343H: Honors AI

Lecture 20:

Probabilistic reasoning over time II

4/3/2014

Kristen Grauman

UT Austin

Slides courtesy of Dan Klein, UC Berkeley

Unless otherwise noted

Contest results

Announcements

- Reminder: Contest qualification runs nightly, final deadline 4/28

- PS 4
  - Extending deadline to Monday 4/14
  - But no shift in PS 5 deadline (4/24)

Recap of calendar

- 3/25 Contest posted
- 4/10 PS 5 posted
- 4/14: PS 4 due (extended from 4/10)
- 4/24 PS 5 due
- 4/28 Contest qualification closes
- 4/29 Final tournament (evening)
- 5/12 (Mon) Final exam, 2-5 pm CPE 2.218

Some context

- First weeks: Search (BFS, A*, minimax, alpha-beta)
  - Find an optimal plan (or solution)
  - Best thing to do from the current state
  - Know transition and cost (reward) functions
  - Either execute complete solution (deterministic) or search again at every step
  - **Know current state**
- Next: MDPs – towards reinforcement learning
  - Still know transition and reward function
  - Looking for a policy: optimal action from every state
- Before midterm: reinforcement learning
  - Policy without knowing transition or reward functions
  - **Still know state**

Some context (cont.)

- Probabilistic reasoning: now state is unknown
  - Bayesian networks: state estimation/inference
  - Prior, net structure, and CPT's known
    - Probabilities and utilities (from before)
    - Conditional independence and inference (exact and approximate)
    - Exact state estimation over time
    - Approximate state estimation over time

    - (…What if they're not known? Machine learning)

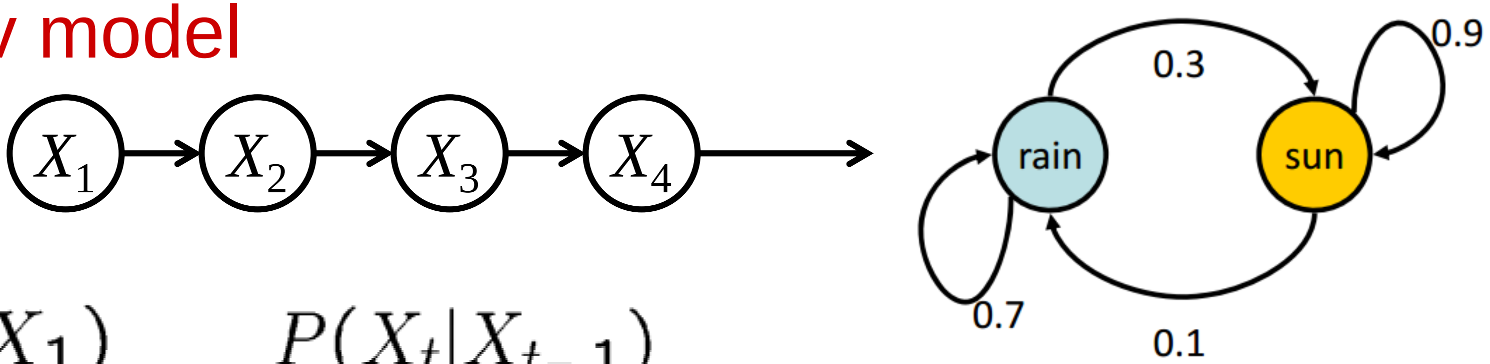
Outline

- Last time:
  - Markov chains
  - HMMs

- Today:
  - Particle filtering
  - Dynamic Bayes’ Nets
  - Most likely explanation queries in HMMs

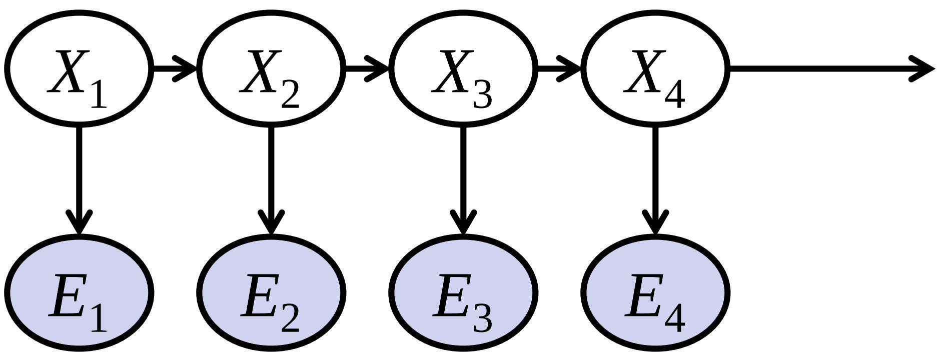
Recap: Reasoning over time

- Markov model



$$P(X_1) \qquad P(X_t|X_{t-1})$$

- Hidden Markov model



$$P(E|X)$$

X	E	P
rain	umbrella	0.9
rain	no umbrella	0.1
sun	umbrella	0.2
sun	no umbrella	0.8

The Forward Algorithm

- We are given evidence at each time and want to know

$$B_t(X) = P(X_t|e_{1:t})$$

- We can derive the following updates

$$\begin{aligned}
P(x_t|e_{1:t}) \propto_X & \; P(x_t, e_{1:t}) \\
&= \sum_{x_{t-1}} P(x_{t-1}, x_t, e_{1:t}) \\
&= \sum_{x_{t-1}} P(x_{t-1}, e_{1:t-1}) P(x_t|x_{t-1}) P(e_t|x_t) \\
&= P(e_t|x_t) \sum_{x_{t-1}} P(x_t|x_{t-1}) P(x_{t-1}, e_{1:t-1})
\end{aligned}$$

This is exactly variable elimination with order X1, X2, …

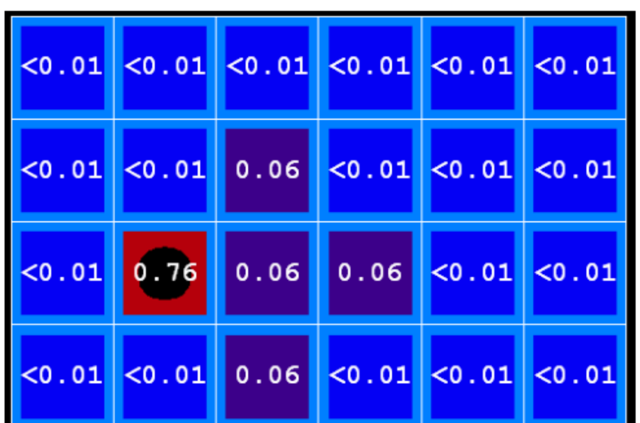
Recap: Filtering with Forward Algorithm

Elapse time: compute P(X_t | $e_{1:t-1}$)

$$P(x_t|e_{1:t-1}) = \sum_{x_{t-1}} P(x_{t-1}|e_{1:t-1}) \cdot P(x_t|x_{t-1})$$

Observe: compute P(X_t | $e_{1:t}$)

$$P(x_t|e_{1:t}) \propto P(x_t|e_{1:t-1}) \cdot P(e_t|x_t)$$



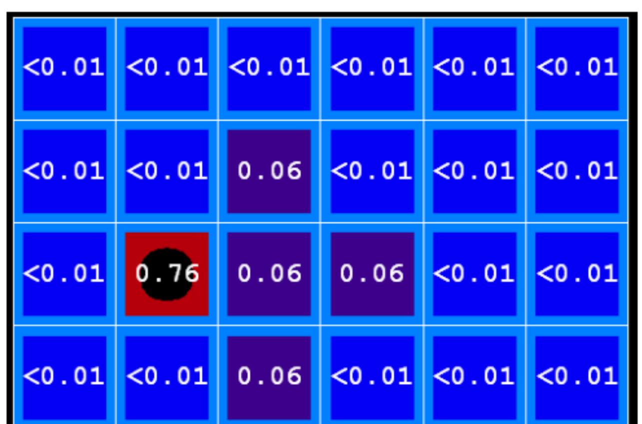
Recap: Filtering with Forward Algorithm

Elapse time: compute P(X_t | $e_{1:t-1}$)

$$P(x_t|e_{1:t-1}) = \sum_{x_{t-1}} P(x_{t-1}|e_{1:t-1}) \cdot P(x_t|x_{t-1})$$

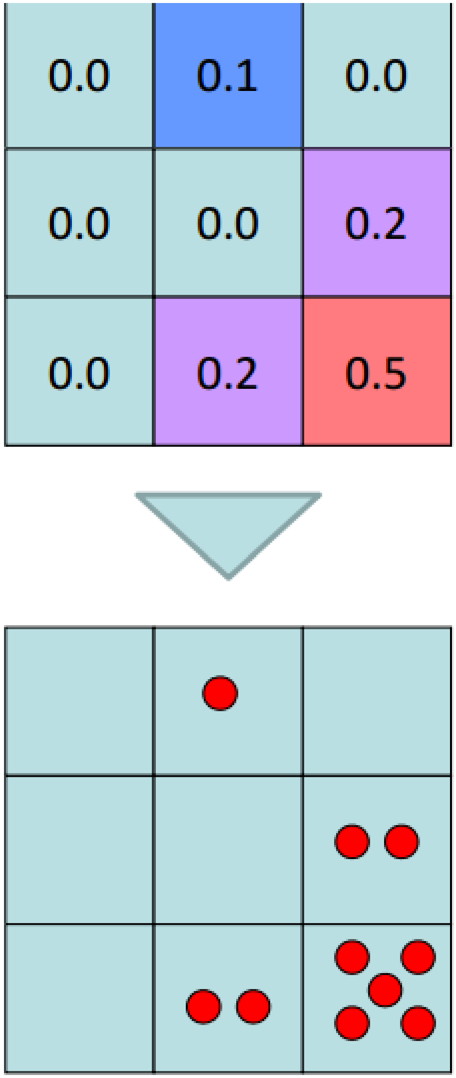
Observe: compute P(X_t | $e_{1:t}$)

$$P(x_t|e_{1:t}) \propto P(x_t|e_{1:t-1}) \cdot P(e_t|x_t)$$



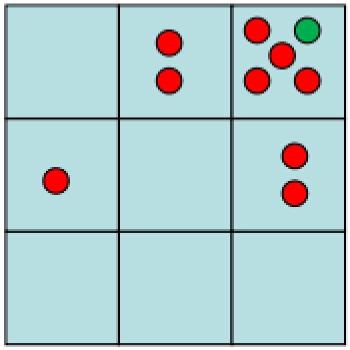
Particle filtering

- Filtering: approximate solution
- Sometimes |X| is too big to use exact inference
  - |X| may be too big to even store B(X)
  - E.g., X is continuous
- Solution: approximate inference
  - Track samples of X, not all values
  - Samples are called *particles*
  - Time per step is linear in the number of samples, but may be large
  - In memory: list of particles, not states



Representation: Particles

- Our representation of P(X) is now a list of N particles (samples)
  - Generally, N << |X|
  - Storing map from X to counts would defeat the point
- P(x) approximated by number of particles with value x
  - So, many x may have P(x) = 0!
  - More particles, more accuracy
- For now, all particles have weight 1.



