Intelligent Systems (AI-2)

Computer Science cpsc422, Lecture 16

Oct, 9, 2019

Lecture Overview

Probabilistic temporal Inferences

- Filtering
- Prediction
- Smoothing (forward-backward)
- Most Likely Sequence of States (Viterbi)
- Approx. Inference In Temporal Models (Particle Filtering)

Most Likely Sequence

- Suppose that in the *rain* example we have the following *umbrella* observation sequence

  [true, true, false, true, true]

- Is the most likely state sequence?

  [rain, rain, no-rain, rain, rain]

- In this case you may have guessed right… but if you have more states and/or more observations, with complex transition and observation models…..

HMMs : most likely sequence (from 322)

Natural Language Processing: e.g., Speech Recognition

- *States:* phoneme \ word
- *Observations*: acoustic signal \ phoneme

HMM 1 HMM2

My name is

Bioinformatics: Gene Finding

- *States:* coding / non-coding region   x x ✓ ✓ ✓ x x
- *Observations:* DNA Sequences → A T C G G A A

For these problems the critical inference is:

find the most likely sequence of states given a sequence of observations

Viterbi Algo

Part-of-Speech (PoS) Tagging

- Given a text in natural language, label (*tag*) each word with its syntactic category
  - E.g, Noun, verb, pronoun, preposition, adjective, adverb, article, conjunction

- ***Input***
  - Brainpower, not physical plant, is now a firm's chief asset.

- ***Output***
  - Brainpower_NN ,_, not_RB physical_JJ plant_NN ,_, is_VBZ now_RB a_DT firm_NN 's_POS chief_JJ asset_NN ._.

Tag meanings

- NNP (Proper Noun singular), RB (Adverb), JJ (Adjective), NN (Noun sing. or mass), VBZ (Verb, 3 person singular present), DT (Determiner), POS (Possessive ending), . (sentence-final punctuation)

POS Tagging is very useful

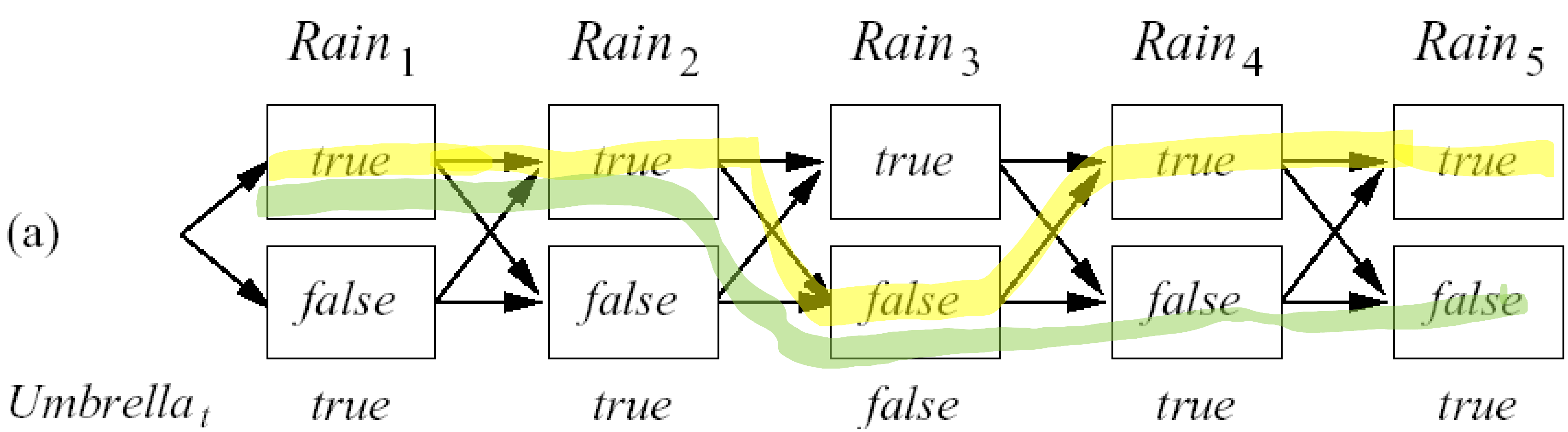
- As a basis for **Parsing** in NL understanding
- **Information Retrieval**
  - ✓ Quickly finding names or other phrases for information extraction
  - ✓ Select important words from documents (e.g., nouns)
- **Speech synthesis**: Knowing PoS produce more natural pronunciations
  - ✓ E.g,. Content (noun) vs. content (adjective); object (noun) vs. object (verb)

Most Likely Sequence (Explanation)

