Intelligent Systems (AI-2)

Computer Science cpsc422, Lecture 15

Feb, 12, 2021

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, Prediction, Smoothing, Viterbi….
Approx. : Particle Filtering

Temporal Models

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

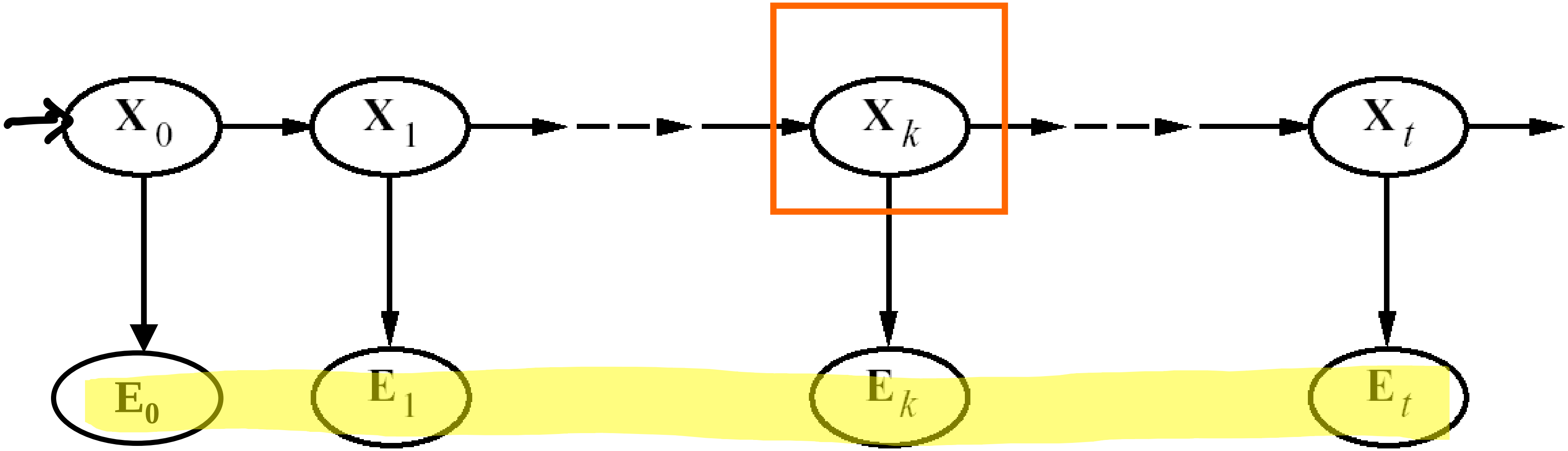
Lecture Overview

Probabilistic temporal Inferences

- Filtering
- Prediction
- **Smoothing (forward-backward)**
- **Most Likely Sequence of States (Viterbi)**

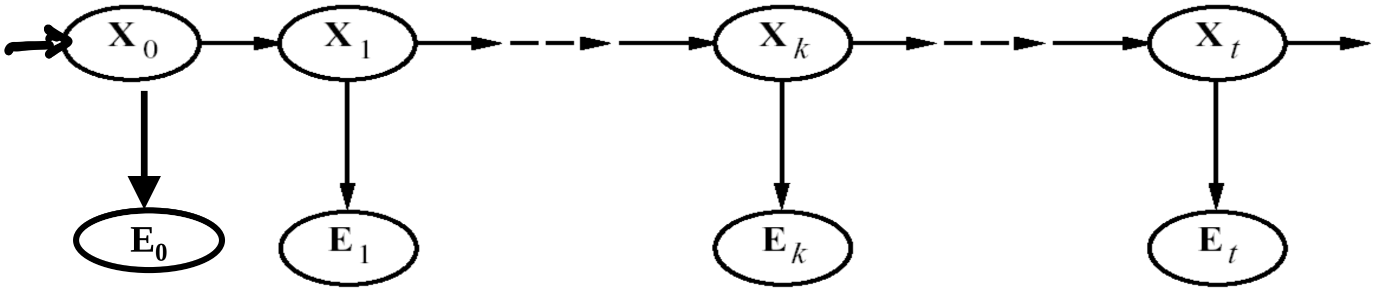
Smoothing

- ***Smoothing***: Compute the posterior distribution over a *past* state given all evidence to date
  - $P(X_k \mid \mathbf{e}_{0:t})$ for $1 \le k < t$



- ***To revise your estimates in the past based on more recent evidence***

Smoothing



- $P(X_k \mid e_{0:t}) = P(X_k \mid e_{0:k}, e_{k+1:t})$ dividing up the evidence

$= \alpha\, P(X_k \mid e_{0:k})\, P(e_{k+1:t} \mid X_k, e_{0:k})$ using…

$= \alpha\, P(X_k \mid e_{0:k})\, P(e_{k+1:t} \mid X_k)$ using…

iclicker

A. Bayes Rule

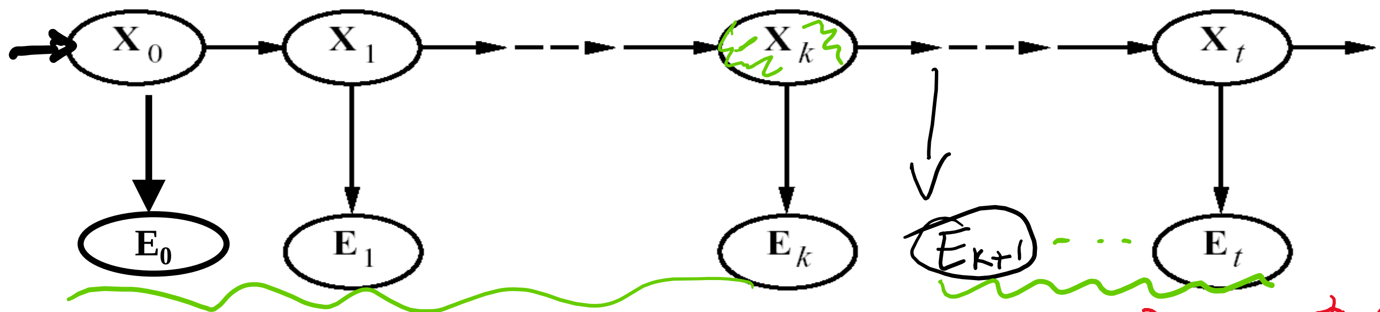
B. Cond. Independence

C. Product Rule

forward message from filtering up to state k, $f_{0:k}$

backward message, $b_{k+1:t}$ computed by a recursive process that runs backwards from *t*