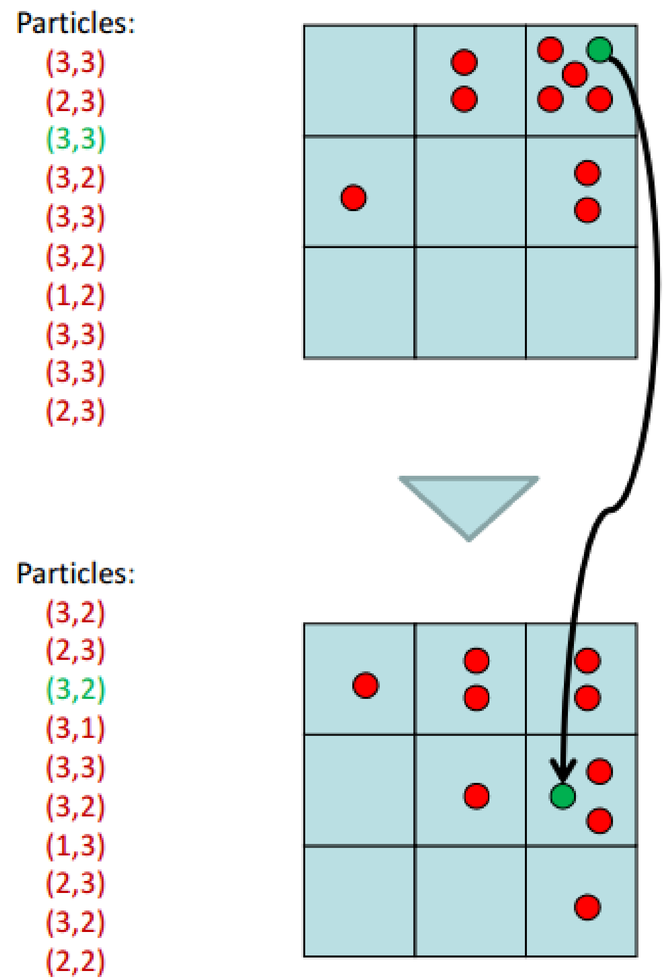
Particles:
(3,3)
(2,3)
(3,3)
(3,2)
(3,3)
(3,2)
(1,2)
(3,3)
(3,3)
(2,3)

Particle filtering: Elapse time

- Each particle is moved by sampling its next position from the transition model

$$x' = \text{sample}(P(X'|x))$$

  - This is like prior sampling –samples' frequencies reflect the transition probabilities
  - Here, most samples move clockwise, but some move in another direction or stay in place
- This captures the passage of time
  - If enough samples, close to exact values before and after (consistent)



Particle filtering: Observe

- Slightly trickier:
  - Don’t sample observation, fix it
  - Similar to likelihood weighting, downweight samples based on the evidence

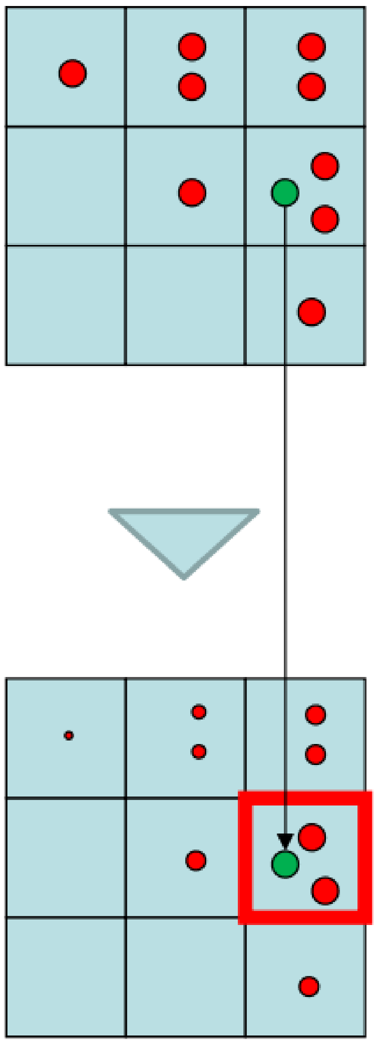
$$w(x) = P(e|x)$$

$$B(X) \propto P(e|X)B'(X)$$

  - As before, the probabilities don’t sum to one, since all have been downweighted.

Particles:
(3,2)
(2,3)
(3,2)
(3,1)
(3,3)
(3,2)
(1,3)
(2,3)
(3,2)
(2,2)

Particles:
(3,2) w=.9
(2,3) w=.2
(3,2) w=.9
(3,1) w=.4
(3,3) w=.4
(3,2) w=.9
(1,3) w=.1
(2,3) w=.2
(3,2) w=.9
(2,2) w=.4



Particle filtering: Observe

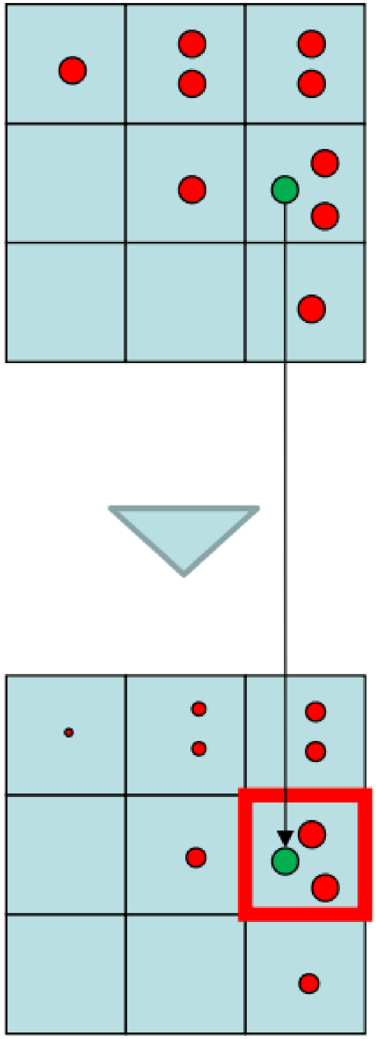
- Slightly trickier:
  - Don't sample observation, fix it
  - Similar to likelihood weighting, downweight samples based on the evidence

$$w(x) = P(e|x)$$

$$B(X) \propto P(e|X)B'(X)$$

  - As before, the probabilities don't sum to one, since all have been downweighted.

Particles:
(3,2)
(2,3)
(3,2)
(3,1)
(3,3)
(3,2)
(1,3)
(2,3)
(3,2)
(2,2)

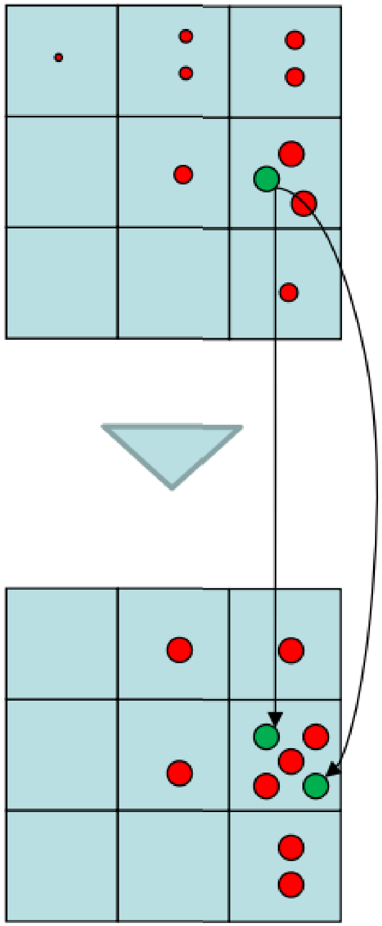


Particles:
(3,2) w=.9
(2,3) w=.2
(3,2) w=.9
(3,1) w=.4
(3,3) w=.4
(3,2) w=.9
(1,3) w=.1
(2,3) w=.2
(3,2) w=.9
(2,2) w=.4

Particle filtering: Resample

- Rather than tracking weighted samples, we resample
- N times, we choose from our weighted sample distribution (i.e., draw with replacement)
- This is like renormalizing the distribution
- Now the update is complete for this time step, continue with the next one

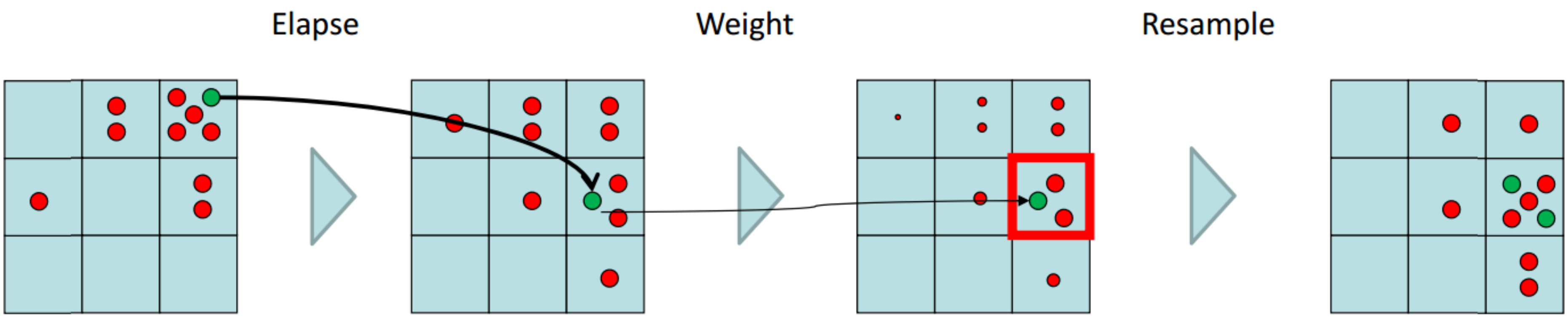
Particles:
(3,2) w=.9
(2,3) w=.2
(3,2) w=.9
(3,1) w=.4
(3,3) w=.4
(3,2) w=.9
(1,3) w=.1
(2,3) w=.2
(3,2) w=.9
(2,2) w=.4