- ***Most Likely Sequence***: $\text{argmax}_{x_{1:T}} P(X_{1:T} \mid e_{1:T})$
- Idea
  - find the most likely path to each state in X_T
  - As for filtering etc. let's try to develop a recursive solution

$Rain_5 = true$

$Rain_5 = false$



Joint vs. Conditional Prob

- You have two binary random variables ***X*** and ***Y***

$$\text{argmax}_x\, P(X \mid Y{=}t) \;?\; \text{argmax}_x\, P(X, Y{=}t)$$

A. Different x

B. Same x

C. It depends

X	*Y*	*P(X , Y)*
t	t	.4
f	t	.2
t	f	.1
f	f	.3

x = t

for both

High level rationale

1. The sequence that is maximizing the conditional prob is the same that is maximizing the joint (see previous clicker question)
2. We will compute the max for the joint and by doing that we can then reconstruct the sequence that is maximizing the joint
3. Which is the same that is maximizing the conditional prob

Most Likely Sequence: Formal Derivation (step 2: compute the max for the joint)

$\max_{\mathbf{x_1,...x_t}} \mathbf{P}(\mathbf{x}_1,\dots\ \mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{e}_{1:t+1}) = \max_{\mathbf{x_1,...x_t}} \mathbf{P}(\mathbf{x}_1,\dots\ \mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{e}_{1:t}, \mathbf{e}_{t+1}) =$ Cond. Prob

$= \max_{\mathbf{x_1,...x_t}} P(\mathbf{e}_{t+1}|\mathbf{e}_{1:t}, \mathbf{x}_1,\dots\ \mathbf{x}_t, \mathbf{x}_{t+1})\ \mathbf{P}(\mathbf{x}_1,\dots\ \mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{e}_{1:t}) =$ Markov Assumption

$= \max_{\mathbf{x_1,...x_t}} P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1})\ \mathbf{P}(\mathbf{x}_1,\dots\ \mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{e}_{1:t}) =$ Cond. Prob

$= \max_{\mathbf{x_1,...x_t}} P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1})\ \mathbf{P}(\mathbf{x}_{t+1}|\ \mathbf{x}_1,\dots\ \mathbf{x}_t, \mathbf{e}_{1:t})\mathbf{P}(\mathbf{x}_1,\dots\ \mathbf{x}_t, \mathbf{e}_{1:t}) =$ Markov Assumption

$= \max_{\mathbf{x_1,...x_t}} \mathbf{P}(\mathbf{e_{t+1}}\ |\mathbf{x}_{t+1})\ \mathbf{P}(\mathbf{x}_{t+1}|\mathbf{x}_t)\ \mathbf{P}(\mathbf{x}_1,\dots\ \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{e}_{1:t}) =$ Move outside the max

$\mathbf{P}(\mathbf{e_{t+1}}\ |\mathbf{x}_{t+1}) \max_{\mathbf{x_t}} (\mathbf{P}(\mathbf{x}_{t+1}|\mathbf{x}_t) \max_{\mathbf{x_1,...x_{t-1}}} \mathbf{P}(\mathbf{x}_1,\dots\ \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{e}_{1:t}))$

Most Likely Sequence: Formal Derivation (step 2: compute the max for the joint)

$\max_{x_1,\ldots x_t} \mathbf{P}(\mathbf{x}_1,\ldots.\ \mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{e}_{1:t+1}) = \max_{x_1,\ldots x_t} \mathbf{P}(\mathbf{x}_1,\ldots.\ \mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{e}_{1:t}, \mathbf{e}_{t+1}) =$ Cond. Prob

$= \max_{x_1,\ldots x_t} P(\mathbf{e}_{t+1}|\mathbf{e}_{1:t}, \mathbf{x}_1,\ldots.\ \mathbf{x}_t, \mathbf{x}_{t+1})\ \mathbf{P}(\mathbf{x}_1,\ldots.\ \mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{e}_{1:t}) =$ Markov Assumption

$= \max_{x_1,\ldots x_t} P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1})\ \mathbf{P}(\mathbf{x}_1,\ldots.\ \mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{e}_{1:t}) =$ Cond. Prob

