactly true in real world!

Possible fixes:
1. Increase order of Markov process.
2. Assign states, e.g., add \( \text{Temp} \), \( \text{Pressure} \).

Example: robot motion.

Augment position and velocity with \( \text{Battery} \).

---

**Inference tasks**

Filtering: \( P(X_t | e_{1:t}) \)

belief state—input to the decision process of a rational agent

Prediction: \( P(X_{t+k} | e_{1:t}) \) for \( k > 0 \)

evaluation of possible action sequences;
like filtering without the evidence

Smoothing: \( P(X_t | e_{1:t}) \) for \( 0 \leq k < t \)

better estimate of past states, essential for learning

Most likely explanation: \( \arg \max_{X_{1:t}} P(X_{1:t} | e_{1:t}) \)

speech recognition, decoding with a noisy channel.

---

**DBNs vs. HMMs**

Every DBN is a single variable DBN; every discrete DBN is an HMM

\[
\begin{align*}
X & \rightarrow X_t \\
Y & \rightarrow Y_t \\
Z & \rightarrow Z_t
\end{align*}
\]

Sparse dependencies \( \Rightarrow \) exponentially fewer parameters;

\( e.g. \), 20 state variables, three parents each

DBN has \( 20 \times 2^2 = 80 \) parameters.
HMM has \( 2^{20} \times 2^3 \approx 10^{16} \)

---

**Exact inference in DBNs**

Naive method: unroll the network and run any exact algorithm

Problem: inference cost for each update grows with \( t \)

Rollup filtering: add slice \( t+1 \), “sum out” slice \( t \) using variable elimination

Largest factor is \( O(2^{t+1}) \), update cost \( O(2^{t+1}) \)

(cf. HMM update cost \( O(2^t) \))
The Location Stack:
Design and Sensor-Fusion for Location-Aware Ubicomp

Jeffrey Hightower

The Location Stack

5 Principles
1. There are fundamental measurement techniques.
2. There are standard ways to combine measurements.
3. There are standard object relationship queries.
4. Applications are concerned with activities.
5. Uncertainty is important.

Principle 4: Applications are concerned with activities.

• Dinner is in progress.
• A presentation is going on in Mueller 153.
• Jeff is walking through his house listening to The Beatles.
• Jane is dispensing ethylene-glycol into beaker #45039.
• Elvis has left the building.

A survey & taxonomy of location technologies

[Highetower and Borriello, IEEE Computer, Aug 2001]
Principle 5: Uncertainty is important.
Example: routing phone calls to nearest handset

Fusion using Monte Carlo localization (MCL)
\[ \text{Bel}(x) = p(x \mid m_1 \ldots m_n) \]
\[ \text{Bel}(x) = \int p(m_1 \mid x) \prod p(x \mid x_{i-1}) \text{Bel}(x_{i-1}) dx_{i-1} \]

MCL details
Motion models:
\[ p(x_{t+1} \mid x_t) \]
Stochastically shift all particles
Sensor likelihood models:
\[ p(m_t \mid x_t) \]

2D MCL Example: Robocup
1. Object
2. Types of Measurements
   - Vision marker distance
   - Odometry
3. Red dot is most likely state.
   - (x, y, orientation)

Adaptive MCL
1. Performance improvement: adjust sample count to best represent the posterior.
   1. Assume we know the true \( \text{Bel}(x) \) represented as a multinomial distribution.
   2. Determine number of samples such that with probability \( (1-p) \), the Kullback-Leibler distance between the true posterior and the particle filter representation is less than \( \epsilon \)

Location Stack Implementation
Location Stack Supported Technologies

1. VersusTech commercial infrared badge proximity system
2. RF Proximity using the Berkeley motes
3. SICK LMS-200 180° infrared laser range finders
4. MIT Cricket ultrasound range beacons
5. Indoor harmonic radar, in progress
6. 802.11b WiFi triangulation system, in progress
7. Cellular telephone E-OTD, planned

Person Tracking with Anonymous and Id-Sensors: Motivation

- Accurate anonymous sensors exist
- Id-sensors are less accurate but provide explicit object identity information.

Person Tracking with Anonymous and Id-Sensors: Concept

- Use Rao-Blackwellised particle filters to efficiently estimate locations
  1. Each particle is an association history between Kalman filter object tracks and observations.
  2. Due to initial id uncertainty, starts by tracking using only anonymous sensors and estimating object id's with sufficient statistics.
  3. Once id estimates are certain enough, sample id them using a fully Rao-Blackwellised particle filter over both object tracks and id assignments.

