Readings: K&F 19.2, 19.3

EM Algorithm & Learning Problems in Real Applications

Lecture 13 – May 9, 2011
CSE 515, Statistical Methods, Spring 2011

Instructor: Su-In Lee
University of Washington, Seattle

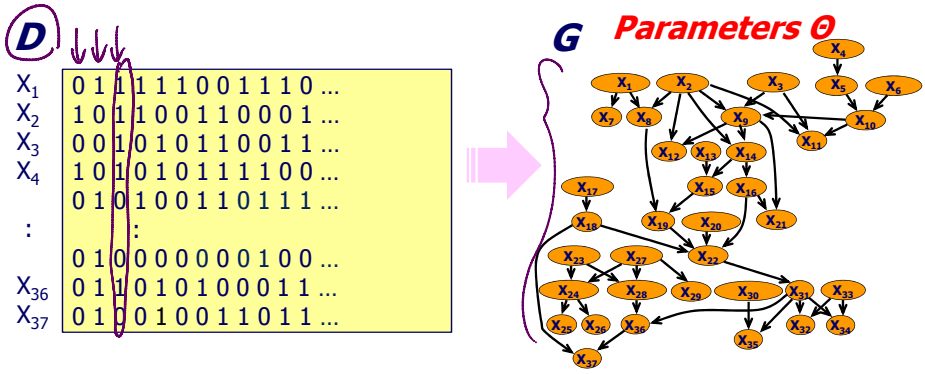
Outline

- Expectation Maximization (EM)
  - Parameter estimation with missing data
- Structural EM
  - Structure learning with missing data
- Learning problems in real world applications
  - Computational biology, natural language processing, robotics, collaborative filtering, etc
  - Student presenters: Nathan, John, Kris
- Mid-quarter review (1-1:20pm)
  - Jim Borgford-Parnell, Center for Engineering Learning and Teaching (CELT)

2

MLE in Bayesian Networks

- Recall that when learning from complete data,



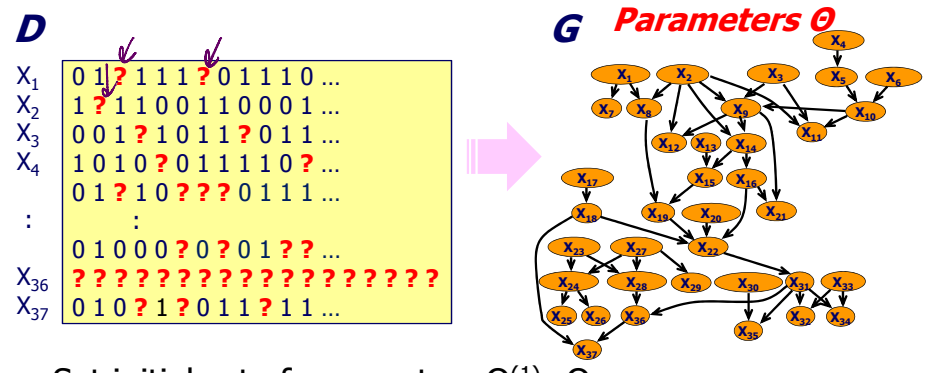
- We would first collect sufficient statistics for each CPD.
- We can estimate parameters that maximize the likelihood with respect to these statistics.
  - For table CPDs:

$$\theta^{MLE}_{X_i=x|\mathbf{Pa}_i=\mathbf{u}} = \frac{M[X_i = x, \mathbf{Pa}_i = \mathbf{u}]}{M[\mathbf{Pa}_i = \mathbf{u}]}$$

3

Expectation Maximization (EM)

- Parameter estimation when D has missing data



- Set initial set of parameters $\Theta^{(1)}=\Theta_0$
- Iterate E-step and M-step until convergence
  - In the t-th iteration,
  - E-step: Given $\Theta^{(t)}$, fill in missing values
  - M-step: Given complete data, learn parameters $\Theta^{(t+1)}$

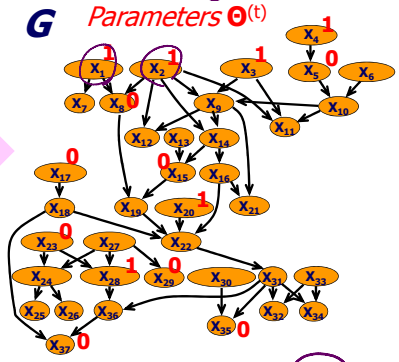
E-step

- Given parameters $\Theta^{(t)}$, fill in missing values

D For m'th training instance

X_1	0 1 **?** 1 1 1 **?** 0 1 1 1 0 …
X_2	1 **?** 1 1 0 0 1 1 0 0 0 1 …
X_3	0 0 1 **?** 1 0 1 1 **?** 0 1 1 …
X_4	1 0 1 0 **?** 0 1 1 1 1 0 **?** …
X_5	0 1 **?** 1 0 **?** **?** **?** 0 1 1 1 …
:	
X_{35}	0 1 0 0 0 **?** 0 **?** 0 1 **?** **?** …
X_{36}	**?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?**
X_{37}	0 1 0 **?** 1 **?** 0 1 1 **?** 1 1 …

G Parameters $\Theta^{(t)}$



Let o[m] be the data case in the m-th instance <?,1,1,…,0,?,0>

$P(X_1=1|o[m])=?$
$P(X_5=1|o[m])=?$
$P(X_{35}=1|o[m])=?$

For each X_i, $P(X_i=x, \mathbf{Pa}_i=\mathbf{u}|o[m])=?$
For each X_i, $P(\mathbf{Pa}_i=\mathbf{u}|o[m])=?$

Inference algorithms (e.g. clique tree algorithm)