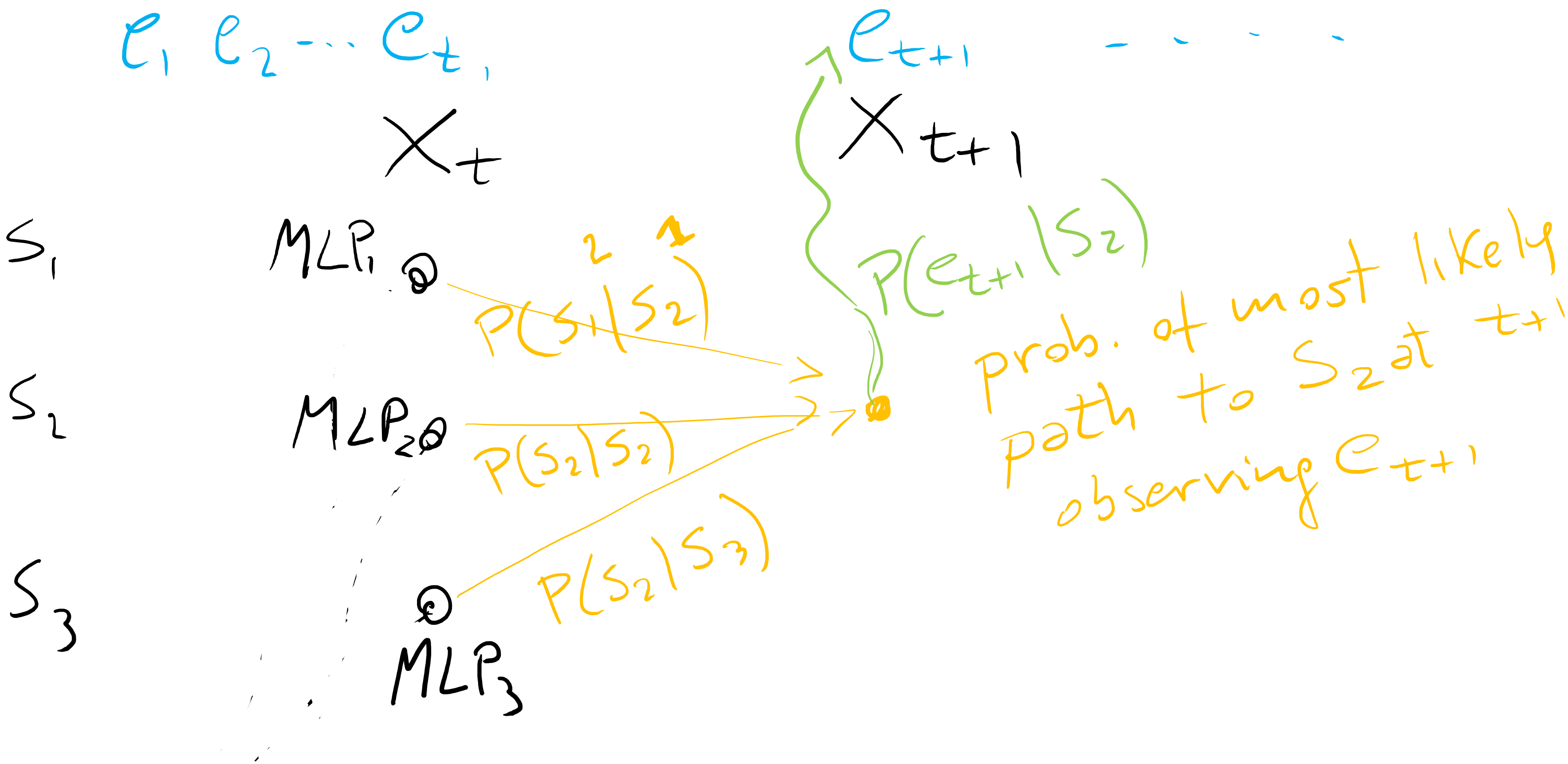
$= \max_{x_1,\ldots x_t} P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1})\ \mathbf{P}(\mathbf{x}_{t+1}|\ \mathbf{x}_1,\ldots.\ \mathbf{x}_t, \mathbf{e}_{1:t})\mathbf{P}(\mathbf{x}_1,\ldots.\ \mathbf{x}_t, \mathbf{e}_{1:t}) =$ Markov Assumption

$= \max_{x_1,\ldots x_t} \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{x}_{t+1})\ \mathbf{P}(\mathbf{x}_{t+1}|\mathbf{x}_t)\ \mathbf{P}(\mathbf{x}_1,\ldots.\ \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{e}_{1:t}) =$ Move outside the max

$\mathbf{P}(\mathbf{e}_{t+1}|\mathbf{x}_{t+1}) \max_{x_t} (\mathbf{P}(\mathbf{x}_{t+1}|\mathbf{x}_t) \max_{x_1,\ldots x_{t-1}} \mathbf{P}(\mathbf{x}_1,\ldots.\ \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{e}_{1:t}))$