(New) Particles:
(3,2)
(2,2)
(3,2)
(2,3)
(3,3)
(3,2)
(1,3)
(2,3)
(3,2)
(3,2)

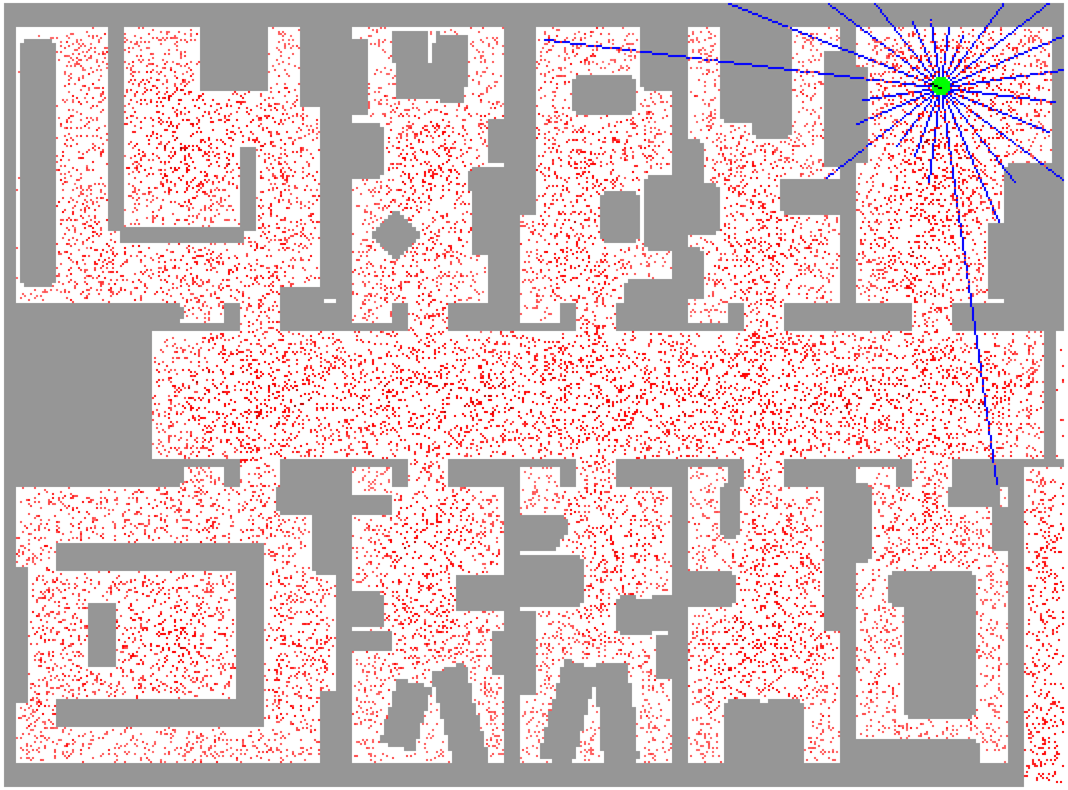
Recap: Particle filtering

- Particles: track samples of states rather than an explicit distribution



Particles:	Particles:	Particles:	(New) Particles:
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(2,3)	(2,3)	(2,3) w=.2	(2,2)
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(3,2)	(3,1)	(3,1) w=.4	(2,3)
(3,3)	(3,3)	(3,3) w=.4	(3,3)
(3,2)	(3,2)	(3,2) w=.9	(3,2)
(1,2)	(1,3)	(1,3) w=.1	(1,3)
(3,3)	(2,3)	(2,3) w=.2	(2,3)
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(2,3)	(2,2)	(2,2) w=.4	(3,2)

Example: robot localization



- http://robots.stanford.edu/videos.html

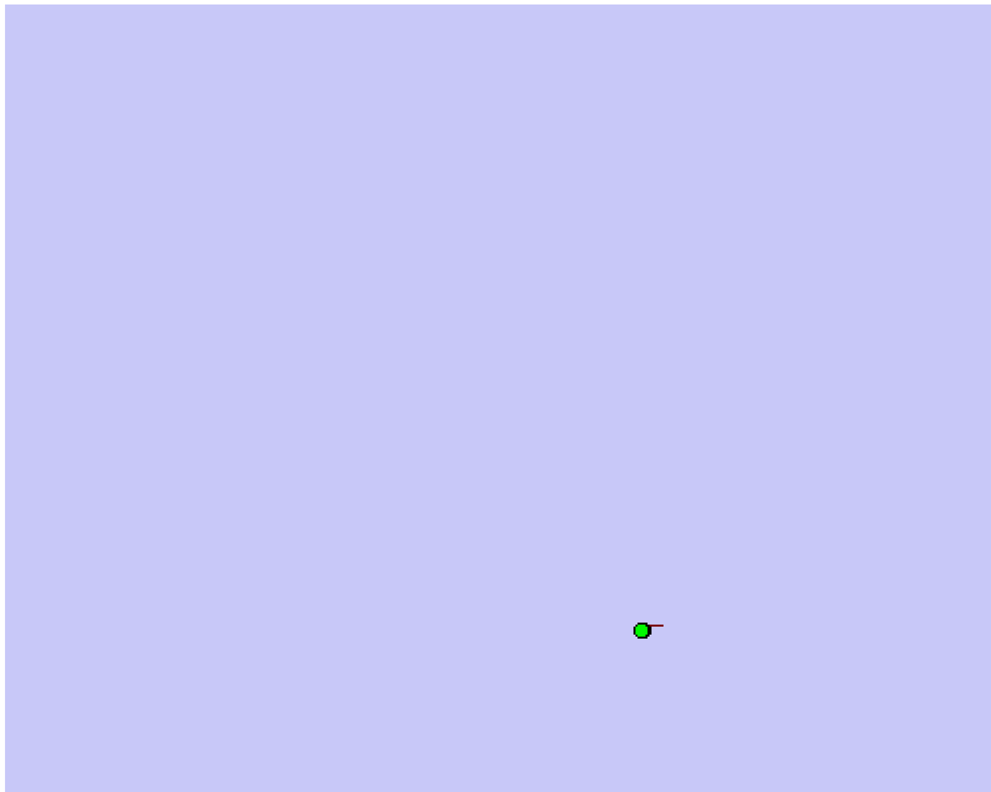
Example: robot localization

http://www.cs.washington.edu/robotics/mcl/

Robot mapping

- SLAM: Simultaneous localization and mapping
  - We do not know map or our location
  - State consists of position AND map!
  - Main techniques: Kalman filtering (Gaussian HMMs) and particle methods

SLAM



http://www.cs.washington.edu/robotics/mcl/

RGB-D Mapping: Result

[Henry, Krainin, Herbst, Ren, Fox; ISER 2010, IJRR 2012]

Object tracking

- http://www.robots.ox.ac.uk/~misard/condensation.html

HMMs summary so far

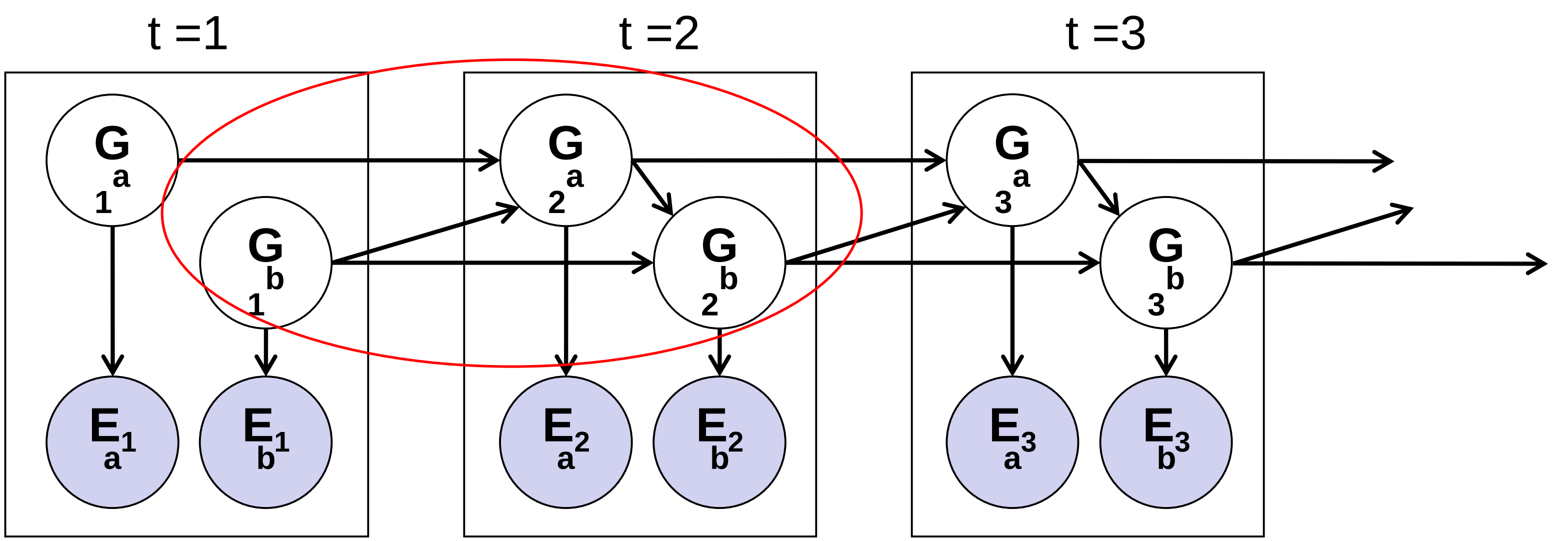
- Markov Models
  - A family of Bayes' nets of a particular regular structure
- Hidden Markov Models (HMMs)
  - Another family of Bayes' nets with regular structure
  - Inference
    - Forward algorithm (repeated variable elimination)
    - Particle filtering (likelihood weighting with some tweaks)

Now

- Dynamic Bayes Nets (brief)
- HMMs: Most likely explanation queries

Dynamic Bayes Nets (DBNs)