Smoothing



$$P(A \mid C B) = \alpha P(B \mid A C) P(A \mid C)$$

- $\boldsymbol{P}(\boldsymbol{X}_k \mid \boldsymbol{e}_{0:t}) = \boldsymbol{P}(\boldsymbol{X}_k \mid \boldsymbol{e}_{0:k}, \boldsymbol{e}_{k+1:t})$ dividing up the evidence

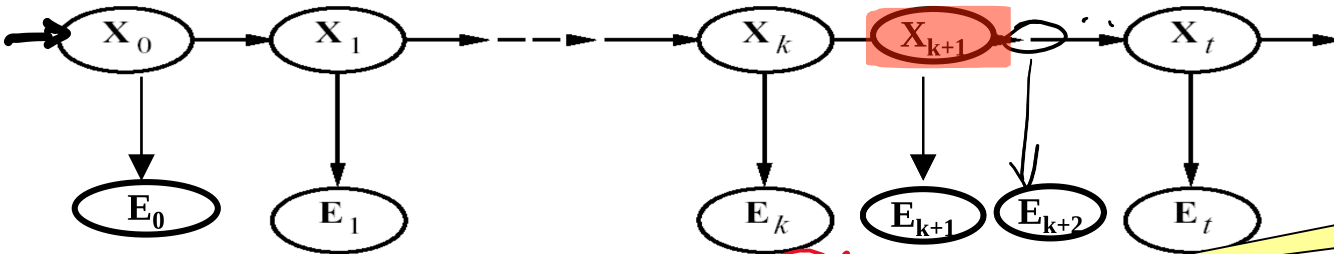
  $= \alpha\, \boldsymbol{P}(\boldsymbol{X}_k \mid \boldsymbol{e}_{0:k})\, \boldsymbol{P}(\boldsymbol{e}_{k+1:t} \mid \boldsymbol{X}_k, \boldsymbol{e}_{0:k})$ derived using Bayes Rule

  $= \alpha\, \boldsymbol{P}(\boldsymbol{X}_k \mid \boldsymbol{e}_{0:k})\, \boldsymbol{P}(\boldsymbol{e}_{k+1:t} \mid \boldsymbol{X}_k)$ By Conditional Independence

forward message from filtering up to state k, $\boldsymbol{f}_{0:k}$

backward message, $\boldsymbol{b}_{k+1:t}$ computed by a recursive process that runs backwards from *t*

Backward Message



Moving Conditioning

$$P(e_{k+1:t} \mid X_k) = \sum_{x_{k+1}} P(e_{k+1:t}, x_{k+1} \mid X_k) = \sum_{x_{k+1}} P(e_{k+1:t} \mid x_{k+1}, X_k)\, P(x_{k+1} \mid X_k) =$$

$$= \sum_{x_{k+1}} P(e_{k+1:t} \mid x_{k+1})\, P(x_{k+1} \mid X_k)$$ by Conditional Independence

$$= \sum_{x_{k+1}} P(e_{k+1}, e_{k+2:t} \mid x_{k+1})\, P(x_{k+1} \mid X_k)$$

Moving Conditioning

$$= \sum_{x_{k+1}} P(e_{k+1} \mid x_{k+1}, e_{k+2:t})\, P(e_{k+2:t} \mid x_{k+1})\, P(x_{k+1} \mid X_k)$$

because e_{k+1} and $e_{k+2:t}$, are conditionally independent given x_{k+1}

$$= \sum_{x_{k+1}} P(e_{k+1} \mid x_{k+1})\, P(e_{k+2:t} \mid x_{k+1})\, P(x_{k+1} \mid X_k)$$

sensor model

recursive call

transition model

- In message notation

$$b_{k+1:t} = \text{BACKWARD}\,(b_{k+2:t}, e_{k+1})$$

"moving" the conditioning

$$P(AB|C) = \frac{P(ABC)}{P(C)} * \frac{P(BC)}{P(BC)} =$$

$$= \frac{P(ABC)}{P(BC)} * \frac{P(BC)}{P(C)} =$$

$$= P(A|BC) * P(B|C)$$

Proof of equivalent statements

X and Y are conditional independent given Z

(1) If $P(X|YZ) = P(X|Z) \Rightarrow$

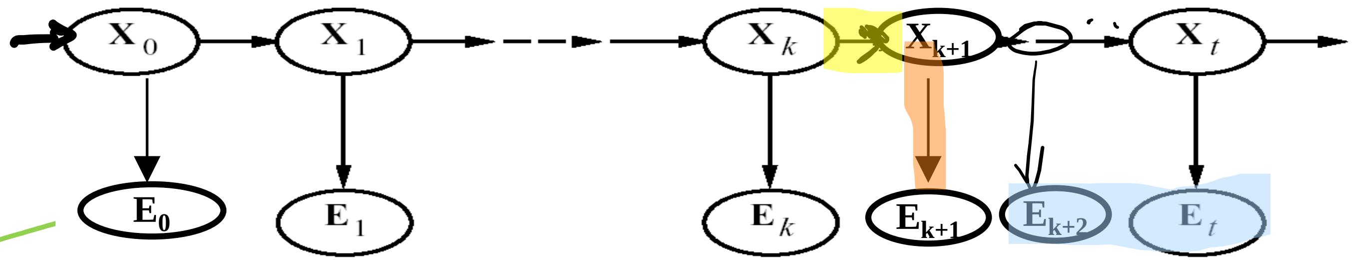
$$\Rightarrow \text{(A)} \quad \frac{P(X,Y,Z)}{P(Y,Z)} = \frac{P(X,Z)}{P(Z)} \Rightarrow$$

$$\Rightarrow \frac{P(X,Y,Z)}{P(X,Z)} = \frac{P(Y,Z)}{P(Z)} \Rightarrow P(Y|X,Z) = P(Y|Z) \quad \text{(2)}$$