Intuition behind solution

$$\mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{x}_{t+1}) \max_{\mathbf{x}_t} \left(\mathbf{P}(\mathbf{x}_{t+1} \mid \mathbf{x}_t) \max_{\mathbf{x}_1,\ldots,\mathbf{x}_{t-1}} \mathbf{P}(\mathbf{x}_1,\ldots,\mathbf{x}_{t-1},\mathbf{x}_t,\mathbf{e}_{1:t})\right)$$



prob. of the most likely path to state S_i after obs $e_{1:t}$

$$\mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{x}_{t+1}) \max_{\mathbf{x}_t} \left(\mathbf{P}(\mathbf{x}_{t+1} \mid \mathbf{x}_t) \max_{\mathbf{x}_1,\ldots\mathbf{x}_{t-1}} \mathbf{P}(\mathbf{x}_1,\ldots \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{e}_{1:t})\right)$$

The probability of the most likely path to S_2 at time *t+1* is:

$$P(e_{t+1} \mid S_2) * \max \left\{ \begin{array}{l} P(S_2 \mid S_1) * MLP_1 \\ P(S_2 \mid S_2) * MLP_2 \\ P(S_2 \mid S_3) * MLP_3 \end{array} \right\}$$

Most Likely Sequence

- Identical to filtering (notation warning: this is expressed for X_{t+1} instead of X_t , it does not make any difference!)

$$P(X_{t+1} \mid e_{1:t+1}) = \alpha\, P(e_{t+1} \mid X_{t+1}) \sum_{x_t} P(X_{t+1} \mid x_t)\, P(x_t \mid e_{1:t})$$

$$\max_{x_1,\ldots x_t} \mathbf{P}(\mathbf{x}_1,\ldots \mathbf{x}_t, \mathbf{X}_{t+1}, \mathbf{e}_{1:t+1})$$

$$= \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}) \max_{x_t} (\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{x}_t) \max_{x_1,\ldots x_{t-1}} \mathbf{P}(\mathbf{x}_1,\ldots \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{e}_{1:t}))$$

Recursive call

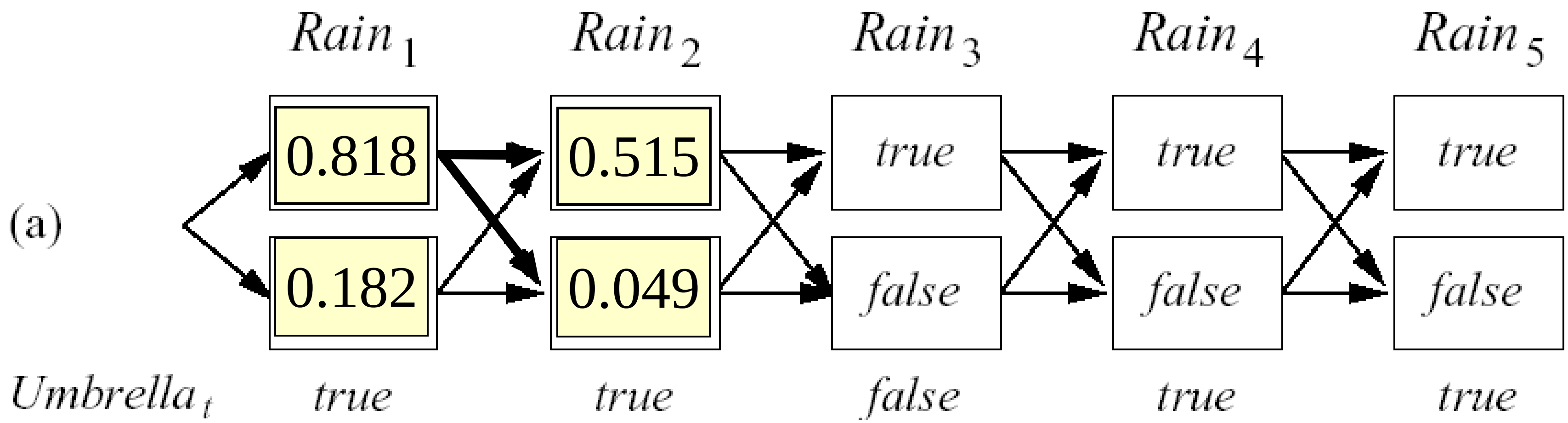
- $f_{1:t} = \mathbf{P}(\mathbf{X}_t \mid \mathbf{e}_{1:t})$ is replaced by
  - $m_{1:t} = \max_{x_1,\ldots x_{t-1}} P(\mathbf{x}_1,\ldots \mathbf{x}_{t-1}, \mathbf{X}_t, \mathbf{e}_{1:t})$ (*)

- the summation in the **filtering** equations is replaced by maximization in the **most likely sequence** equations