5

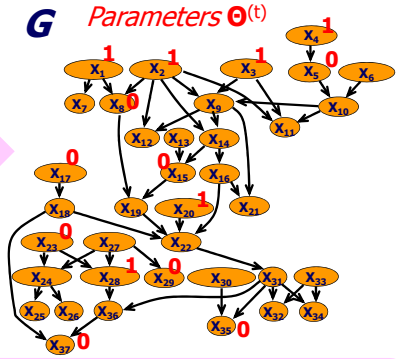
E-step – expected counts

- Given parameters $\Theta^{(t)}$, fill in missing values

D For m'th training instance

X_1	0 1 **?** 1 1 1 **?** 0 1 1 1 0 …
X_2	1 **?** 1 1 0 0 1 1 0 0 0 1 …
X_3	0 0 1 **?** 1 0 1 1 **?** 0 1 1 …
X_4	1 0 1 0 **?** 0 1 1 1 1 0 **?** …
X_5	0 1 **?** 1 0 **?** **?** **?** 0 1 1 1 …
:	
X_{35}	0 1 0 0 0 **?** 0 **?** 0 1 **?** **?** …
X_{36}	**?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?** **?**
X_{37}	0 1 0 **?** 1 **?** 0 1 1 **?** 1 1 …

G Parameters $\Theta^{(t)}$



Let o[m] be the data case in the m-th instance

$P(X_1=1|o[m])=?$
$P(X_5=1|o[m])=?$
$P(X_{35}=1|o[m])=?$

For each X_i, $P(X_i=x, \mathbf{Pa}_i=\mathbf{u}|o[m])=?$
For each X_i, $P(\mathbf{Pa}_i=\mathbf{u}|o[m])=?$

If $\mathbf{Pa}_i=1$ in o[m], $P(\mathbf{Pa}_i=1|o[m])=1$ and $P(\mathbf{Pa}_i=0|o[m])=0$
If $\mathbf{Pa}_i=?$ in o[m], $0 \le P(\mathbf{Pa}_i=1|o[m]) \le 1$

"Soft" (expected) counts

E-step cont. & M-step

- Compute sufficient statistics:
- If D is complete data,

$$M[X_i = x, \mathbf{Pa}_i = \mathbf{u}] = \sum_m \mathbf{1}(X_i = x, \mathbf{Pa}_i = \mathbf{u} \mid o[m])$$

- Given "soft" counts (expected sufficient statistics; ESS),

$$\overline{M}_{\theta^{(t)}}[X_i = x, \mathbf{Pa}_i = \mathbf{u}] = \sum_m P(X_i = x, \mathbf{Pa}_i = \mathbf{u} \mid o[m], \theta^{(t)})$$

$$\overline{M}_{\theta^{(t)}}[\mathbf{Pa}_i = \mathbf{u}] = \sum_m P(\mathbf{Pa}_i = \mathbf{u} \mid o[m], \theta^{(t)})$$

- M-step: Treat the ESS as those from complete data, and set the parameters to the MLE with respect to the ESS

$$\theta^{(t+1)}_{X_i = x \mid \mathbf{Pa}_i = \mathbf{u}} = \frac{\overline{M}_{\theta^{(t)}}[X_i = x, \mathbf{Pa}_i = \mathbf{u}]}{\overline{M}_{\theta^{(t)}}[\mathbf{Pa}_i = \mathbf{u}]}$$

7

Expectation Maximization (EM)

- Formal Guarantees:
  - $L(D:\Theta^{(t+1)}) \geq L(D:\Theta^{(t)})$
    - Each iteration improves the likelihood (read Andrew Ng's lecture note on EM)
  - If $\Theta^{(t+1)}=\Theta^{(t)}$, then $\Theta^{(t)}$ is a stationary point of $L(D:\Theta)$
    - Usually, this means a local maximum
- Main cost:
  - Computations of expected counts in E-Step
  - Requires inference for each instance in training set
    - Exactly the same as in gradient ascent!

8

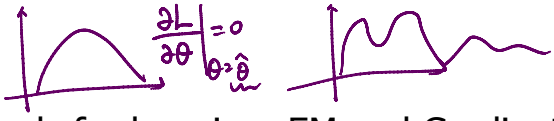
EM – Practical Considerations

- Initial parameters
  - Highly sensitive to starting parameters Θ_0
  - Choose randomly
  - Choose by guessing from another source
- Stopping criteria
  - Small change in data likelihood
  - Small change in parameters
- Avoiding bad local maxima
  - Multiple restarts
  - Early pruning of unpromising starting points

9

Partially Observed Data: Parameter Estimation

- Log-likelihood function is not concave



- Methods for learning: EM and Gradient Ascent
  - Exploit inference for learning
- Challenges
  - Exploration of a complex likelihood/posterior
    - More missing data ⇒ many more local maxima
  - Inference
    - Main computational bottleneck for learning
    - Learning large networks ⇒ exact inference is infeasible
      ⇒ resort to approximate inference

10

Structure Learning w. Missing Data

- Distinguish two learning problems
  - Learning structure for a given set of random variables
  - Introduce new hidden variables (K&F 19.5)
    - How do we recognize the need for a new variable?
    - Where do we introduce a newly added hidden variable within G?
    - Open ended and less understood…