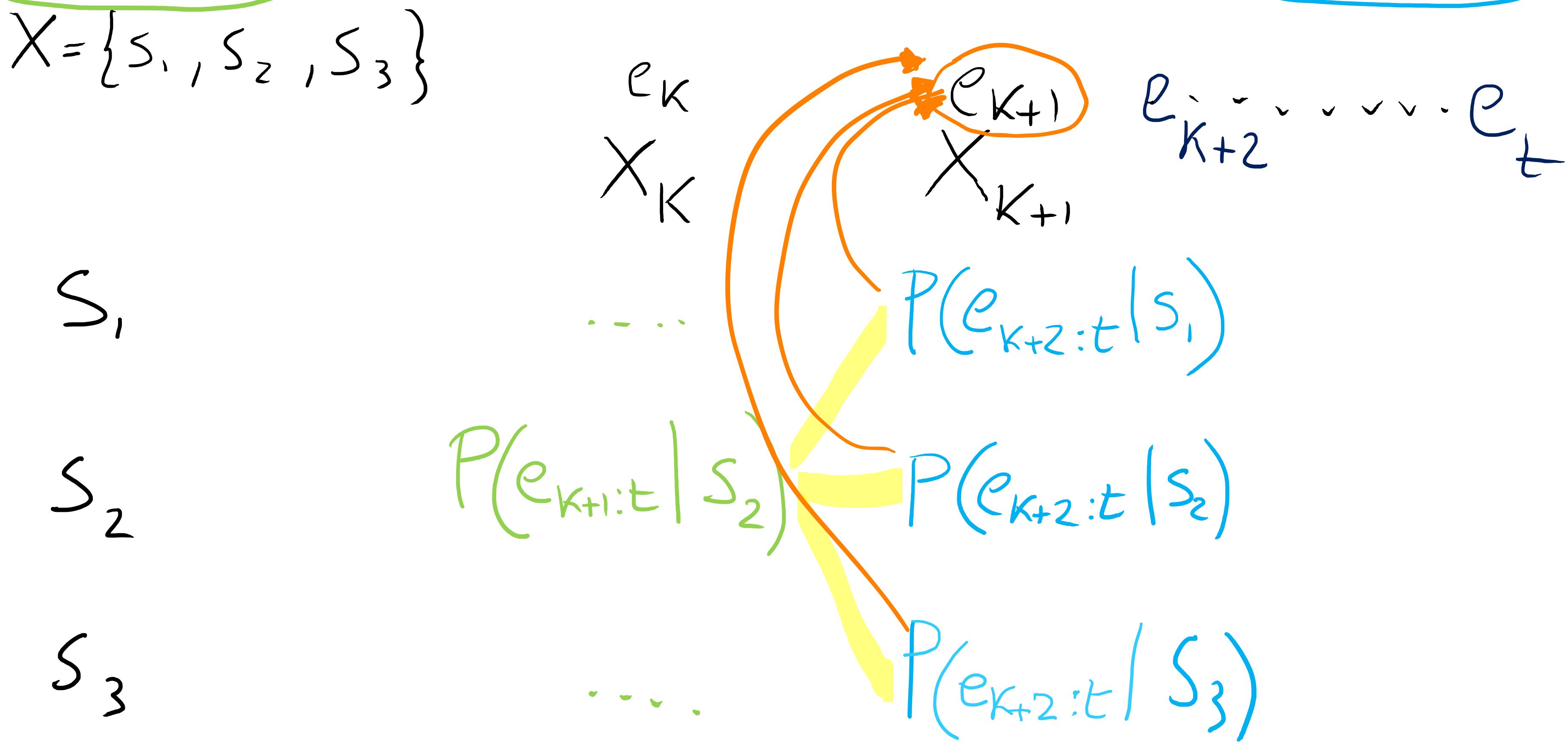
(3) $$P(X,Y|Z) = \frac{P(X,Y,Z)}{P(Z)} \overset{\text{from A}}{\Rightarrow} \frac{P(Y,Z)\,P(X,Z)}{P(Z)} \cdot \frac{1}{P(Z)}$$

$$= \frac{P(Y,Z)}{P(Z)} \cdot \frac{P(X,Z)}{P(Z)} = P(Y|Z) \cdot P(X|Z)$$

More Intuitive Interpretation (Example with three states)



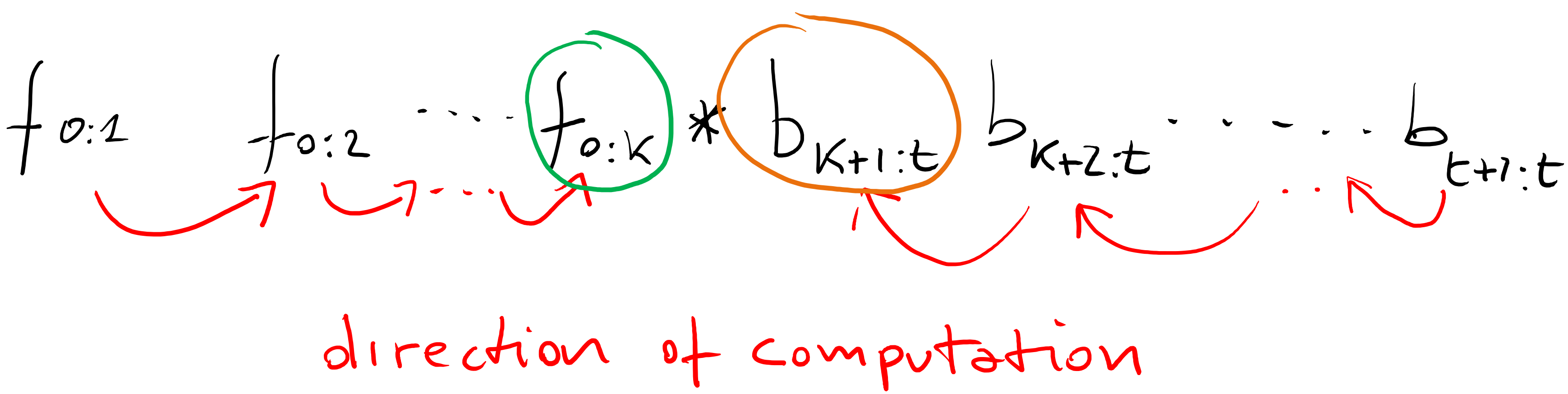
$$P(e_{k+1:t} \mid X_k) = \sum_{x_{k+1}} P(x_{k+1} \mid X_k) P(e_{k+1} | x_{k+1}) P(e_{k+2:t} | x_{k+1})$$