Rain Example

- $\max_{\mathbf{x}_1,\dots\mathbf{x}_t} \mathbf{P}(\mathbf{x}_1,\dots \mathbf{x}_t, \mathbf{X}_{t+1}, \mathbf{e}_{1:t+1}) = \mathbf{P}(\mathbf{e}_{t+1} | \mathbf{X}_{t+1}) \max_{\mathbf{x}_t} [(\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t)\ m_{1:t}]$

  $m_{1:t} = \max_{\mathbf{x}_1,\dots\mathbf{x}_{t-1}} \mathbf{P}(\mathbf{x}_1,\dots \mathbf{x}_{t-1}, \mathbf{X}_t, \mathbf{e}_{1:t})$



- $m_{1:1}$ is just $\mathbf{P}(R_1|u) = \langle 0.818, 0.182 \rangle$
- $m_{1:2}$ =

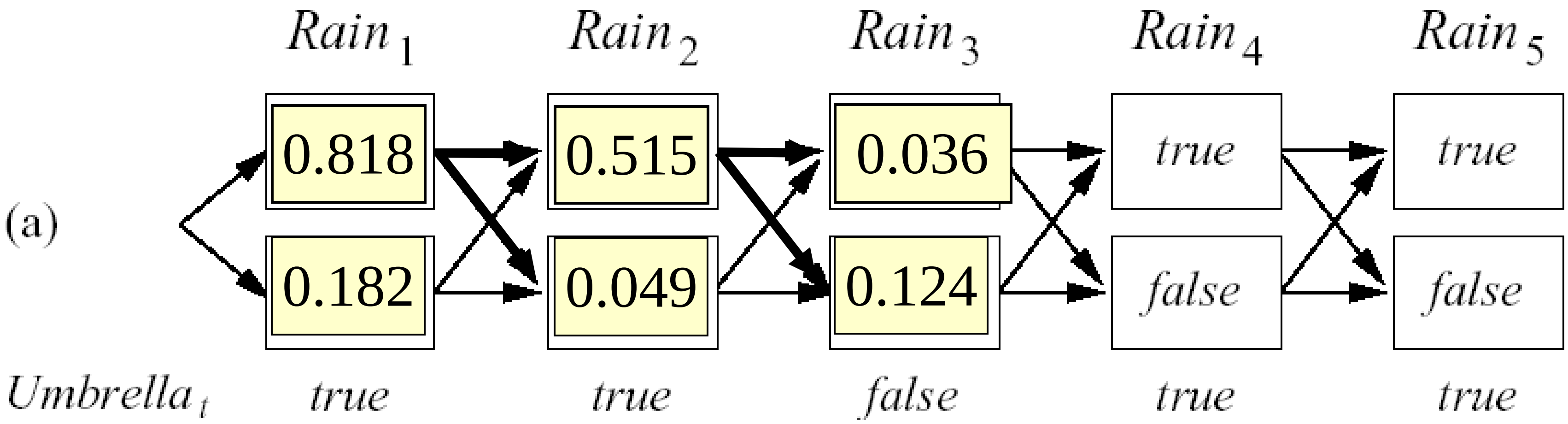
what is the most likely way to end up in Rain=T
from Rain=T or from Rain=F ?

$\mathbf{P}(u_2|R_2) \langle \max [P(r_2|r_1) * 0.818, P(r_2|\neg r_1)\ 0.182], \max [P(\neg r_2|r_1) * 0.818, P(\neg r_2|\neg r_1)\ 0.182] =$

$= \langle 0.9, 0.2 \rangle \langle \max(0.7*0.818, 0.3*0.182), \max(0.3*0.818, 0.7*0.182) =$

$= \langle 0.9, 0.2 \rangle * \langle 0.573, 0.245 \rangle = \langle 0.515, 0.049 \rangle$