Structure Learning w. Missing Data

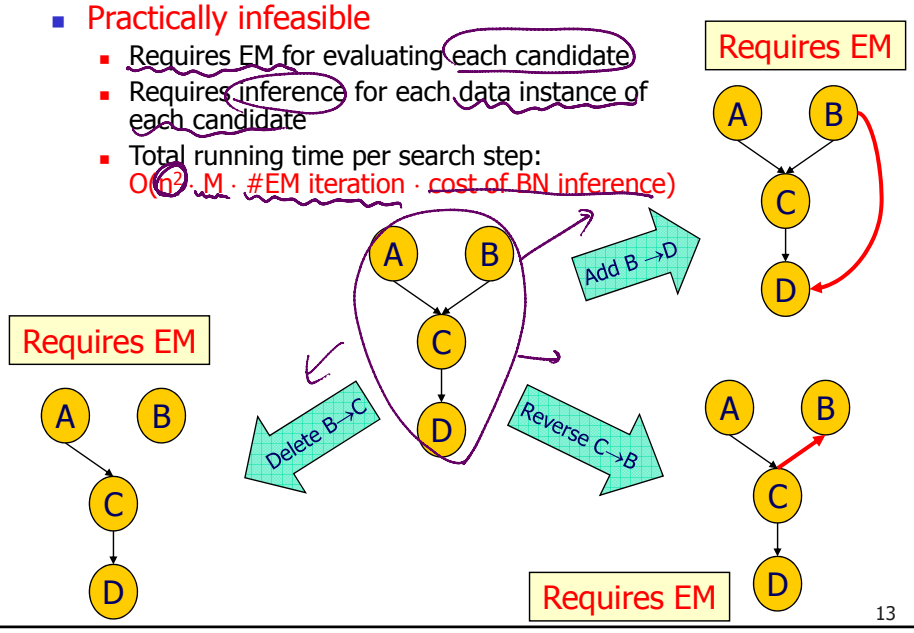
- Theoretically, there is no problem
  - Define score, and search for structure that maximizes it

$$Score_{BIC}(G:D) = l(\hat{\theta}_G : D) - \frac{\log M}{2} Dim(G)$$

  - Likelihood term will require gradient ascent or EM
- Practically infeasible
  - Typically we have $O(n^2)$ candidates at each search step

Typical Search

- Practically infeasible
  - Requires EM for evaluating each candidate
  - Requires inference for each data instance of each candidate
  - Total running time per search step: $O(n^2 \cdot M \cdot \#\text{EM iteration} \cdot \text{cost of BN inference})$



Structural EM

- Basic idea: use expected sufficient statistics to learn structure, not just parameters
  - Use current network to complete the data using EM
  - Treat the completed data as "real" to score candidate structures
  - Pick the candidate network with the best score
  - Use the previous completed counts to evaluate networks in the next step
  - After several steps, compute a new data completion from the current network

Structural EM

- Set the initial parameters $\Theta^{(1)}=\Theta_0$, structure $G^{(1)}=G_0$
- Iterate:
  - Given parameters/structure $\Theta^{(t)}$ and $G^{(t)}$, fill in missing values with estimates → D^*
  - Given the complete data D^*, learn structure $G^{(t+1)}$
  - Given the complete data D^* and structure $G^{(t+1)}$, learn parameters $\Theta^{(t+1)}$

Structural EM Benefits

- Many fewer EM runs
- Score relative to completed data is decomposable!
  - Utilize same benefits as structure learning w. complete data
  - Each candidate network requires fewer re-computations
  - Here savings is large since each sufficient statistics computation requires inference
- As in EM, we optimize a simpler score
- Can show improvements and convergence

$$Score_{BIC}(\langle G,\theta_G\rangle : D) - Score_{BIC}(\langle G^Q,\theta_G^Q\rangle : D) \geq 0$$

$$\geq E_Q\left[Score_{BIC}(\langle G,\theta_G\rangle : D^+)\right] - E_Q\left[Score_{BIC}(\langle G^Q,\theta_G^Q\rangle : D^+)\right]$$

D^+: Data imputed a few iterations ago

  - An SEM step that improves in D^+ space, improves real score

Where are we?

Week	Dates	Topics and Lecture Notes	Readings
I. Probabilistic Graphical Models Representation			
1	3/28	Introduction to the class	2.1, 2.2, 2.3
	3/30	Bayesian network representation	3.1, 3.2, 3.3
2	4/4	Local probability models	3.4, 5
	4/6	Undirected graphical models I	4.1, 4.2, 4.3
3	4/11	Undirected graphical models II + P-DAGs	4.4, 4.5, 4.6
II. Exact Inference			
	4/13	Inference: exact inference	9.1, 9.2, 9.3
4	4/18	Exact inference in BNs	9.4, 9.5, 9.6
	4/20	Exact inference: Clique Trees	10.1, 10.2, 10.3, 10.4
III. Learning			
5	4/25	Learning: parameter estimation	17
	4/27	Parameter learning in BNs	17
6	5/2	Structure learning in BNs	18
	5/4	Partially observed data (learning with missing data)	19
7	5/9	More on learning (TBD)	
IV. Approximate Inference			
	5/11	Approximate inference: particle-based I	12
8	5/16	Approximate inference: particle-based II	12
	5/18	Global approximate inference I	11
9	5/23	Global approximate inference II	11
V. Special Topics & Applications			
	5/25	Markov Decision Processes (Instructor: Mausam)	
10	5/30	(memorial day)	
	6/1	Temporal models (DBNs, HMMs)	
		Final examination @	

1. Probabilistic model **representation**

2. Exact inference in BNs
$P(\mathbf{X}=x|\mathbf{E}=e)=?$

3. Learning parameters/structure
- Learning CPDs, structure from data

4. Approximate inference
$P(\mathbf{X}=x|\mathbf{E}=e)=?$

5. Applications
- Decision making, temporal processes

17

LEARNING PROBLEMS IN REAL APPLICATIONS

Learning Problems in Real Applications

- Robotics, AI, Natural Language Processing, Computational Biology, Computer Vision…

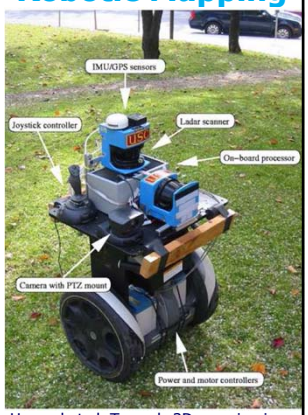
- Robotic mapping
- Collaborative filtering
- Part-of-speech tagging
  - Presented by Nathan Imse
- Peptide identification in MSMS
  - Presented by John Halloran
- Finding tumor-specific mutations
  - Presented by Kris Weber
- Text classification
  - Collective classification of web pages
- Computer vision
  - Image segmentation and de-noising

19

Background

- Robotic mapping: acquiring a spatial model of a robot's environment.
  (Sebastian Thrun, 2002)
  - Maps are commonly used for robot navigation (e.g. localization).
  - Robots must posses sensors (sonar, laser, radar, GPS, etc) to be able to perceive the outside world.
  - All sensors are subject to errors, often referred to as measurement noise.