- We want to track multiple variables over time, using multiple sources of evidence
- Idea: Repeat a fixed Bayes net structure at each time
- Variables from time *t* can condition on those from *t-1*



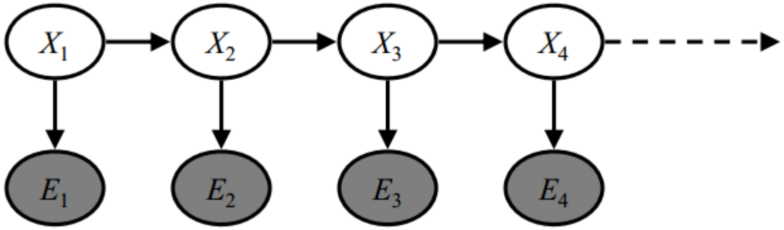
- Discrete valued dynamic Bayes nets are also HMMs

DBN Particle Filters

- A particle is a complete sample for a time step
- **Initialize**: Generate prior samples for the t=1 Bayes net
  - Example particle: $\mathbf{G_1^a}$ = (3,3) $\mathbf{G_1^b}$ = (5,3)
- **Elapse time**: Sample a successor for each particle
  - Example successor: $\mathbf{G_2^a}$ = (2,3) $\mathbf{G_2^b}$ = (6,3)
- **Observe**: Weight each entire sample by the likelihood of the evidence conditioned on the sample
  - Likelihood: $P(\mathbf{E_1^a} \mid \mathbf{G_1^a}) * P(\mathbf{E_1^b} \mid \mathbf{G_1^b})$
- **Resample:** Select prior samples (tuples of values) in proportion to their likelihood

HMMs: MLE queries

- HMMs defined by
  - States X
  - Observations E
  - Initial distribution: $P(X_1)$
  - Transitions: $P(X|X_{-1})$
  - Emissions: $P(E|X)$

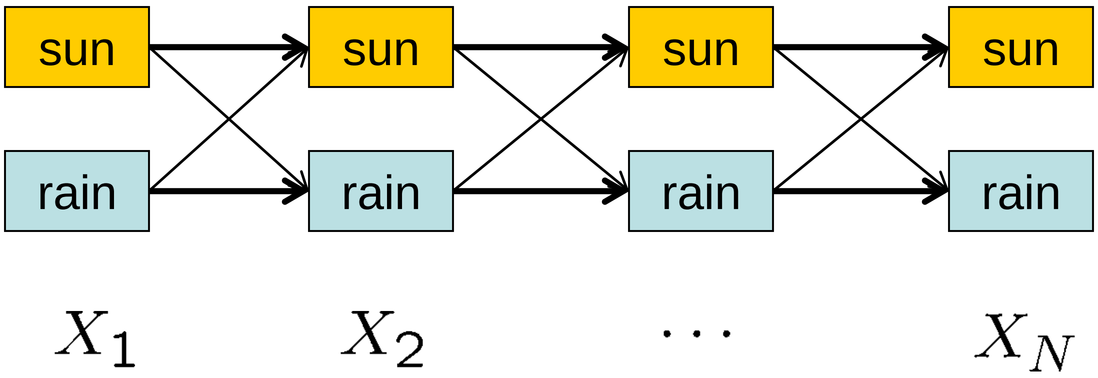


New query: most likely explanation: $\arg\max_{x_{1:t}} P(x_{1:t}|e_{1:t})$

New method: Viterbi algorithm

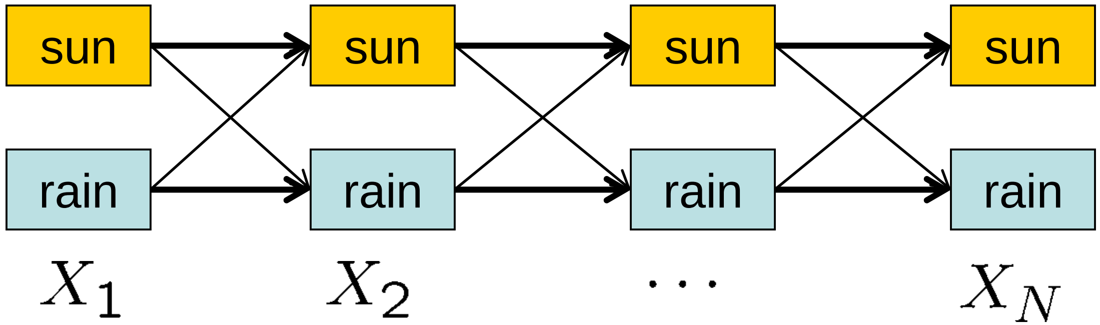
State Trellis

- State trellis: graph of states and transitions over time



- Each arc represents some transition $x_{t-1} \rightarrow x_t$
- Each arc has weight $P(x_t|x_{t-1})P(e_t|x_t)$
- Each path is a sequence of states
- The product of weights on a path is the seq’s probability
- Forward algorithm computes sums of paths, Viterbi computes best paths.

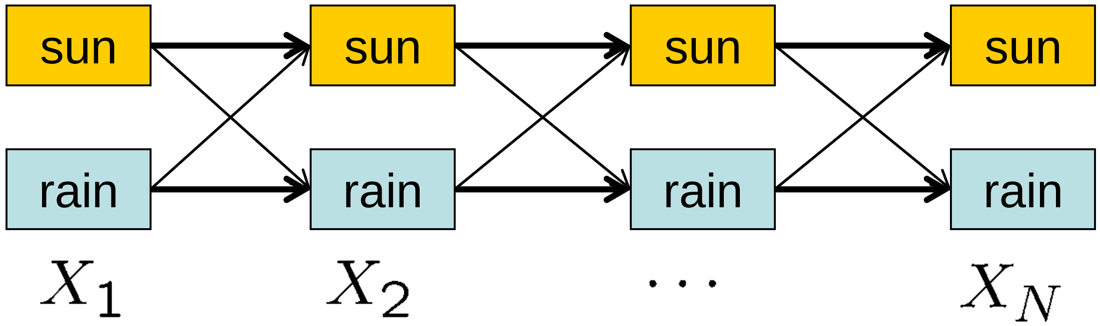
Forward Algorithm (Sum)



$$f_t[x_t] = P(x_t, e_{1:t})$$

$$= P(e_t|x_t) \sum_{x_{t-1}} P(x_t|x_{t-1}) f_{t-1}[x_{t-1}]$$

Viterbi Algorithm (Max)



$$m_t[x_t] = \max_{x_{1:t-1}} P(x_{1:t-1}, x_t, e_{1:t})$$

$$= P(e_t|x_t) \max_{x_{t-1}} P(x_t|x_{t-1}) m_{t-1}[x_{t-1}]$$

Example: Photo Geo-location

Where was this picture taken?

Instance recognition works quite well

Example: Photo Geo-location

Where was this picture taken?

Example: Photo Geo-location

Where was this picture taken?

Example: Photo Geo-location

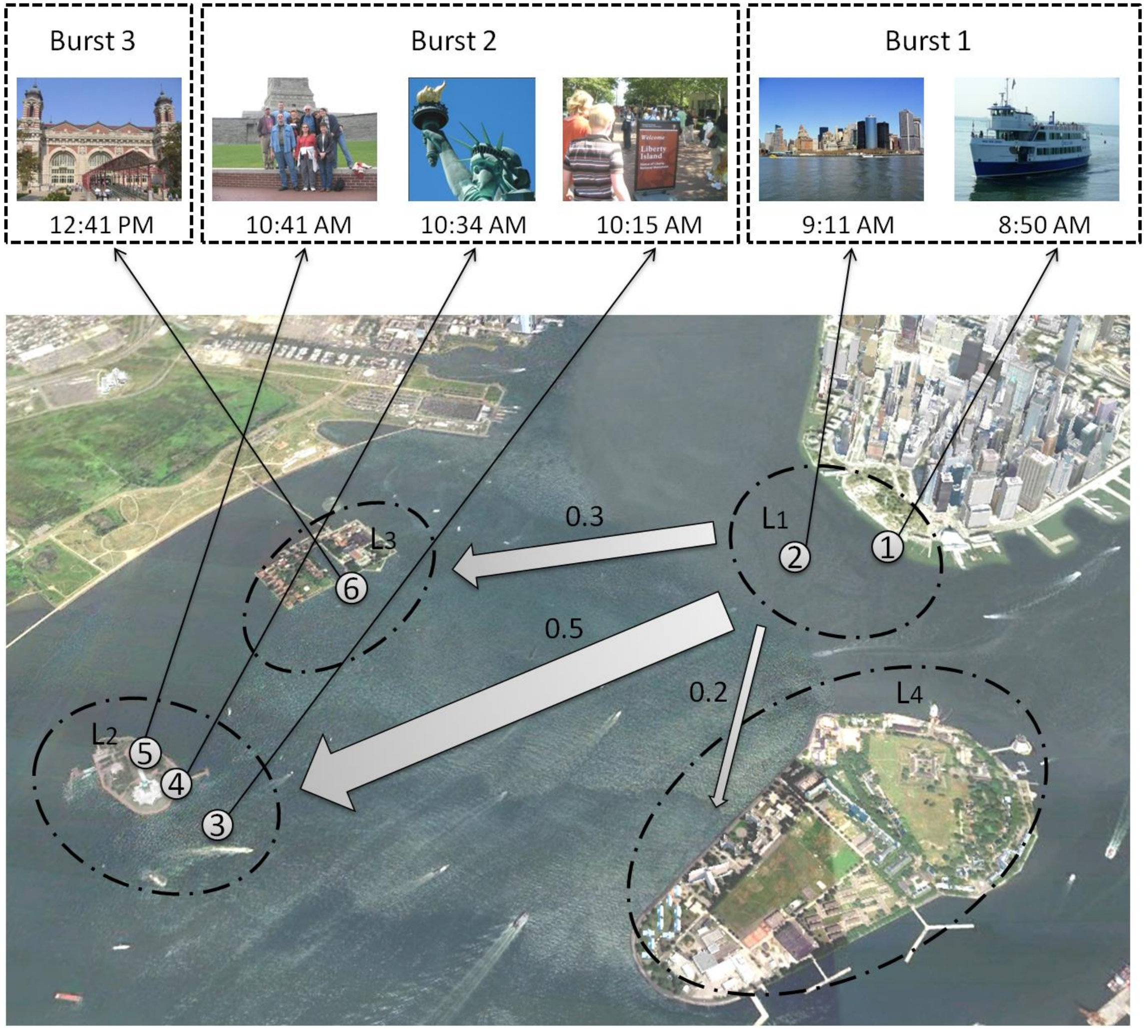
Where was *each picture in this sequence* taken?