Rain Example



$m_{\mathbf{1:3}} =$

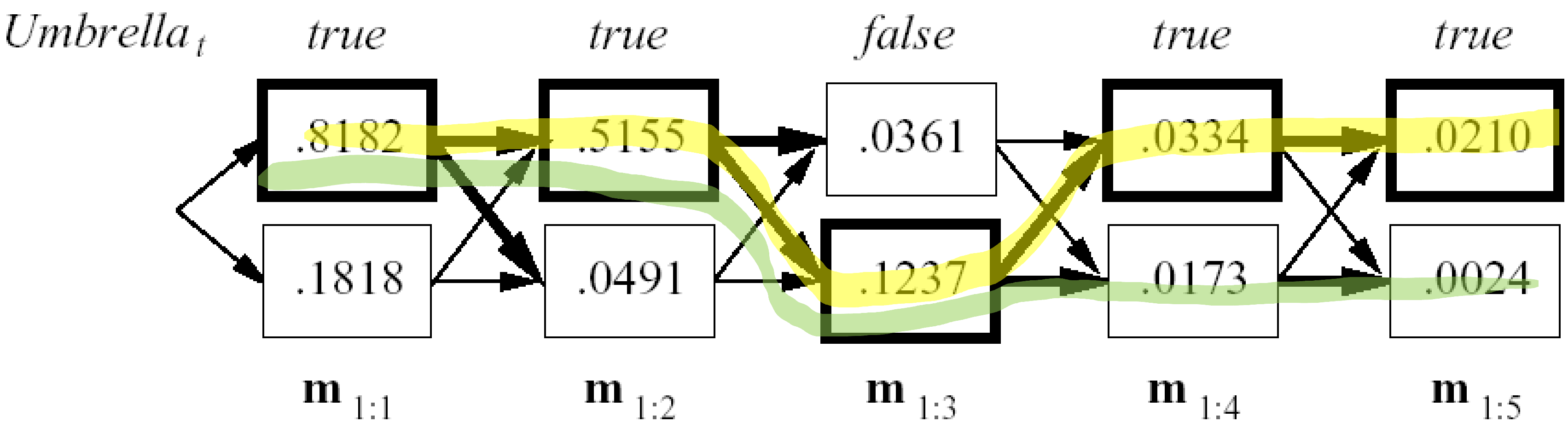
$\mathbf{P}(\neg u_{\mathbf{3}}|R_3) < \max [P(r_3|r_2) * 0.515, P(r_3|\neg r_2) * 0.049], \max [P(\neg r_3|r_2) * 0.515, P(\neg r_3|\neg r_2)\ 0.049) =$

$= <0.1, 0.8> < \max(0.7 * 0.515, 0.3 * 0.049), \max(0.3 * 0.515, 0.7 * 0.049) =$

$= <0.1, 0.8> * <0.36, 0.155> = <0.036, 0.124>$

Viterbi Algorithm

- Computes the most likely sequence to $\mathbf{X}_{t+1}$ by
  - running forward along the sequence
  - computing the *m* message at each time step
  - Keep back pointers to states that maximize the function
  - in the end the message has the prob. Of the most likely sequence to each of the final states
  - we can pick the most likely one and build the path by retracing the back pointers



Viterbi Algorithm: Complexity

T = number of time slices

S = number of states

- Time complexity?

A. $O(T^2 S)$ **B.** $O(T S^2)$ **C.** $O(T^2 S^2)$

- Space complexity

A. $O(T S)$ **B.** $O(T^2 S)$ **C.** $O(T^2 S^2)$

Lecture Overview

Probabilistic temporal Inferences

- Filtering
- Prediction
- Smoothing (forward-backward)
- Most Likely Sequence of States (Viterbi)
- **Approx. Inference In Temporal Models (Particle Filtering)**

Limitations of Exact Algorithms

- HMM has very large number of states

- Our temporal model is a Dynamic Belief Network with several “state” variables

Exact algorithms do not scale up ☹

What to do?

Approximate Inference

Basic idea:

- Draw N samples from known prob. distributions
- Use those samples to estimate unknown prob. distributions

Why sample?

- Inference: getting N samples is faster than computing the right answer (e.g. with Filtering)

Simple but Powerful Approach: Particle Filtering

Idea from Exact Filtering: should be able to compute $\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t+1})$ from $P(\mathbf{X}_t \mid \mathbf{e}_{1:t})$

".. One slice from the previous slice…"

Idea from Likelihood Weighting

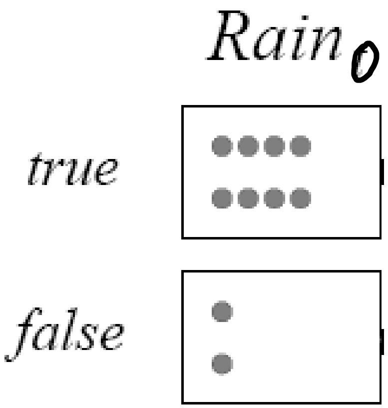
- Samples should be weighted by the probability of evidence given parents

New Idea: run multiple samples simultaneously through the network

Particle Filtering

- Run all **N samples together** through the network, one slice at a time

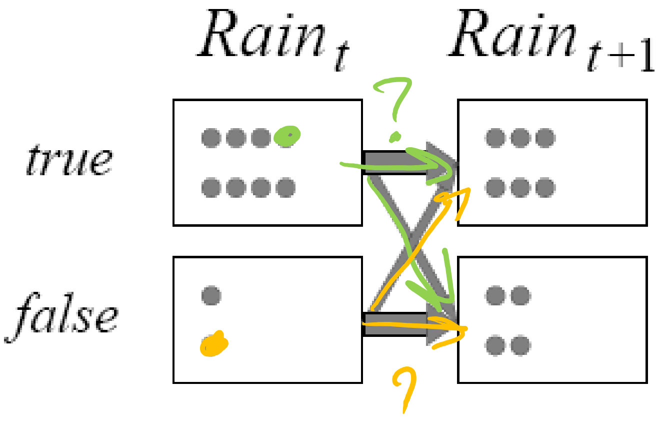
STEP 0: Generate a population on N initial-state samples by sampling from initial state distribution $P(\mathbf{X}_0)$