Robotic Mapping



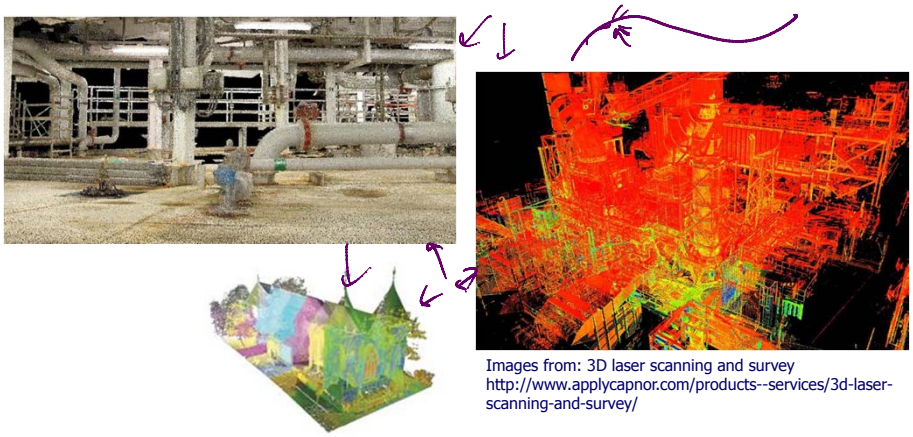
Howard et al. Towards 3D mapping in large urban environments, Intelligent Robots and Systems, 2004.

- Object maps
  - A family mapping algorithms addresses the problem of building maps composed of basic geometric shapes or objects, such as lines, walls and so on.
  - Let's focus on one by Thrun et al. (2004) that explicitly tries to use the probabilistic model to capture the structure in the environment.

20

Training Data

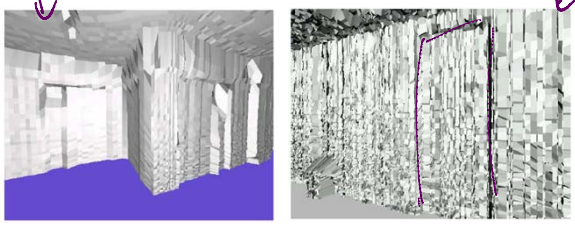
- Data: A point cloud representation of an indoor environment.
  - The point cloud can be obtained by collecting a sequence of point clouds, measured along a robot's motion trajectory.



Images from: 3D laser scanning and survey http://www.applycapnor.com/products--services/3d-laser-scanning-and-survey/

Problem Statement

- Goal: Take the points obtained over the trajectory and fit the points using polygons, to derive a 3D map of the surfaces in the robot's environment.
  - However, the noise in the laser measurements, combined with the errors in localization, leads adjacent polygons to have slightly different surface normals, giving rise to a very jagged representation of the environment.



Polygonal map generated from raw data. The display without texture shows the level of noise involved. In particular, it illustrates the difficulty of separating the door from the nearby wall. Thrun et al. (2004)

- Probabilistic model: Fit a more compact representation of the environment to the data, reducing the noise and providing a smoother, more realistic output.

Modeling

- The model consists of a set of 3D planes $p_1, \ldots, p_K$, each characterized by two parameters (α_k, β_k)
  - α_k : a unit-length vector in $\mathbf{R}^3$ that encodes the plane's surface normal vectors
  - β_k : a scalar that denotes its distance to the origin of the global coordinate system.

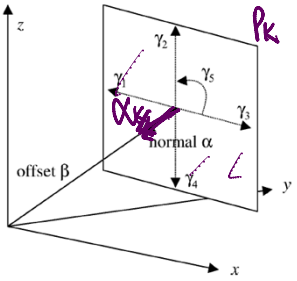


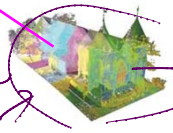
Illustration of the parameters in the planar surface model
Thrun et al. (2004)

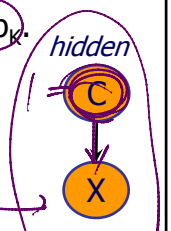
  - The distance of any point **x** to the plane is

$$d(\mathbf{x}, p_k) = |\alpha_k \mathbf{x} - \beta_k|$$

Probabilistic Model

- Each point x_m belongs to one of the planes $p_1,\ldots,p_K$.





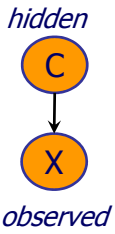
- This assignment can be modeled via a set of correspondence variables C_m
  - C_m=k if the measurement point x_m was generated by the k-th plane p_k.
  - C_m is not observed (hidden variable)

- Define $P(\mathbf{X}_m \mid C_m = k; \theta_k)$ to be $\propto N(d(\mathbf{x}, p_k); 0, \sigma^2)$
  - Allow an additional value C_m=0 that encodes points that are not generated by any of the planes
- Data: each point x_m is a training instance, e.g. x_m =(1,1,1)

Learning

Probabilistic model

- Define $P(\mathbf{X}_m \mid C_m = k : \theta_k)$ to be $\propto \mathrm{N}(d(\mathbf{x}, p_k) : 0, \sigma^2)$
  - Allow an additional value C_m=0 that encodes points that are not generated by any of the planes