Idea: Exploit the beaten path

- Learn dynamics model from “training” tourist photos
- Exploit timestamps and sequences for novel “test” photos

[Chen & Grauman CVPR 2011]

Idea: Exploit the beaten path

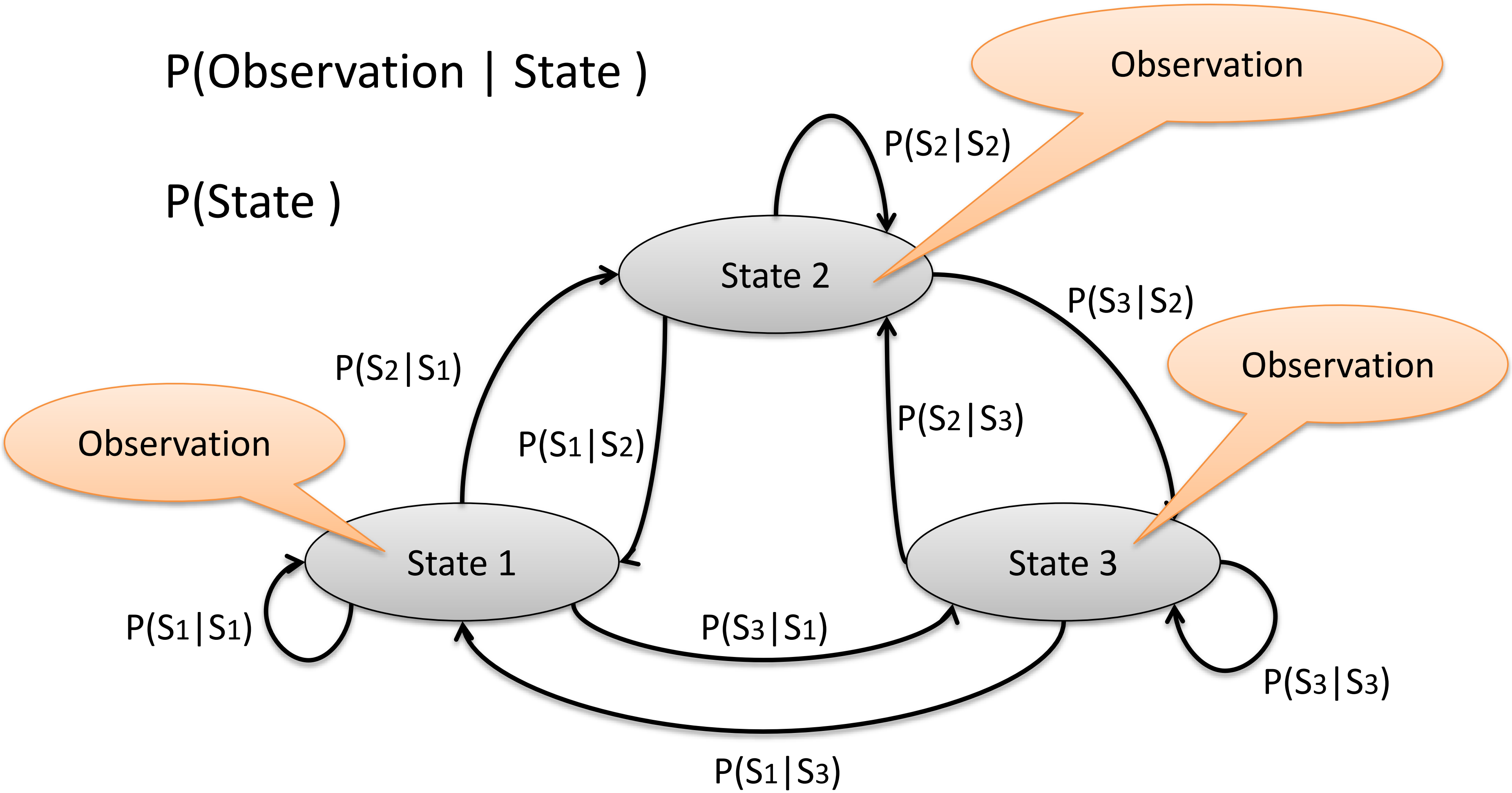


[Chen & Grauman CVPR 2011]

Hidden Markov Model

P(Observation | State)

P(State)

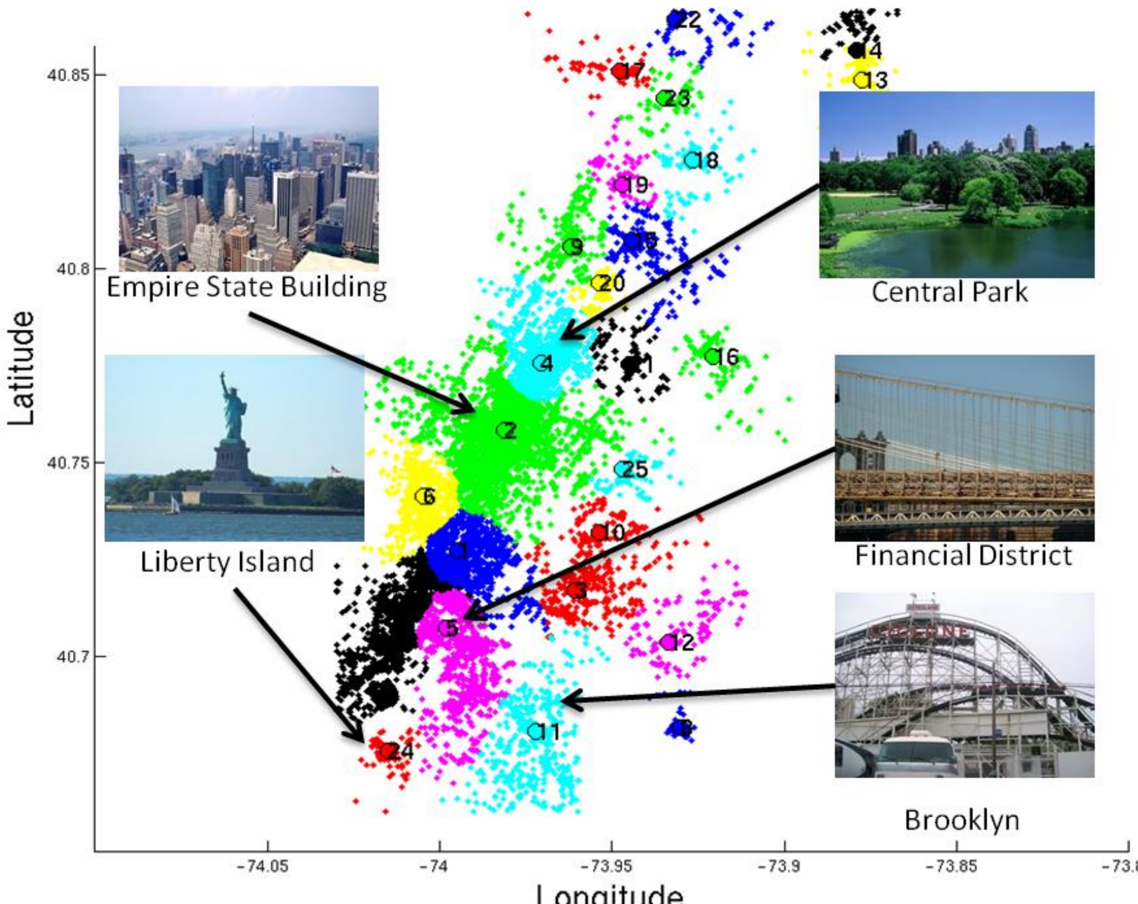


[Chen & Grauman CVPR 2011]

Discovering a city's locations

Define states with data-driven approach:

New York

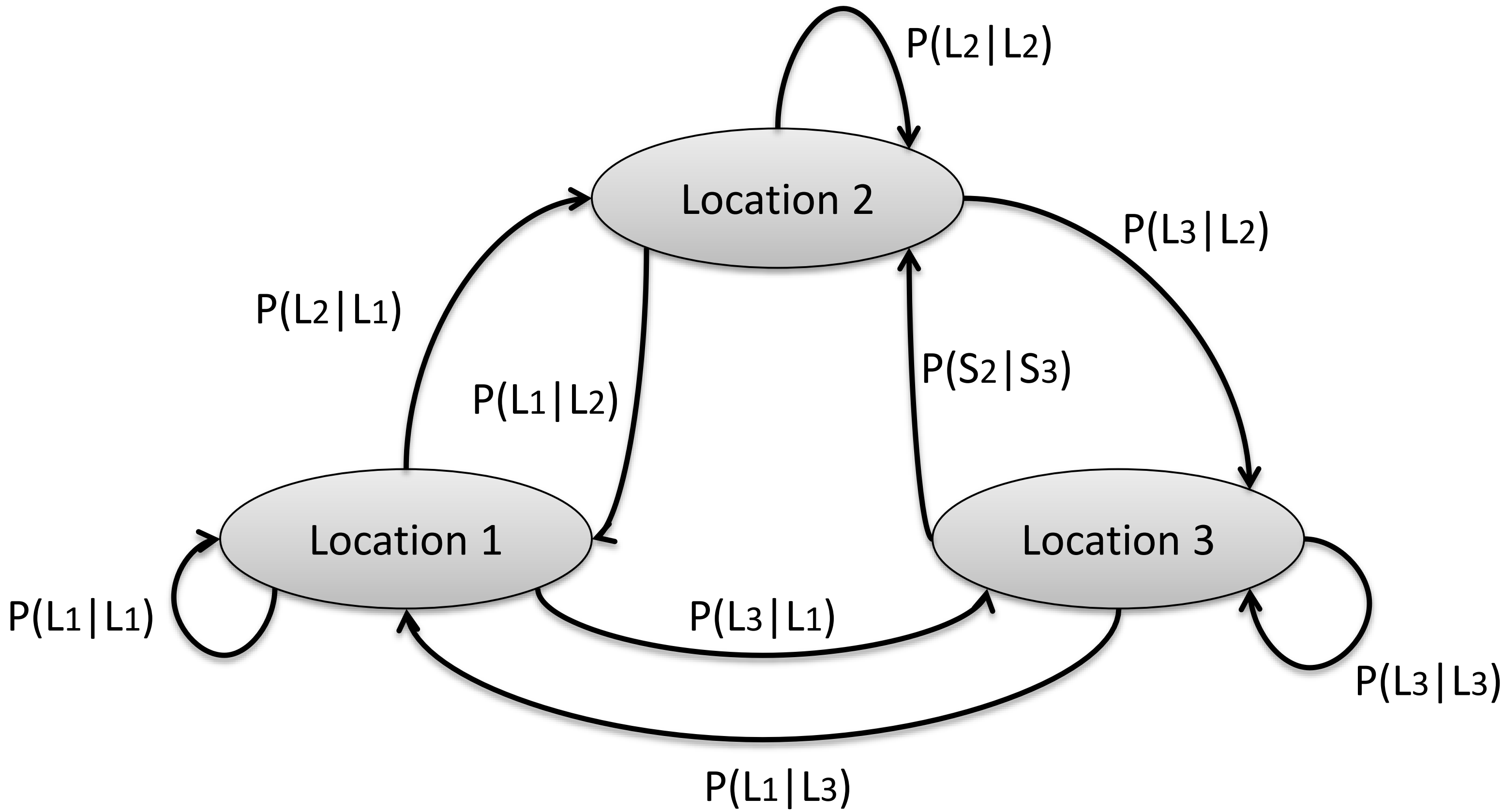


mean shift clustering on the GPS coordinates of the training images

[Chen & Grauman CVPR 2011]