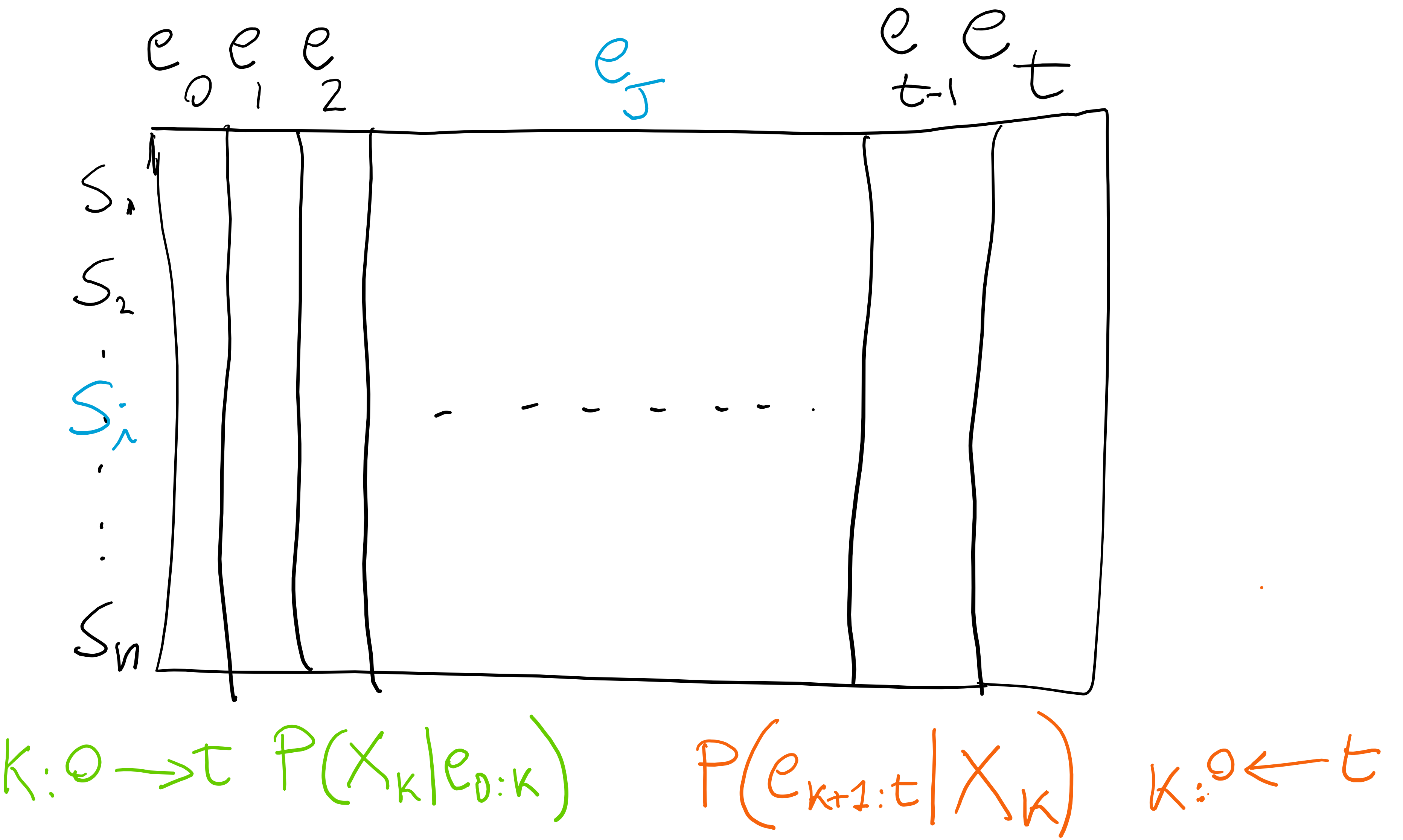
Forward-Backward Procedure

- To summarize, we showed
  - $P(X_k \mid e_{0:t}) = \alpha\, P(X_k \mid e_{0:k}\,)\, P(e_{k+1:t} \mid X_k)$
- Thus,
  - $\boldsymbol{P(X_k \mid e_{0:t})} = \alpha\, \boldsymbol{f}_{0:k}\, \boldsymbol{b}_{k+1:t}$

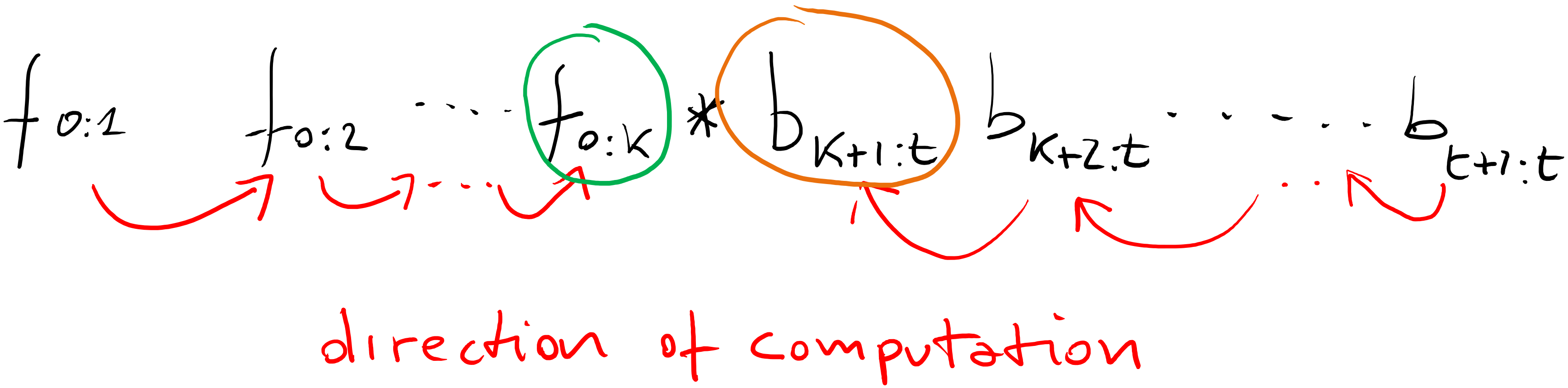
and this value can be computed by recursion through time, running forward from *0* to *k* and backwards from *t* to *k+1*