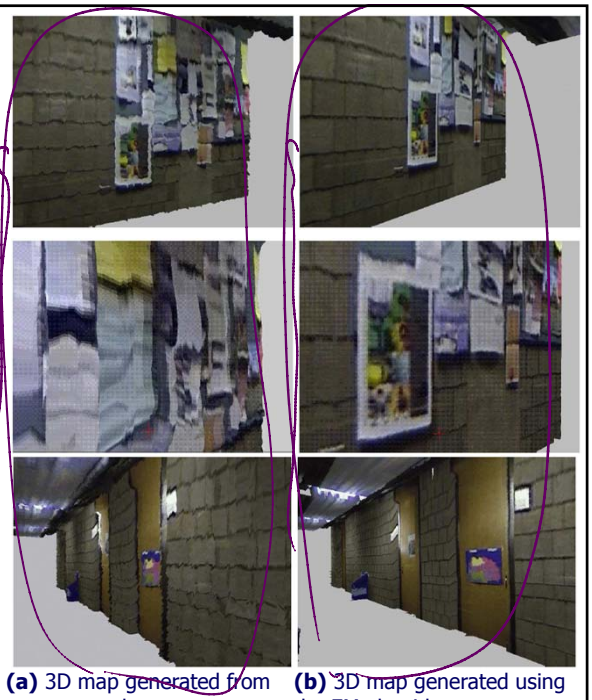


- Goal: the assignment of points to planes (C_m for all m's), the parameters (α_k, β_k) that characterize the planes.

- EM algorithm
  - E-step: computes the assignment to the correspondence variables C_m's by assigning the weight of each point proportionately to its distance to each of them.
  - M-step: recomputes the parameters of each plane to fit the points assigned to it.

Results I

- Notice that the map in (b) is smoother and appears to be visually more accurate than the one in (a)



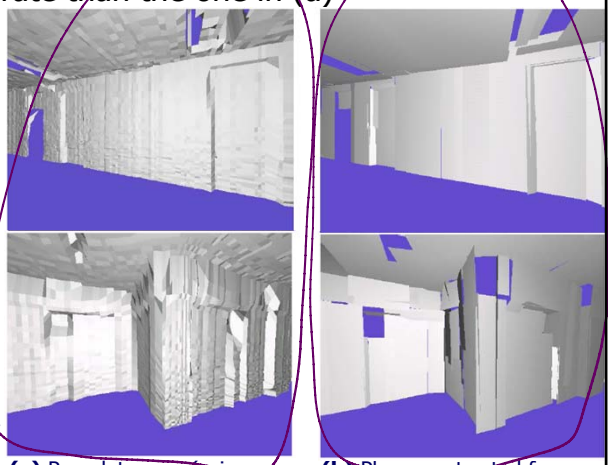
Thrun et al. *A Real-time EM Algorithm for Acquiring Multi-Planar Maps of Indoor Environments with Mobile Robots.* IEEE Transactions on Robotics and Automation, 2004.

(a) 3D map generated from raw sensor data

(b) 3D map generated using the EM algorithm

Results II

- Maps generated in real-time, of office environments
- Notice that the map in (b) is smoother and appears to be visually more accurate than the one in (a)



Thrun et al. *A Real-time EM Algorithm for Acquiring Multi-Planar Maps of Indoor Environments with Mobile Robots.* IEEE Transactions on Robotics and Automation, 2004.

(a) Raw data map (using a high-accuracy range finder)

(b) Planes, extracted from the map using EM

27

Learning Problems in Real Applications

- Robotics, AI, Natural Language Processing, Computational Biology, Computer Vision…

- Robotic mapping
- Collaborative filtering
- Part-of-speech tagging
  - Presented by Nathan Imse
- Peptide identification in MSMS
  - Presented by John Halloran
- Finding tumor-specific mutations
  - Presented by Kris Weber
- Text classification
  - Collective classification of web pages
- Computer vision
  - Image segmentation and de-noising

28

Hidden Markov Models for Part of Speech Tagging

Nathan Imse
njimse@uw.edu

What Is Part of Speech Tagging?

- Labeling words based on their function
  - Nouns: person, place, thing
  - Verb: action
  - Adjective: describes a noun
- Tagsets often have anywhere from 10-90 tags

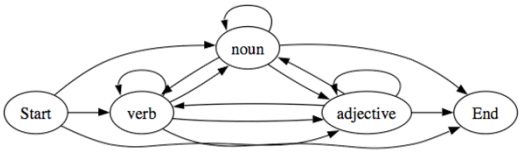
Nouns	Verbs	Adjectives
Cat, dog, ball, car, zoo, human, book	Book, chase, is, cook, live, sleep, sing	Red, big, heavy, human, short, cold, smelly

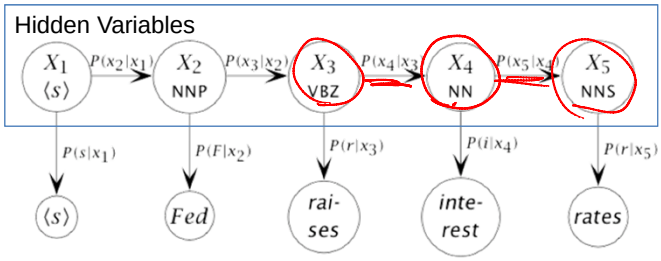
Why Tag the Part of Speech?