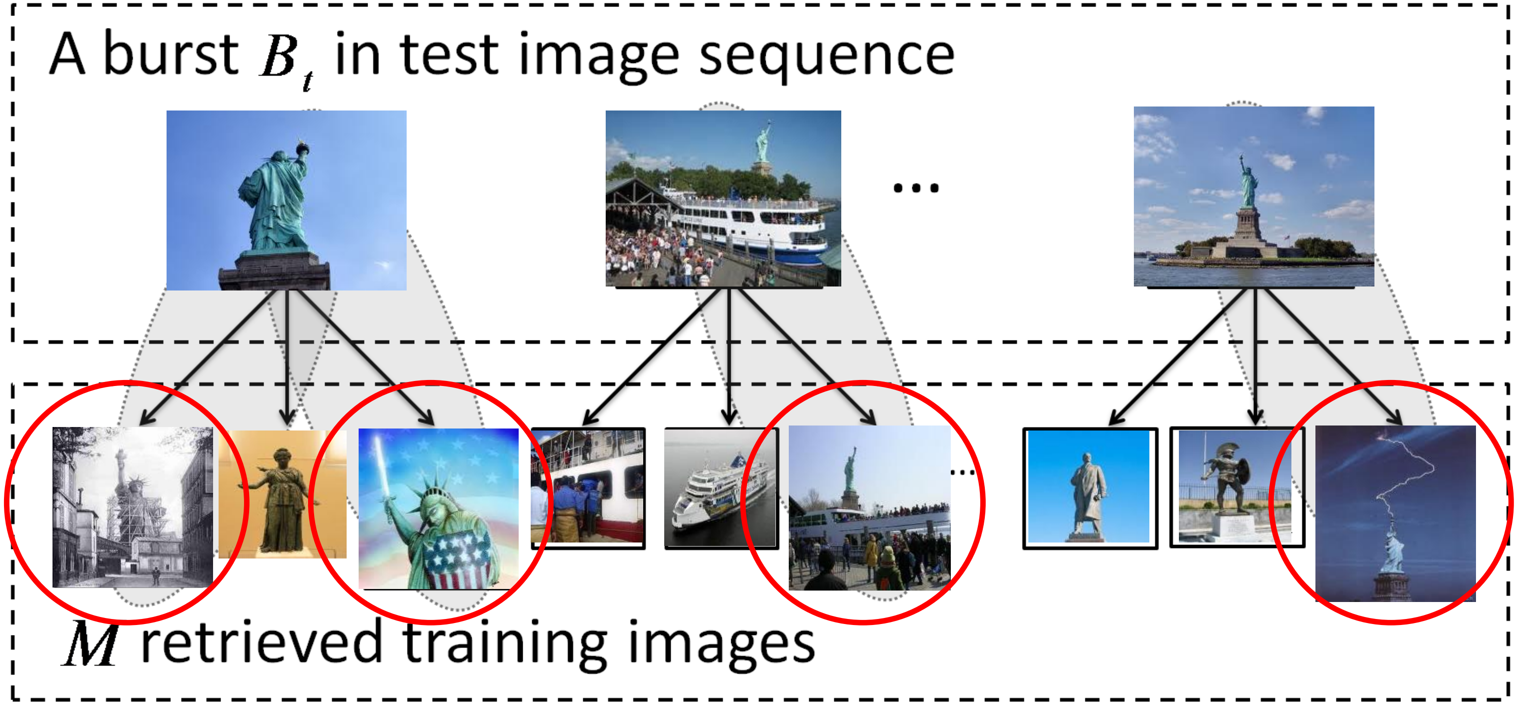
Observation model

P(Observation | State) = P(   | Liberty Island)



[Chen & Grauman CVPR 2011]

Observation model



[Chen & Grauman CVPR 2011]

Location estimation accuracy

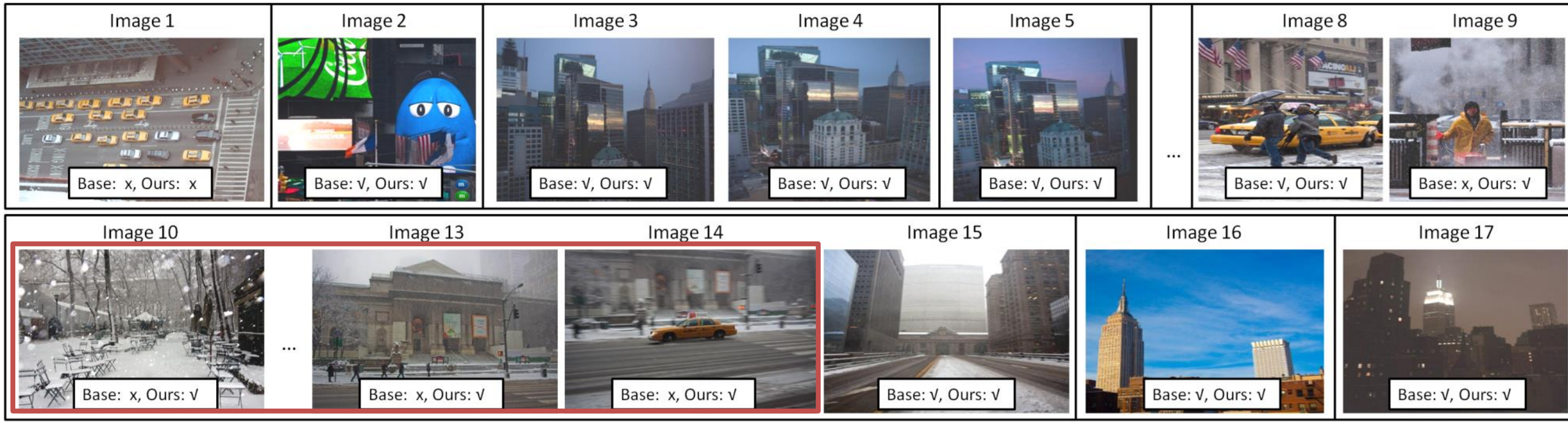
	NN	Img-HMM	Burst Only	Burst-HMM (Ours)
Avg/seq	0.1502	0.1608	0.1764	**0.2036**
Overall	0.1592	0.1660	0.2617	**0.2782**

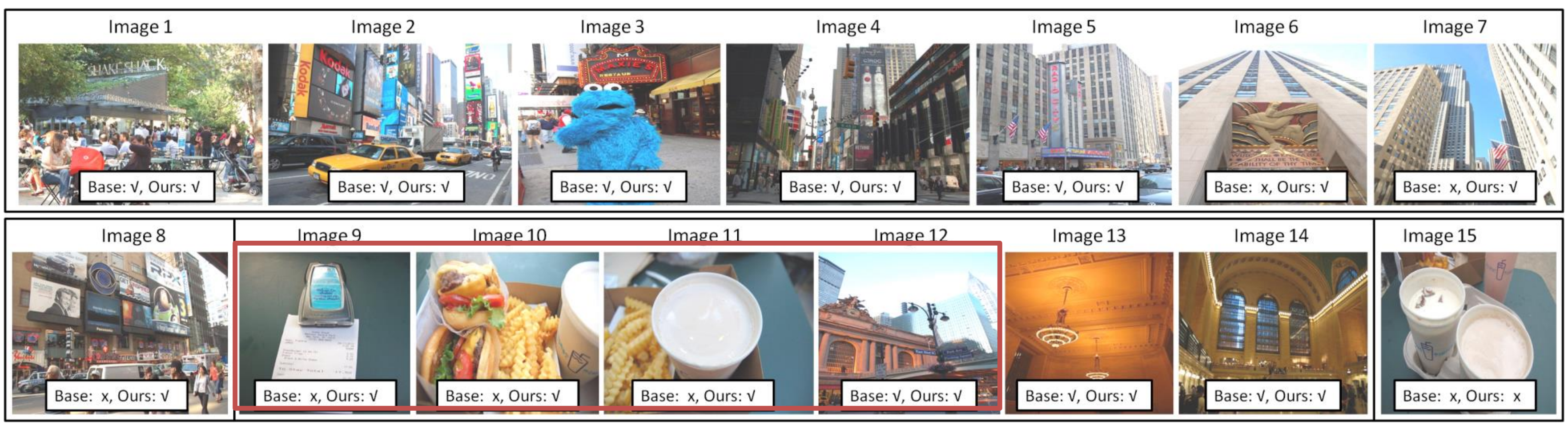
(a) Rome dataset

	NN	Img-HMM	Burst Only	Burst-HMM (Ours)
Avg/seq	0.2323	0.2124	0.2099	**0.3021**
Overall	0.2302	0.2070	0.2055	**0.3143**