Forward-Backward Procedure fills a matrix n x t



How is it Backward initialized?



- The backwards phase is initialized with making an *unspecified* observation $\mathbf{e}_{t+1}$ at $t+1$……

$$\mathbf{b}_{t+1:t} = \mathbf{P}(\mathbf{e}_{t+1} | X_t) = \mathbf{P}(\textit{unspecified} \mid X_t) = ?$$

A. 0

B. 0.5

C. 1

How is it Backward initialized?

- The backwards phase is initialized with making an unspecified observation $\boldsymbol{e}_{t+1}$ at t+ 1……

  $\boldsymbol{b}_{t+1:t} = \mathbf{P}(\boldsymbol{e}_{t+1} | \boldsymbol{X}_t) = \mathbf{P}(\textit{unspecified} \mid \boldsymbol{X}_t) = 1$

- You will observe something for sure! It is only when you put some constraints on the observations that the probability becomes less than 1

Rain Example

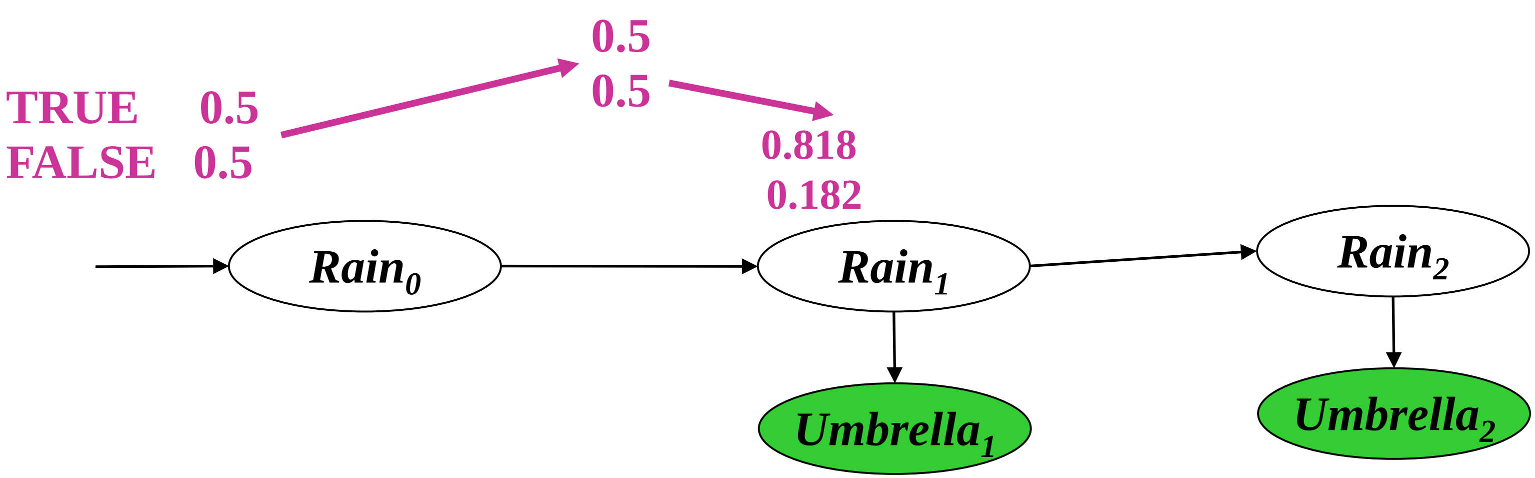
- Let's compute the probability of rain at t = 1, given umbrella observations at t=1 and t =2
- From $\mathbf{P}(\mathbf{X}_k \mid \mathbf{e}_{1:t}) = \alpha\, \mathbf{P}(\mathbf{X}_k \mid \mathbf{e}_{1:k})\, \mathbf{P}(\mathbf{e}_{k+1:t} \mid \mathbf{X}_k)$ we have

$$\mathbf{P}(R_1 \mid \mathbf{e}_{1:2}) = \mathbf{P}(R_1 \mid u_1, u_2) = \alpha\, \mathbf{P}(R_1 \mid u_1)\, \mathbf{P}(u_2 \mid R_1)$$

forward message from filtering up to state 1

backward message for propagating evidence backward from time 2

- $\mathbf{P}(R_1 \mid u_1) = \langle 0.818, 0.182 \rangle$ as it is the filtering to $t = 1$ that we did in lecture 14



Rain Example

- From $\mathbf{P}(\mathbf{e}_{k+1:t} \mid \mathbf{X}_k) = \sum_{\mathbf{x}_{k+1}} P(\mathbf{e}_{k+1}|\mathbf{x}_{k+1})\, P(\mathbf{e}_{k+2:t} |\mathbf{x}_{k+1})\, \mathbf{P}(\mathbf{x}_{k+1} \mid \mathbf{X}_k)$
- $\mathbf{P}(u_2 \mid R_1) = \sum P(u_2|r)\, P(|r)\, P(r \mid R_1) =$