- Named Entity Recognition
  - Extract names of people, companies, and cities from text
- Word Sense Disambiguation
  - Differentiate between different meanings of the same word (e.g. bank, run, book)
- Parsing
  - Build a structure of the sentence with a much simpler grammar (<90 tags vs. 200,000+ words)

HMM Representation: Hidden Variables

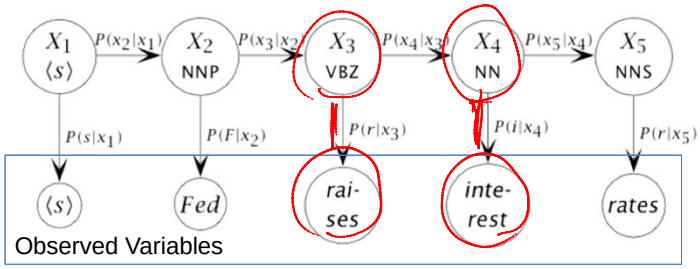
- Hidden variables are Part of Speech (POS) tags
- Edges between POS tags are the probability of one following the other





HMM Representation: Observed Variables

- Observed variables are words, numbers, punctuation, etc…
- Observed variables are leaves in the network
  - No outgoing edges
- Edge from hidden variable to observed variable is the probability of that observation given that hidden variable



Training and Testing

- Training data consists of annotated text, either in parse trees or in flat text
  - Pierre/NNP Vinken/NNP ,/, 61/CD years/NNS old/JJ ,/, will/MD join/VB the/DT board/NN ….
- Testing data does not have tags
  - "Fed raises interest rates"
  - Tags are considered "hidden"

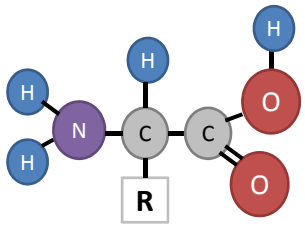
Decoding/Inference

- Viterbi Algorithm is common
  - Dynamic programming
  - Optimal global sequence, with probability
- Implemented for a course last fall
  - Trained on ~1900 sentences from the Wall Street Journal
  - 88% accuracy (with smoothing)

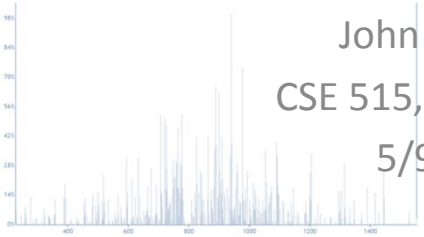
	Bigram	Trigram
States	47	2162
Observations	~7900	~7900
Transitions	~890	~99500
Emissions	~8700	~400800

Credit

The diagram of the HMM network as a Bayesian Network on slides 4 & 5 are the work of Andrew McCallum from Umass, Amherst



A Dynamic Bayesian Network for Peptide Identification in MSMS



John Halloran

CSE 515, Spring 2011

5/9/2011

Problem Statement

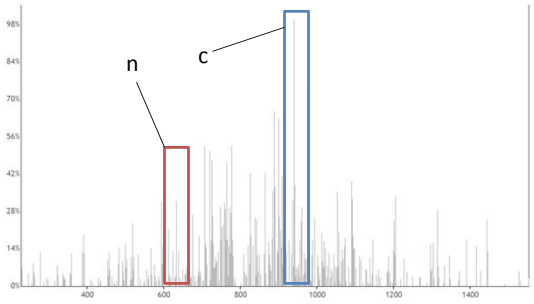
- Tandem mass spectrometry
  - Spectrum of measured m/z intensities generated from sample proteins (peptides)
- **Given MSMS spectrum, identify peptide which generated spectra**
- Typically look up peptides in database of the mapped organism

Model

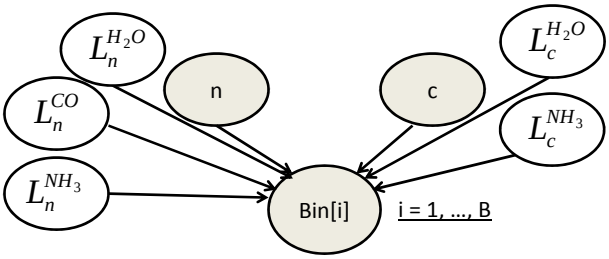
- Normalize spectra, quantize into B bins
- Model fragmentation event:
  - Candidate Peptide: IEQFMEEMYQDK $^{++}$

Fragment Ions
IEQFM^{+}
EEMYQDK^{+}

- Check n=mass(IEQFM^{+}), c=mass(EEMYQDK^{+}) bins values, calculate Gaussian likelihood ratio based on values



Model



- n, c, Bin[i] values are observed
- Loss of CO, NH_3, H_2O(?) - Losses are hidden variables (Bernoulli)
- Peptide length N, N-1 such events: concatenate to form DBN

Training

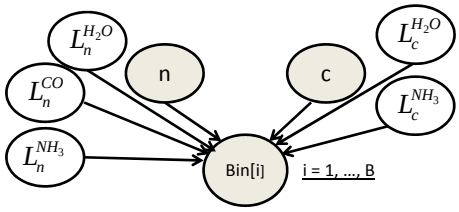
- Learned parameters:
  - Gaussian variances, neutral loss probabilities
- Training Data:
  - 1208 high quality E-coli matches
  - Matches generated combining 7 peptide-identification algorithms
- Parameters trained using EM

Inference

- Candidate peptide C and B bins assigned score of log(P(bins,C)) , where:

$$P(bins, C) = \prod_{i}^{n-1} \sum_{losses} p(loss) \prod_{j}^{B} p_I(b_j) \frac{p_A(b_n)}{p_I(b_n)} \frac{p_A(b_c)}{p_I(b_c)}$$