N = 10

Particle Filtering

STEP 1: Propagate each sample for $\boldsymbol{x_t}$ forward by sampling the next state value $\boldsymbol{x_{t+1}}$ based on $P(\boldsymbol{X_{t+1}} | \boldsymbol{X_t})$



(a) Propagate

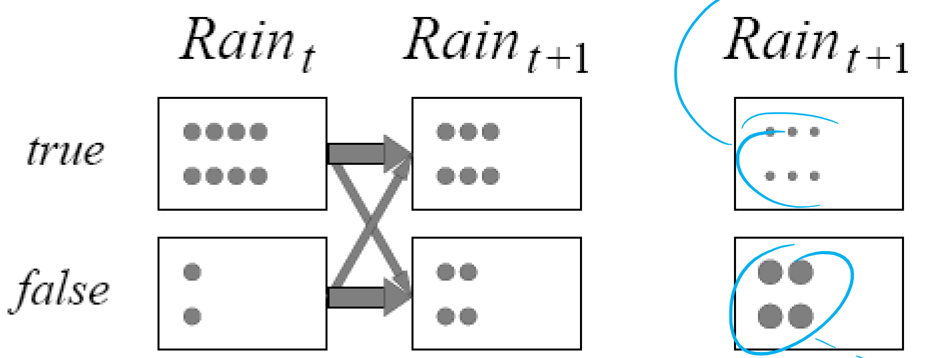
R_t	$P(R_{t+1}=t)$
t	0.7
f	0.3

Particle Filtering

STEP 2: **Weight each sample by the likelihood** it assigns to the evidence

- E.g. assume we observe ***not umbrella*** at t+1

weighted 0.1



(a) Propagate (b) Weight

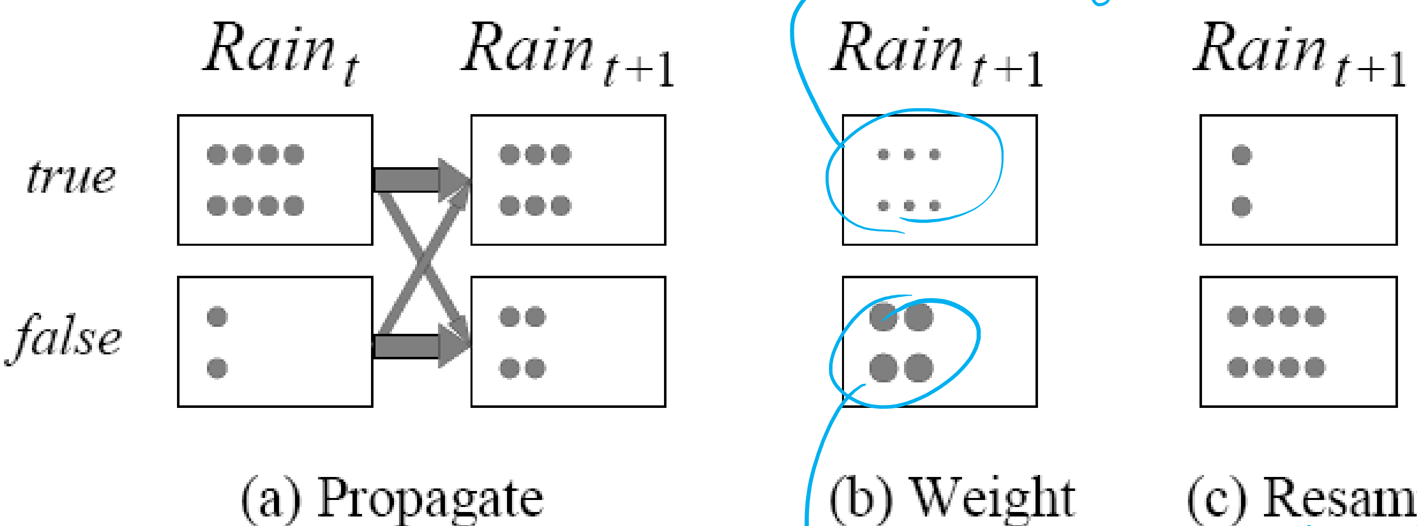
R_t	$P(u_t)$	$P(\neg u_t)$
t	0.9	0.1
f	0.2	0.8