$r \in r_2, \neg r_2$

Term corresponding to the Fictitious unspecified observation sequence $e_{3:2}$

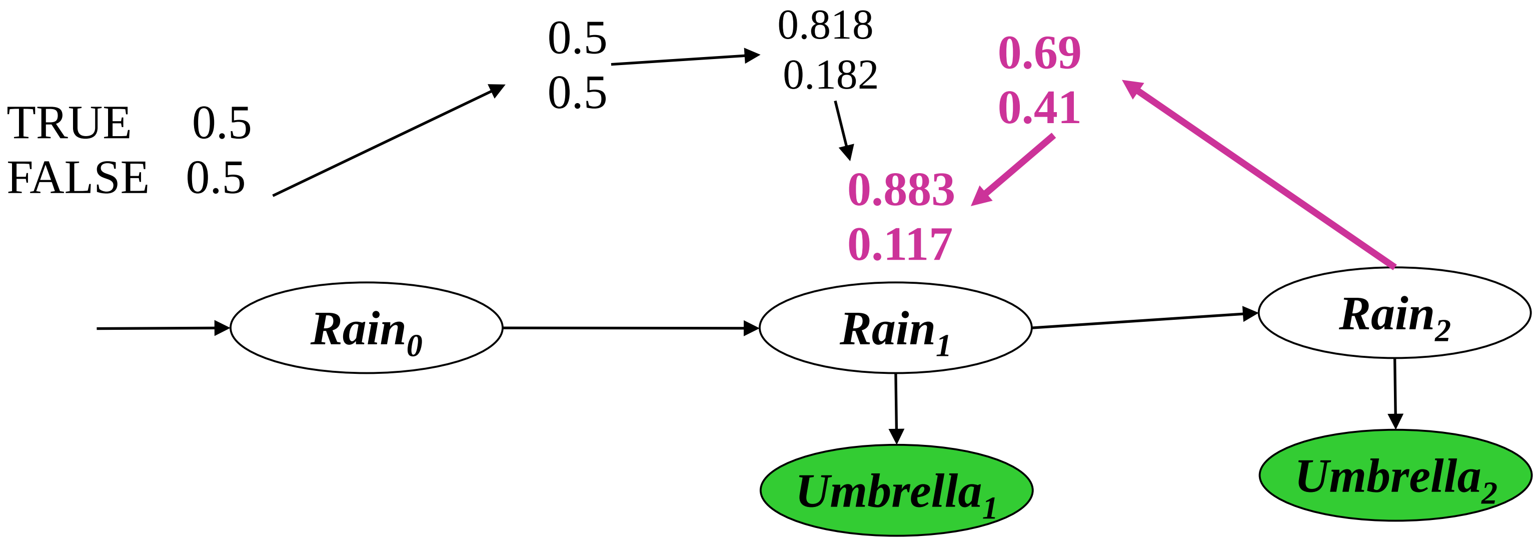
- $P(u_2|r_2)\, P(|r_2) < P(r_2 \mid r_1), P(r_2 \mid \neg r_1) > +$

$P(u_2|\neg r_2)\, P(|\neg r_2) < P(\neg r_2 \mid r_1), P(\neg r_2 \mid \neg r_1) >$

$= (0.9 * 1 * <0.7,0.3>) + (0.2 * 1 * <0.3, 0.7>) = <0.69,0.41>$

Thus

- $\alpha\, \mathbf{P}(R_1 | u_1)\, \mathbf{P}(u_2 \mid R_1) = \alpha <0.818, 0.182> * <0.69, 0.41> \sim <0.883, 0.117>$



Lecture Overview

Probabilistic temporal Inferences

- Filtering
- Prediction
- Smoothing (forward-backward)
- **Most Likely Sequence of States (Viterbi)**

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 HMM 2

Bioinformatics: Gene Finding

- *States:* coding / non-coding region
- *Observations:* DNA Sequences → ATCGGAA

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)
- **Word-sense disambiguation**
  - ✓ …I made her duck.. (*how many meanings does this sentence have*)?
- **Speech synthesis**: Knowing PoS produce more natural pronunciations
  - ✓ E.g,. Content (noun) vs. content (adjective); object (noun) vs. object (verb)

State of the art for sequence labeling (including POS)

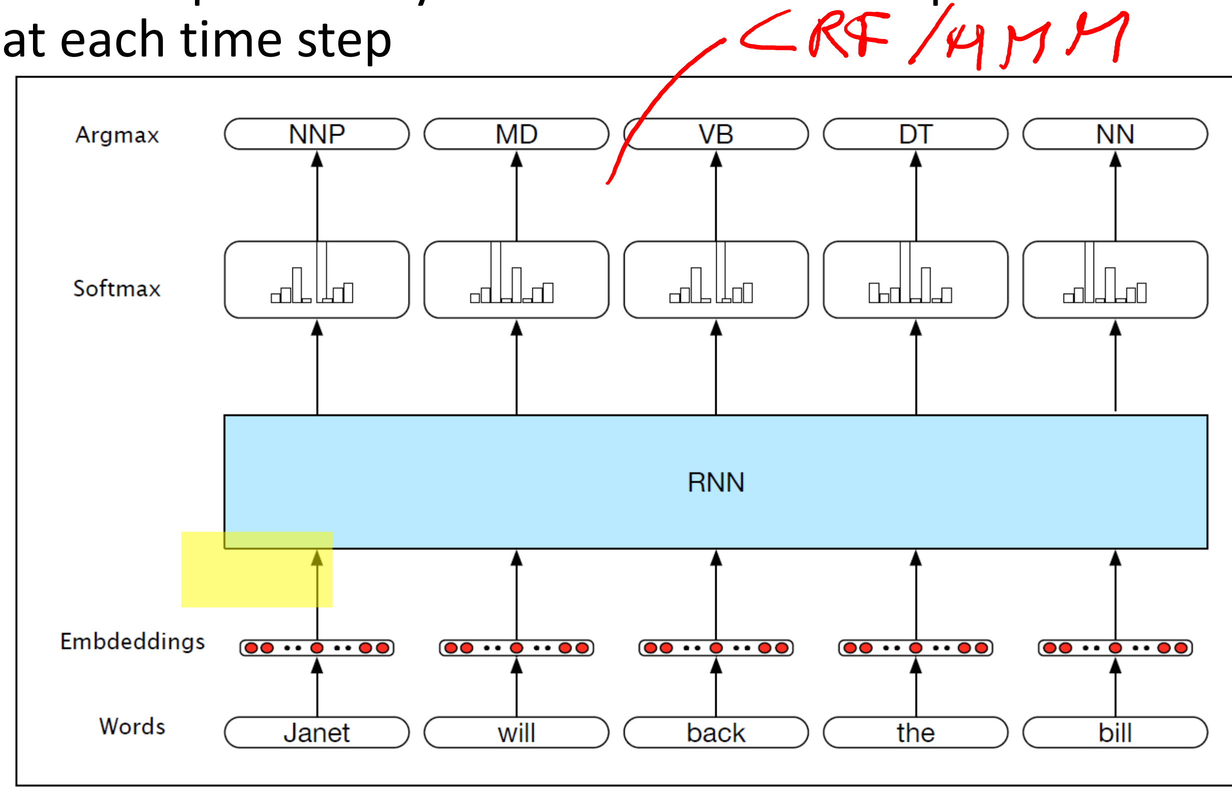
- **C**onditional **R**andom **F**ields (will see these in a few weeks - Viterbi can be applied)
- **R**ecurrent **N**eural **N**etworks (Slightly better performance than CRFs)
- CRF and RNN can be combined (see next slide)
- NOT REQUIRED FOR 422