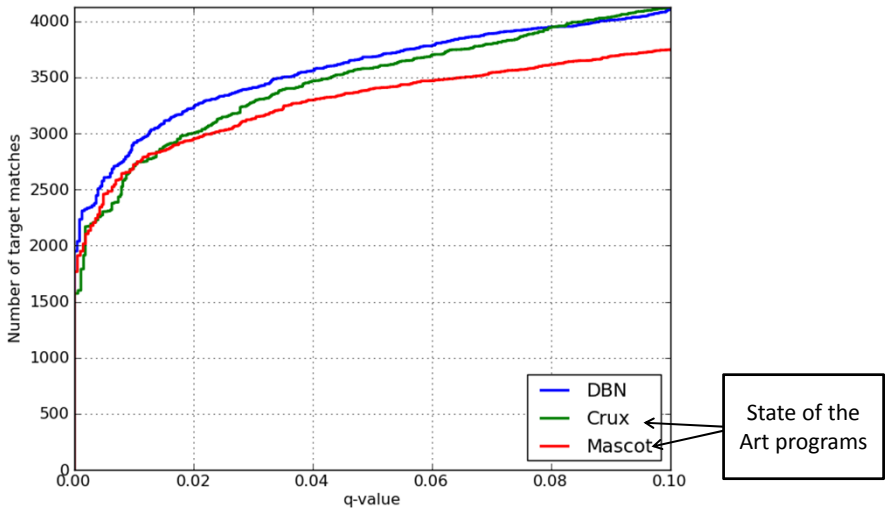
Likelihood ratios



Results

- Testing performed using decoy database:
  - Permute target database values
  - Good algorithm should score targets high, decoys low
  - Performance metric: false discovery rates (FDR)

Results



q-value: think FDR

Questions?

- Model implemented using GMTK (Graphical Models Toolkit) developed by Dr. Jeff Bilmes

Cancer Genomics Problem: Finding tumor-specific mutations

Problem Statement:

Given:

- a human cancer patient
- sample from tumor tissue
- sample from normal tissue
- DNA sequence of both samples

Identify the mutations present in the tumor tissue that are not present in the normal tissue

Background: DNA, Genotypes, Mutations

TAACGCGGTCCCG…

- At the simplest level, we can think of DNA as a sequence of letters from the alphabet, {A,C,G,T}

TAAAGCGGTCCCG…
TAAAGCGGTTCCG…

Genotype AA Genotype CT

- Humans have 2 copies of each chromosome.
- 10 possible "genotypes" at each position

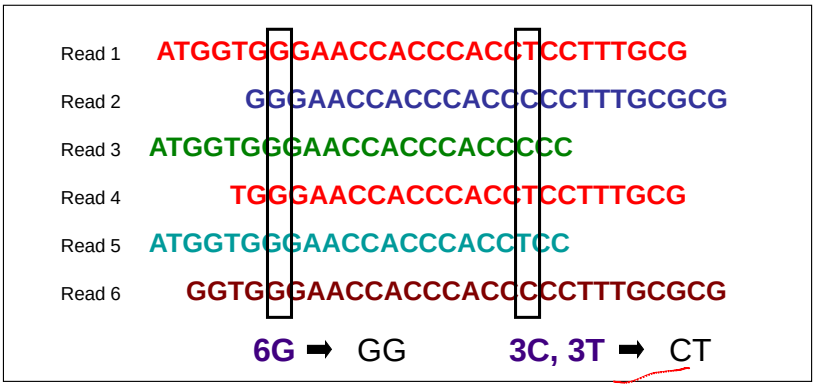
TAAAGCGGTCCCG…
TATAGCGGTTCCG…

Genotype AT Genotype CT

- We are trying to find single base mutations in the tumor DNA

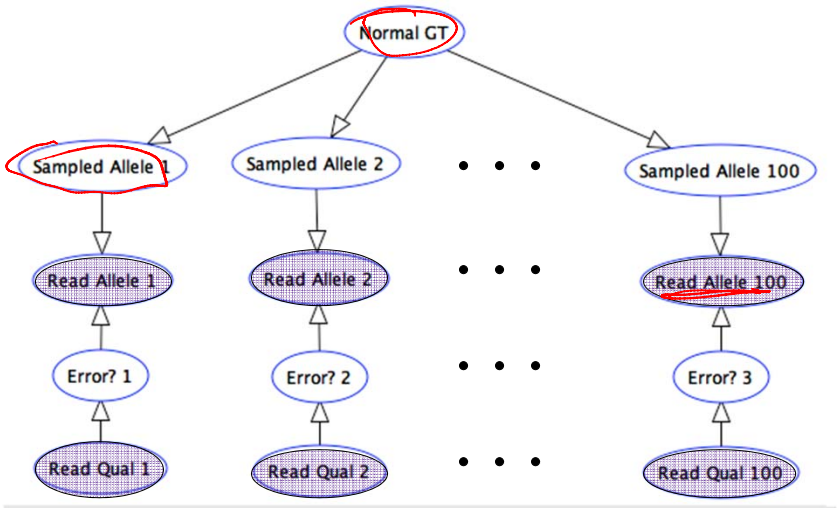
Problem: Call Genotypes from Data

- Our data: lots of overlapping short reads.
- At each position, we count the number of reads for each base…

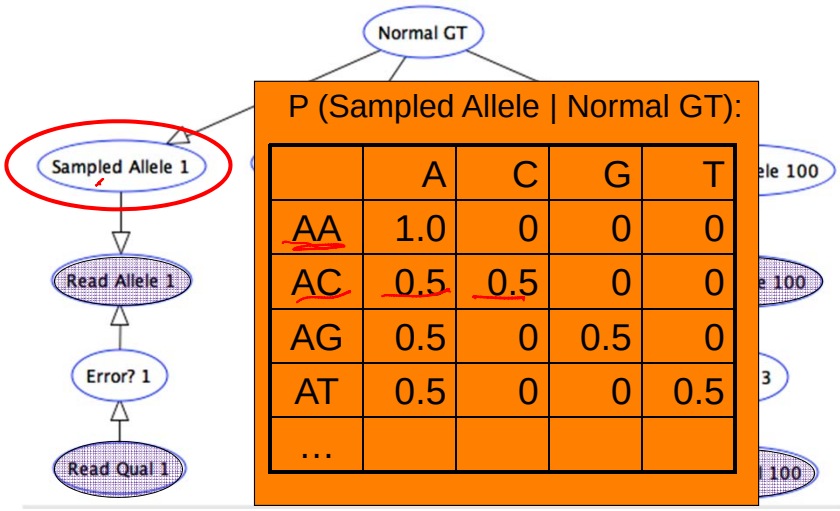


- But the data is noisy: **1C, 9G** ➡ CG? or GG?