(b) New York City dataset

Qualitative Result – New York

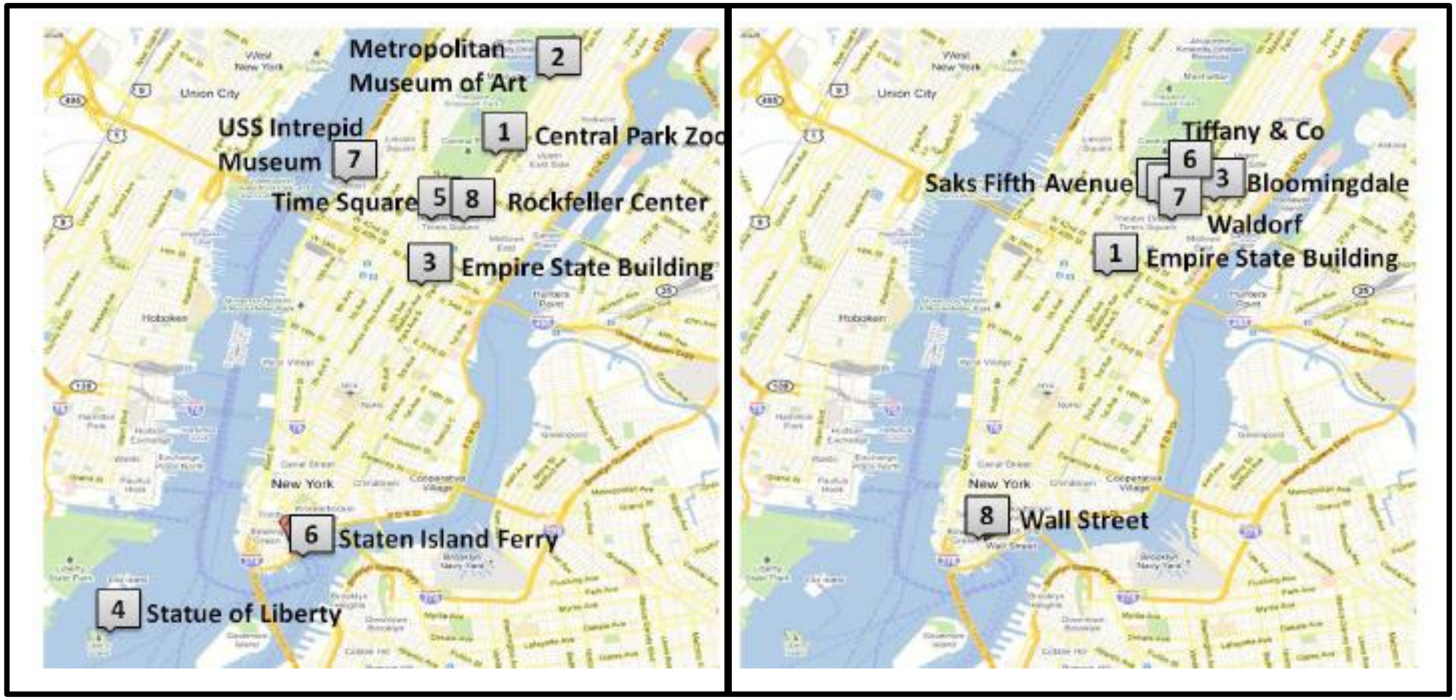




[Chen & Grauman CVPR 2011]

Discovering travel guides' beaten paths

Routes from travel guide book for New York

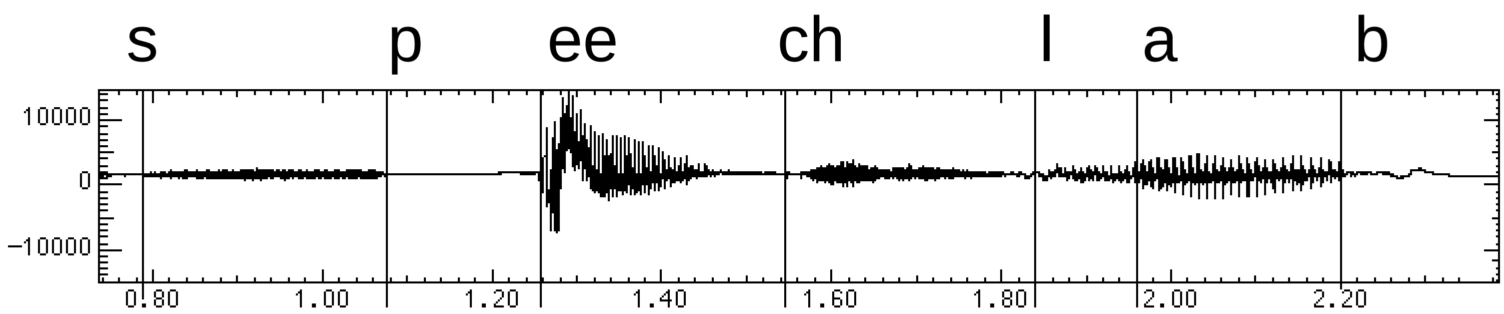


	Rand. Walk	Rand. Walk(TS)	Guidebook
Route Prob.	$6.3 \cdot 10^{-12}$	$4.2 \cdot 10^{-11}$	$2.0 \cdot 10^{-4}$

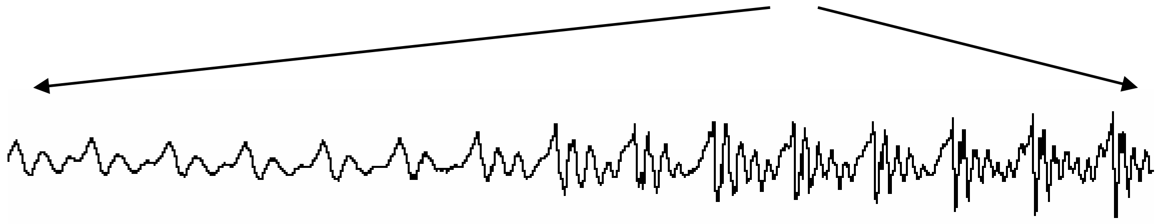
[Chen & Grauman CVPR 2011]

Digitizing speech

- Speech input is an acoustic wave form



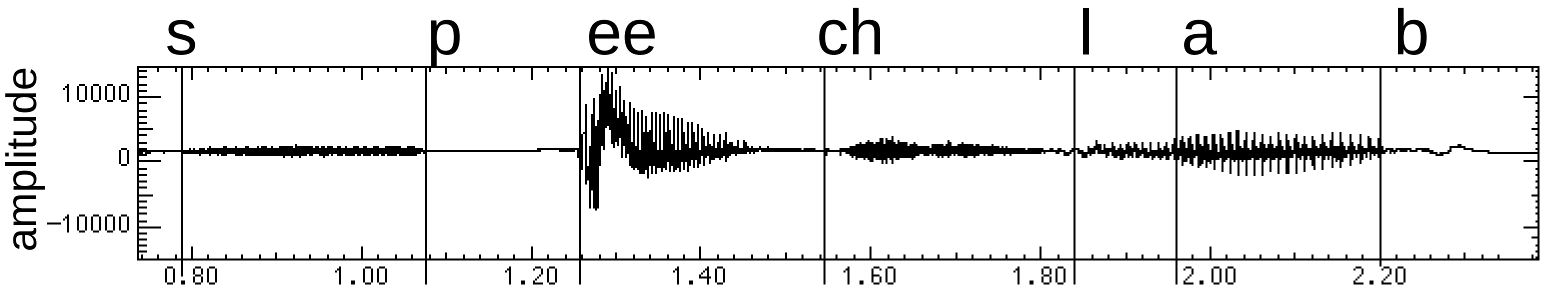
“l” to “a” transition:



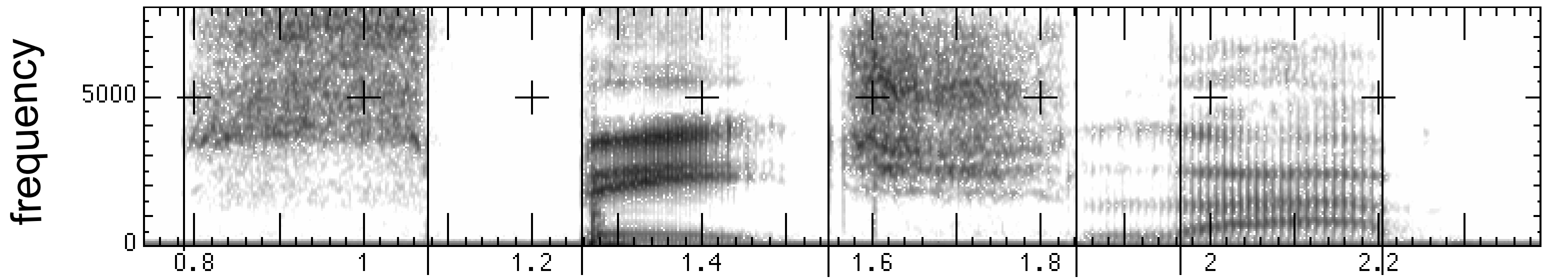
Graphs from Simon Arnfield’s web tutorial on speech, Sheffield:
http://www.psyc.leeds.ac.uk/research/cogn/speech/tutorial/

Spectral Analysis

- Frequency gives pitch; amplitude gives volume
  - sampling at ~8 kHz phone, ~16 kHz mic (kHz=1000 cycles/sec)