Sequence Labeling (e.g., POS): SOTA ~2018 RNN + CRF with Viterbi

- **Input**: pre-trained embeddings
- **Output**: softmax layer provides a probability distribution over the part-of-speech tags as output at each time step

- Choosing max probability label for each item does not necessarily result in optimal (or even very good) tag sequence

- Combine with Viterbi for *most likely sequence*, usually implemented adding CRF layer



POS tagging state of the art + tools

- Neural Approaches (on several languages)
- Barbara Plank, Anders Søgaard, and Yoav Goldberg. Multilingual part-of-speech tagging with **bidirectional long short-term memory models and auxiliary loss**. ACL 2016.

Neural Approach to Semantic Role Labeling

BIO/IOB labeling..

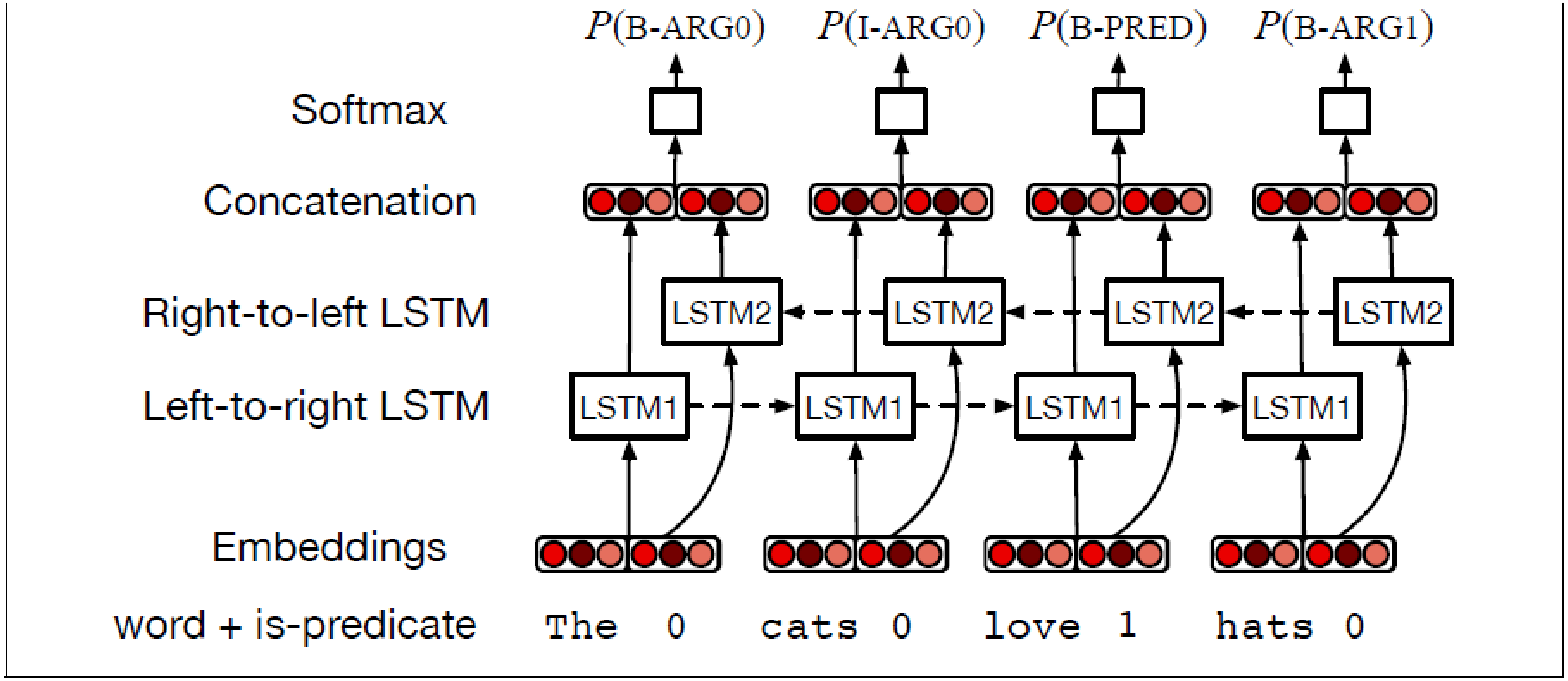


Figure 18.6 A bi-LSTM approach to semantic role labeling. Most actual networks are much deeper than shown in this figure; 3 to 4 bi-LSTM layers (6 to 8 total LSTMs) are common. The input is a concatenation of an embedding for the input word and an embedding of a binary variable which is 1 for the predicate to 0 for all other words. After He et al. (2017).