Basic PGM - Just Calls Normal GT



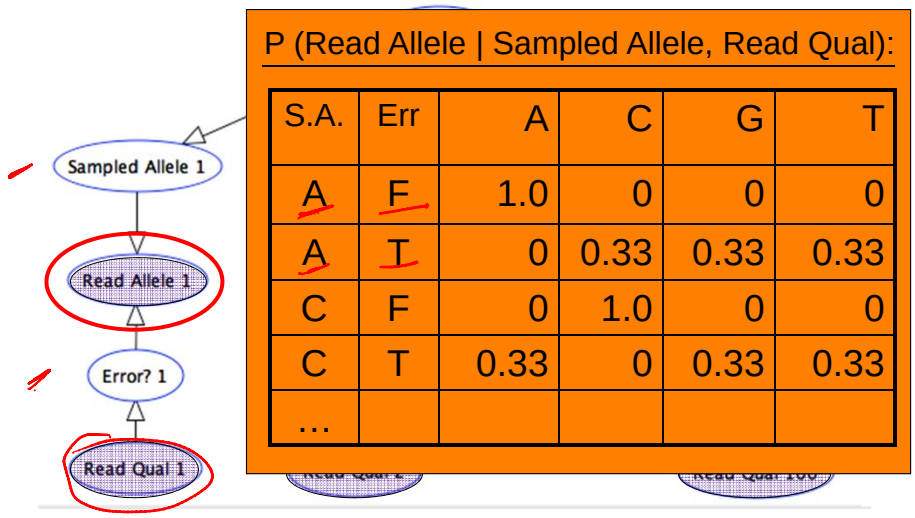
Local Probability Models



P (Sampled Allele | Normal GT):

	A	C	G	T
AA	1.0	0	0	0
AC	0.5	0.5	0	0
AG	0.5	0	0.5	0
AT	0.5	0	0	0.5
…				

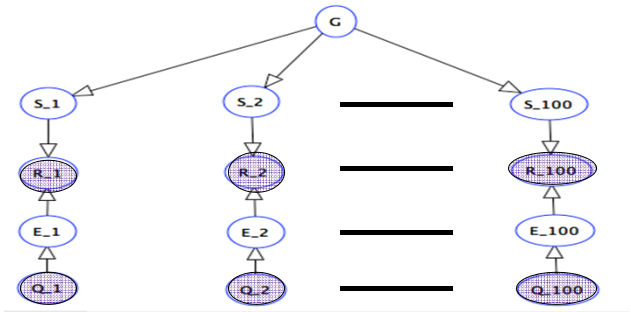
Local Probability Models



P (Read Allele | Sampled Allele, Read Qual):

S.A.	Err	A	C	G	T
A	F	1.0	0	0	0
A	T	0	0.33	0.33	0.33
C	F	0	1.0	0	0
C	T	0.33	0	0.33	0.33
…					

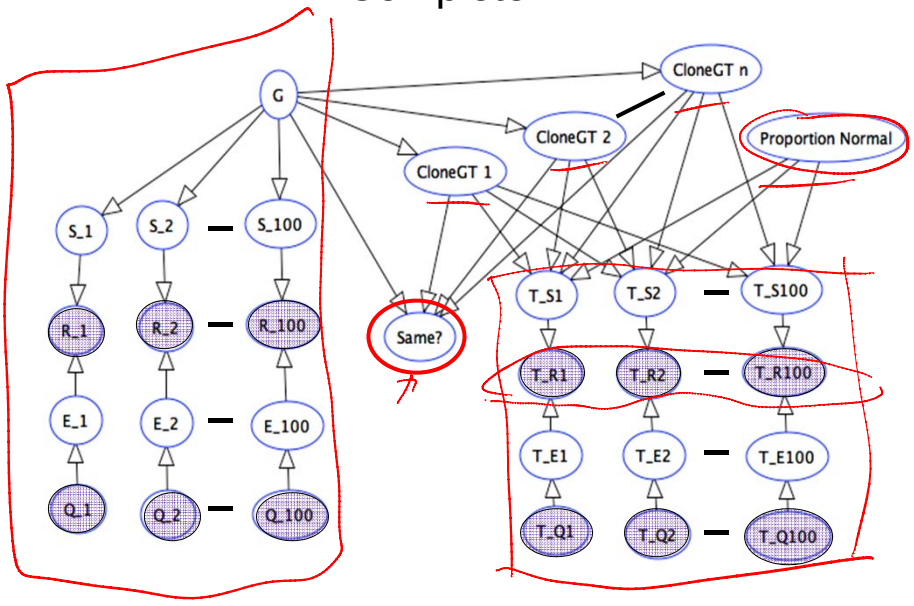
Inference: Find most likely normal genotype



$$P(G, r_1, \ldots, r_{100}, q_1, \ldots, q_{100}) = P(G) \prod_{i=1}^{100} \sum_{S_i, E_i} P(q_i) P(E_i|q_i) P(S_i|G) P(r_i|S_i, E_i)$$

Return $\arg\max_g P(G, r_1, \ldots, r_{100}, q_1, \ldots, q_{100})$

Complete BN



Example Results

A list of possible somatic mutations, ranked by probability:

Chrom	Position	Normal Genotype	Mutated Genotype	Probability
9	53,246,683	AA	AT	.9998
3	4,327,802	CT	TT	.9992
1	10,234,906	GG	AG	.9934
9	52,132,888	GG	CG	.9897
…				