Experimental Setup
Person Tracking with Anonymous and Id-Sensors: Result

- Our 2 phase Rao-Blackwellised particle filter algorithm is quite effective.

Conclusion

Relying on a single location technology to support all UbiComp applications is inappropriate. Instead, the Location Stack provides:

1. The ability to fuse measurements from many technologies including both anonymous and id-sensors while preserving sensor uncertainty models.
2. Design abstractions enabling system evolution as new sensor technologies are created.
3. A common vocabulary to partition the work and research problems appropriately.

Example Applications

- Spelling and grammar checkers
- Finding information on the WWW
- Spoken language control systems: banking, shopping
- Classification systems for messages, articles
- Machine translation tools

The Dream
Motivation and Outline

- Background
  - Definitions
- The Problem
  - 100,000+ pages
- The Solution
  - Ranking docs
  - Vector space
  - Probabilistic approaches
- Extensions
  - Relevance feedback, clustering, query expansion, etc.

What is Information Retrieval

- Given a large repository of documents, how do I get at the ones that I want
  - Examples: Lexus/Nexus, Medical reports, AltaVista
- Different from databases
  - Unstructured (or semi-structured) data
  - Information is (typically) text
  - Requests are (typically) word-based

Information Retrieval Task

- Start with a set of documents
- User specifies information need
  - Keyword query, Boolean expression, high-level description
- System returns a list of documents
  - Ordered according to relevance
- Known as the ad-hoc retrieval problem

Measuring Performance

- Precision \( \frac{tp}{tp + fp} \)
  - Proportion of selected items that are correct
- Recall \( \frac{tp}{tp + fn} \)
  - Proportion of target items that were selected
- Precision-Recall curve
  - Shows tradeoff

Basic IR System

- Use word overlap to determine relevance
  - Word overlap alone is inaccurate
- Rank documents by similarity to query
- Computed using Vector Space Model

Vector Space Model

- Represent documents as a matrix
  - Words are rows
  - Documents are columns
  - Cell \( i,j \) contains the number of times word \( i \) appears in document \( j \)
  - Similarity between two documents is the cosine of the angle between the vectors representing those words
Vector Space Example

- System and human system engineering
- A survey of user opinion of computer system response time
- The EPS user interface management system
- The generation of random, binary, ordered trees
- Graph minors IV: Widths of trees and well-quasi-ordering
- Graph minors: A survey

Vector Space Example cont.

Vector Space Example cont.

Similarity in Vector Space

\[ \cos(\theta_{ab}) = \frac{A \cdot B}{|A||B|} \]

Answering a Query Using Vector Space

- Represent query as vector
- Compute distances to all documents
- Rank according to distance
- Example: “computer system”

Common Improvements

- The vector space model
  - Doesn’t handle morphology (eat, eats, eating)
  - Favors common terms
- Possible fixes
  - Stemming
  - Convert each word to a common root form
  - Stop lists
  - Term weighting

Handling Common Terms

- Stop list
  - List of words to ignore
    - “a”, “and”, “but”, “to”, etc.
- Term weighting
  - Words which appear everywhere aren’t very good discriminators – give higher weight to rare words
tf * idf

\[ w_{ik} = tf_{ik} \cdot \log \left( \frac{N}{n_k} \right) \]

- \( w_{ik} \): weight of term \( k \) in document \( D_i \)
- \( tf_{ik} \): frequency of term \( k \) in document \( D_i \)
- \( idf_k \): inverse document frequency of term \( k \) in collection \( C \)
- \( N \): total number of documents in the collection \( C \)
- \( n_k \): number of documents in \( C \) that contain \( k \)

\[ idf_k = \log \left( \frac{N}{n_k} \right) \]

Inverse Document Frequency

- IDF provides high values for rare words and low values for common words.