weighted 0.8

Particle Filtering

STEP 3: Create a new population from the population at $\boldsymbol{X_{t+1}}$, *i.e. resample the population so that the probability that each sample is selected is proportional to its weight*

tot weight .6



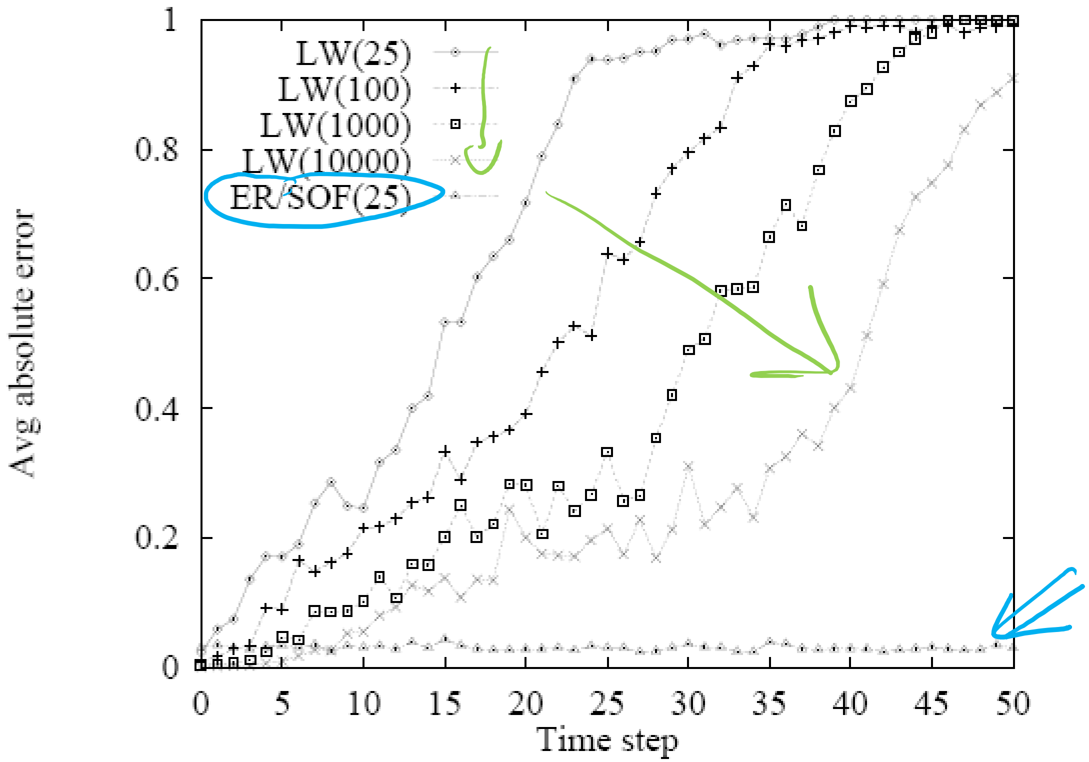
tot weight 3.2

$$\left(\frac{3.2}{3.8}, \frac{.6}{3.8} \right)$$

➢ Start the Particle Filtering cycle again from the new sample

Is PF Efficient?

In practice, approximation error of particle filtering remains bounded overtime



It is also possible to prove that the approximation maintains bounded error with high probability

(with specific assumption: probs in transition and sensor models >0 and <1)

422 big picture

StarAI (statistical relational AI)
Hybrid: Det +Sto

Prob CFG
Prob Relational Models
Markov Logics

Deterministic | Stochastic

Query

Logics
First Order Logics
Ontologies

- Full Resolution
- SAT

Belief Nets
Approx. : Gibbs
Markov Chains and HMMs
Forward, Viterbi….
Approx. : Particle Filtering
Undirected Graphical Models
Markov Networks
Conditional Random Fields

Planning

Markov Decision Processes and Partially Observable MDP

- Value Iteration
- Approx. Inference

Reinforcement Learning

Applications of AI

Representation
Reasoning Technique

Learning Goals for today's class

➢You can:

- Describe the problem of finding the most likely sequence of states (given a sequence of observations), derive its solution (Viterbi algorithm) by manipulating probabilities and applying it to a temporal model
- Describe and apply Particle Filtering for approx. inference in temporal models.

TODO for Mon

- **Keep working on Assignment-2: due Fri Oct 20**
- **Midterm : October 25**

TODO for Fri

- **Keep working on Assignment-2: due Fri Oct 18**
- **Midterm : October 25**