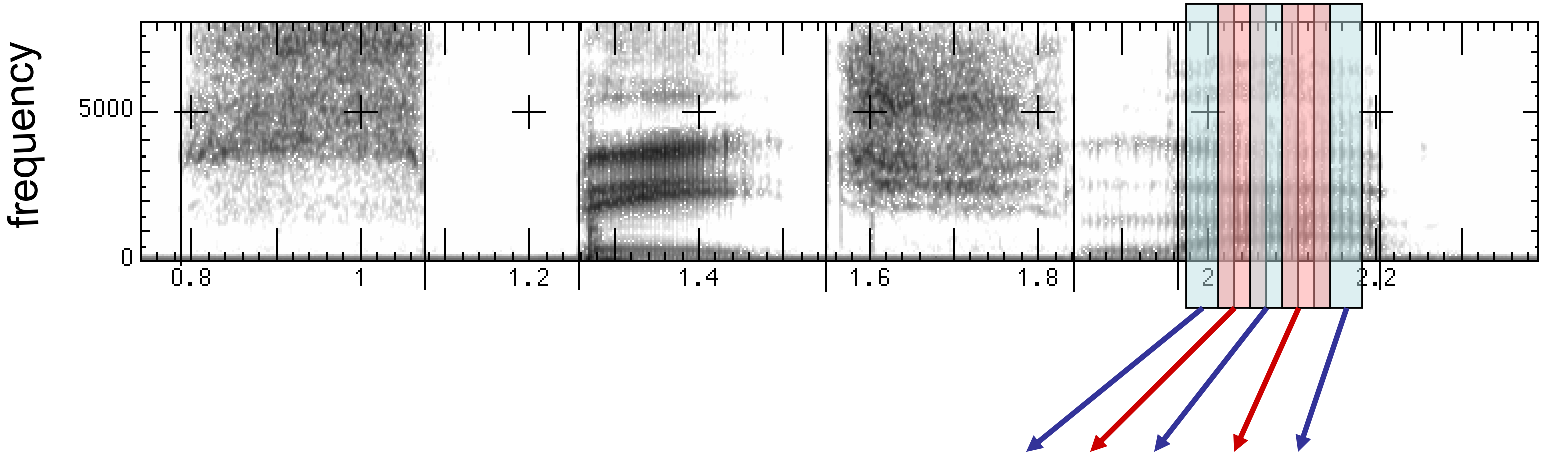


- Fourier transform of wave displayed as a spectrogram
  - darkness indicates energy at each frequency



Acoustic Feature Sequence

- Time slices are translated into acoustic feature vectors (~39 real numbers per slice)



……………………………………………$e_{12}e_{13}e_{14}e_{15}e_{16}$………

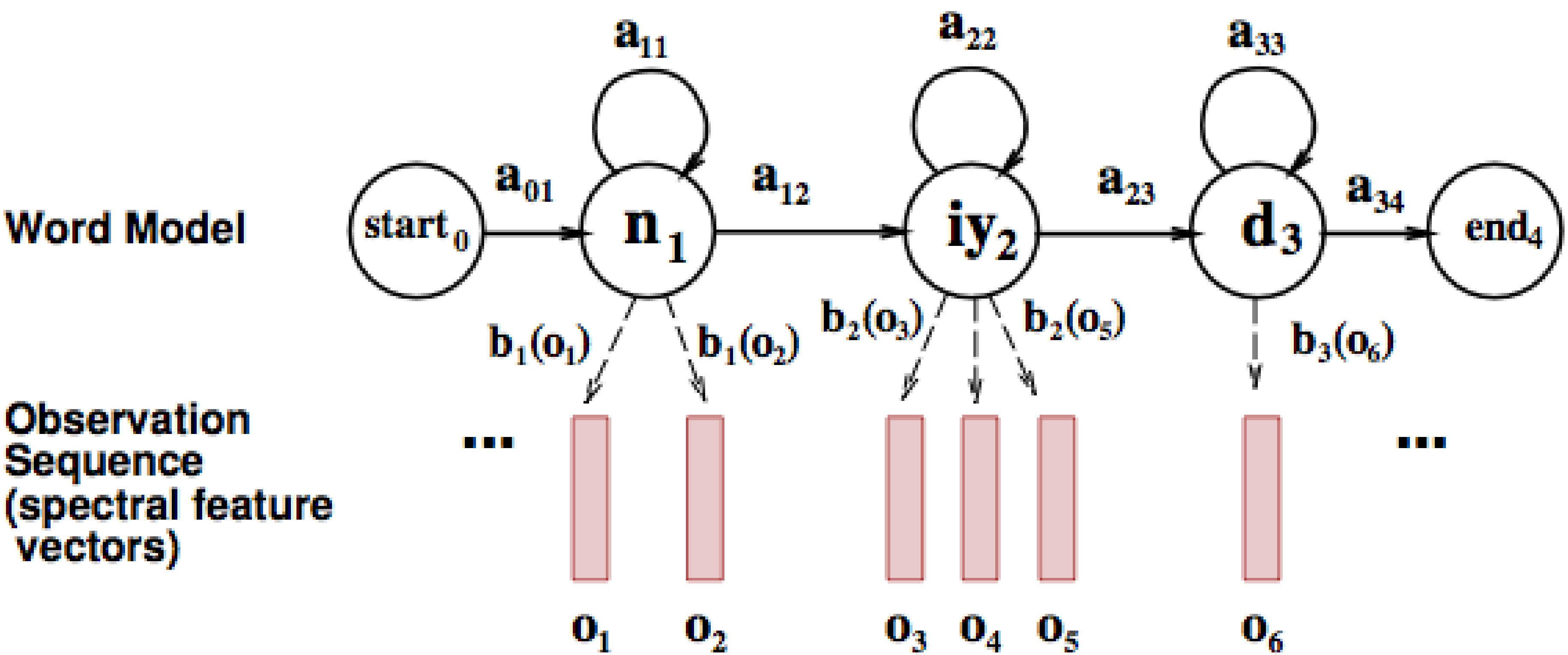
- These are the observations, now we need the hidden states X

Speech State Space

- HMM specification
  - P(E|X) encodes which acoustic vectors are appropriate for each phoneme (each kind of sound)
  - P(X | X’) encodes how sounds can be strung together

- State space
  - We will have one state for each sound in each word
  - Mostly, states advance sound by sound
  - Build a little state graph for each word and chain them together to form the state space X

States in a word



Transitions with Bigrams

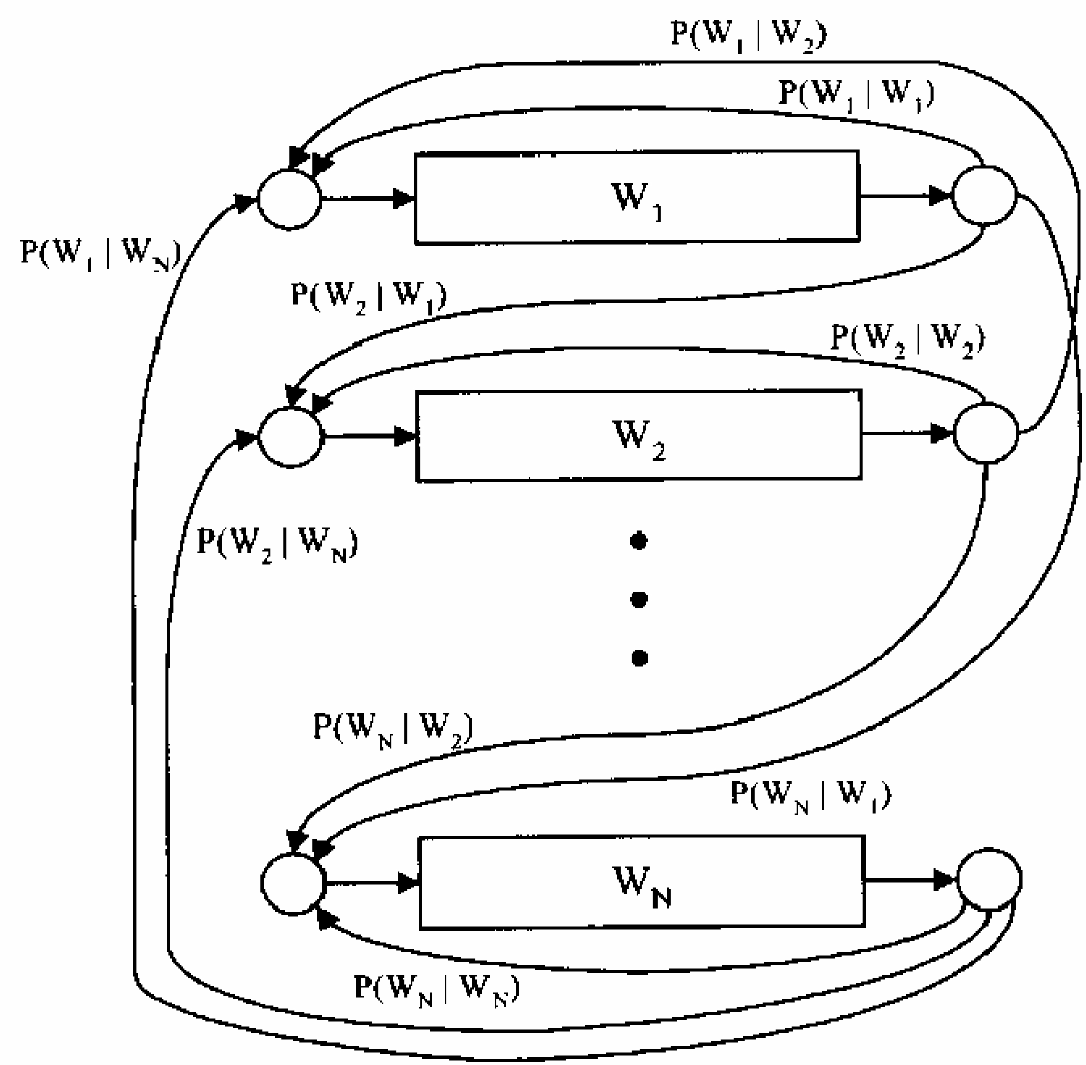


Figure from Huang et al page 618

Training Counts

198015222 the first
194623024 the same
168504105 the following
158562063 the world
…
14112454 the door

23135851162 the *

$$\hat{P}(\mathrm{door}|\mathrm{the}) = \frac{14112454}{23135851162}$$

$$= 0.0006$$

Decoding

- Finding the words given the acoustics is an HMM inference problem

- We want to know which state sequence $x_{1:T}$ is most likely given the evidence $e_{1:T}$:

$$
\begin{aligned}
x^*_{1:T} &= \arg\max_{x_{1:T}} P(x_{1:T} | e_{1:T}) \\
&= \arg\max_{x_{1:T}} P(x_{1:T}, e_{1:T})
\end{aligned}
$$

- From the sequence x, we can simply read off the words

Recap: Probabilistic reasoning over time

- Markov Models

- Hidden Markov Models (HMMs)
  - Forward algorithm (repeated variable elimination) to infer belief state
  - Particle filtering (likelihood weighting with some tweaks)
  - Viterbi algorithm to infer most likely explanation

- Dynamic Bayes Nets
  - Particle filtering

End of Part II!

- Now we're done with our unit on probabilistic reasoning

- Last part of class: machine learning

Next: Machine learning

- Up until now: how to use a model to make optimal decisions

- Machine learning: how to acquire a model from data/experience
  - Learning parameters (e.g., probabilities)
  - Learning structure (e.g., BN graphs)
  - Learning hidden concepts (e.g., clustering)