For a collection of 10000 documents:

- \( \log \left( \frac{10000}{10000} \right) = 0 \)
- \( \log \left( \frac{10000}{5000} \right) = 0.301 \)
- \( \log \left( \frac{10000}{20} \right) = 2.698 \)
- \( \log \left( \frac{10000}{1} \right) = 4 \)

Probabilistic IR

- Vector space model robust in practice
- Mathematically ad-hoc
  - How to generalize to more complex queries? (intel or microsoft) and (not stock)
- Alternative approach: model problem as finding documents with highest probability of being relevant to the query
  - Requires making some simplifying assumptions about underlying probability distributions
  - In certain cases can be shown to yield same results as vector space model

Probability Ranking Principle

- For a given query \( Q \), find the documents \( D \) that maximize the odds that the document is relevant (R):

\[ \frac{P(r \mid D, Q)}{P(\neg r \mid D, Q)} = \frac{P(Q \mid D, r)}{P(Q \mid \neg D, r)} \frac{P(r \mid D)}{P(r \mid \neg D)} \]

Probability of document relevance to any query – i.e., the inherent quality of the document

But where do we get that number?
Bayesian nets for text retrieval

- Documents $d_1, d_2, \ldots$
- Words $w_1, w_2, \ldots$
- Concepts $c_1, c_2, \ldots$
- Query operators \((\text{AND/OR/NOT})\)
- Information need $q_0$

Bayesian nets for text retrieval

- Documents $d_1, d_2, \ldots$
- Words $w_1, w_2, \ldots$
- Concepts $c_1, c_2, \ldots$
- Query operators \((\text{AND/OR/NOT})\)
- Information need $q_0$

Conditional Probability Tables

- $P(d) =$ prior probability document $d$ is relevant
- Uniform model: $P(d) = 1 / \text{Number docs}$
- In general, document quality $P(r \mid d)$
- $P(w \mid d) =$ probability that a random word from document $d$ is $w$
  - Term frequency
- $P(c \mid w) =$ probability that a given document word $w$ has the same meaning as a query word $c$
  - Thesaurus
- $P(q \mid c_1, c_2, \ldots) =$ canonical form of operators AND, OR, NOT, etc.

Example

- Document Network
- Query Network

Details

- Set head $q_0$ of user query to “true”
- Compute posterior probability $P(D \mid q_0)$
- “User information need” doesn’t have to be a query - can be a user profile, e.g., other documents user has read
- Instead of just words, can include phrases, inter-document links
- Link matrices can be modified over time.
  - User feedback
  - The promise of “personalization”
Extensions

- Meet demands of web-based systems
- Modified ranking functions for the web
- Relevance feedback
- Query expansion
- Document clustering
- Latent Semantic Indexing
- Other IR tasks

IR on the Web

- Query AltaVista with “Java”
  - Almost 10^7 pages found
- Avoiding latency
  - User wants (initial) results fast
- Solution
  - Rank documents using word-overlap
  - Use special data structure - inverted index

Improved Ranking on the Web

- Not just arbitrary documents
- Can use HTML tags and other properties
  - Query term in <TITLE></TITLE>
  - Query term in <IMG>, <HREF>, etc. tag
  - Check date of document (prefer recent docs)
  - PageRank (Google)

PageRank

- Idea: Good pages link to other good pages
  - Round 1: count in-links
  - Round 2: sum weighted in-links
  - Round 3: and again, and again…
- Implementation: Repeated random walk on snapshot of the web
  - weight ≈ frequency visited

Relevance Feedback

- System returns initial set of documents
- User identifies relevant documents
- System refines query to get documents more like those identified by user
  - Add words common to relevant docs
  - Reposition query vector closer to relevant docs
- Lather, rinse, repeat…

Query Expansion

- Given query, add words to improve recall
  - Workaround for synonym problem
- Example
  - boat → boat OR ship
- Can involve user feedback or not
- Can use thesaurus or other online source
  - WordNet
Document Clustering

- Group similar documents
  - Similar means “close in vector space”
- If a document is relevant, return whole cluster
- Can be combined with relevance feedback
- GROUPER
  http://www.cs.washington.edu/research/clustering

Clustering Algorithms

- K-means
  Initialize k cluster centers
  Loop
  Assign all document to closest center
  Move cluster centers to better fit assignment
  Until little movement

- Hierarchical Agglomerative Clustering
  Initialize each document in a singleton cluster
  Loop
  Merge two closest clusters
  Until k clusters exist

Latent Semantic Indexing

- Creates modified vector space
- Captures transitive co-occurrence information
  - If docs A & B don’t share any words, with each other, but both share lots of words with doc C, then A & B will be considered similar
- Simulates query expansion and document clustering (sort of)

Variations on a Theme

- Text Categorization
  - Assign each document to a category
  - Example: automatically put web pages in Yahoo hierarchy
- Routing & Filtering
  - Match documents with users
  - Example: news service that allows subscribers to specify “send news about high-tech mergers”