Global approach

Probably skip in class, but check textbook 18.6.2 if something similar needed for your project

- **Exploit global constraints between tags**; e.g., a tag I-ARG0 must follow another I-ARG0 or B-ARG0.
- **Apply Viterbi decoding**
  - start with the simple softmax output (the entire probability distribution over tags for each word)
  - Hard IOB constraints can act as the transition probabilities in the Viterbi decoding (Thus the transition from state I-ARG0 to I-ARG1 would have probability 0).
  - Alternatively, the training data can be used to learn bigram tag transition probabilities as if doing HMM decoding.

HMM — Here

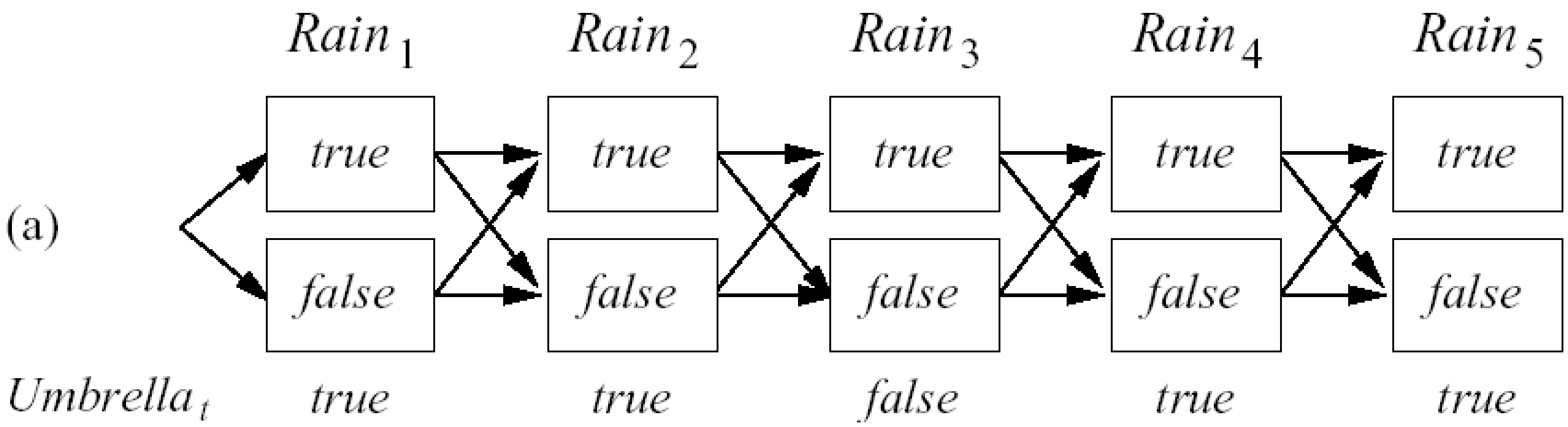
$P(s_0)$

$P(s_t | s_{t-1})$ OR

$P(o_t | s_t)$

Most Likely Sequence (Explanation)

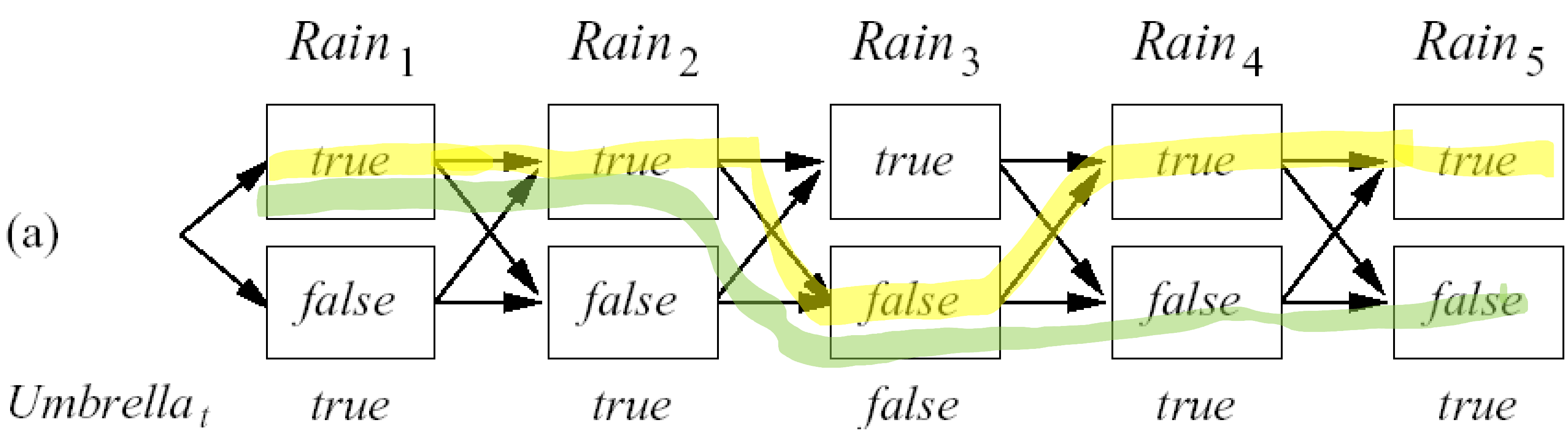
- ***Most Likely Sequence***: $\text{argmax}_{x_{1:T}} P(X_{1:T} \mid e_{1:T})$
- Idea
  - find the most likely path to each state in X_T
  - As for filtering etc. we will develop a recursive solution



Most Likely Sequence (Explanation)

- ***Most Likely Sequence***: $\text{argmax}_{\boldsymbol{x_{1:T}}} \boldsymbol{P(X_{1:T} \mid e_{1:T})}$
- Idea
  - find the most likely path to each state in $\boldsymbol{X_T}$
  - As for filtering etc. we will develop a recursive solution

Rain5 = true
Rain5 = false



Learning Goals for today's class

➢You can:

- Describe the **smoothing problem** and derive a solution by manipulating probabilities
- Describe the problem of finding the **most likely sequence of states** (given a sequence of observations)
- Derive recursive solution (if time)

TODO for Mon
(not this coming week)

- Keep working on Assignment-2: due Mon March 1
- Midterm : Mon March 8