UNIVERSITY OF LINCOLN

CMP9132M

Advanced Artificial Intelligence

Lecture 7: “Hidden Markov Models II”

Nicola Bellotto & Heriberto Cuayahuitl

Filtering

UNIVERSITY OF LINCOLN

$$f_t = P(x_t \mid e_1, e_2, \ldots, e_t)$$
$$= \alpha_t P(e_t \mid x_t) \Sigma_{x_{t-1}} [P(x_t \mid x_{t-1}) P(x_{t-1} \mid e_1, e_2, \ldots, e_{t-1})]$$
$$= \alpha_t P(e_t \mid x_t) \Sigma_{x_{t-1}} [P(x_t \mid x_{t-1}) f_{t-1}]$$
$$= \mathbf{FORWARD}(f_{t-1}, e_t)$$

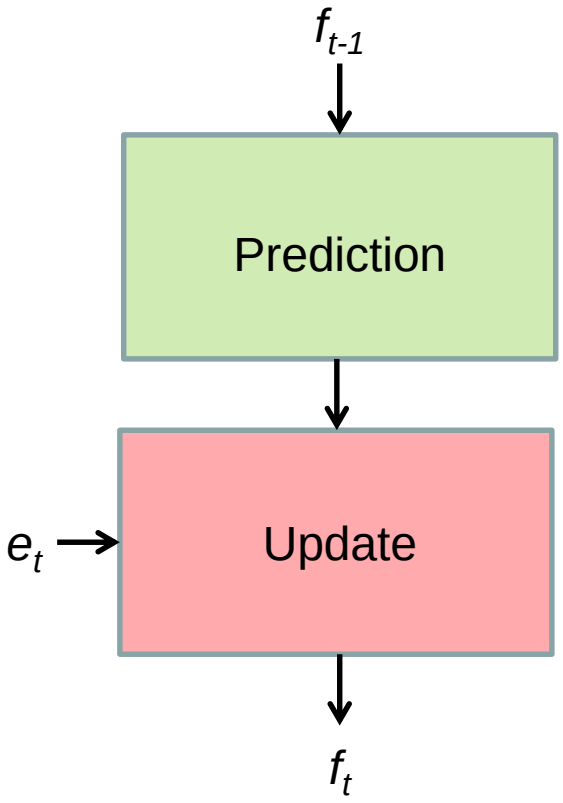
- Note that **FORWARD** consists of two main steps:
  - Prediction based on previous estimate

$$\Sigma_{x_{t-1}} [P(x_t \mid x_{t-1}) P(x_{t-1} \mid e_1, e_2, \ldots, e_{t-1})]$$

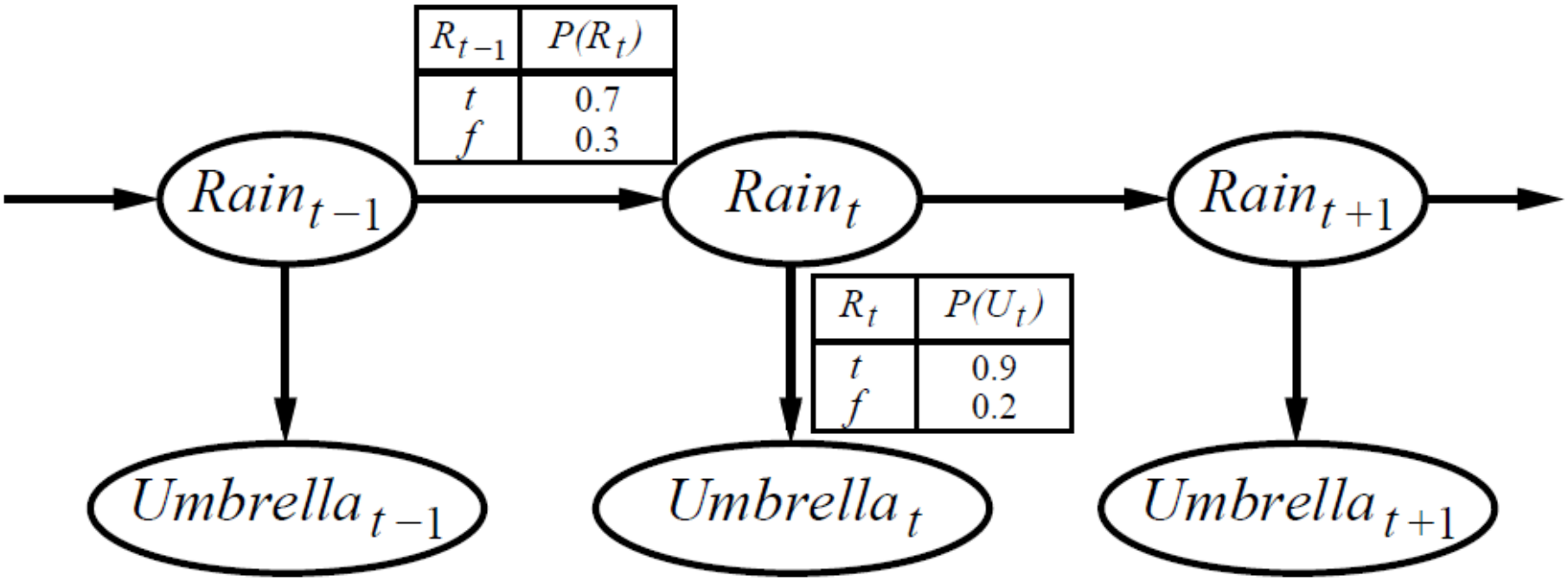
  - Update based on current observation (+ normalisation)

$$P(e_t \mid x_t)$$

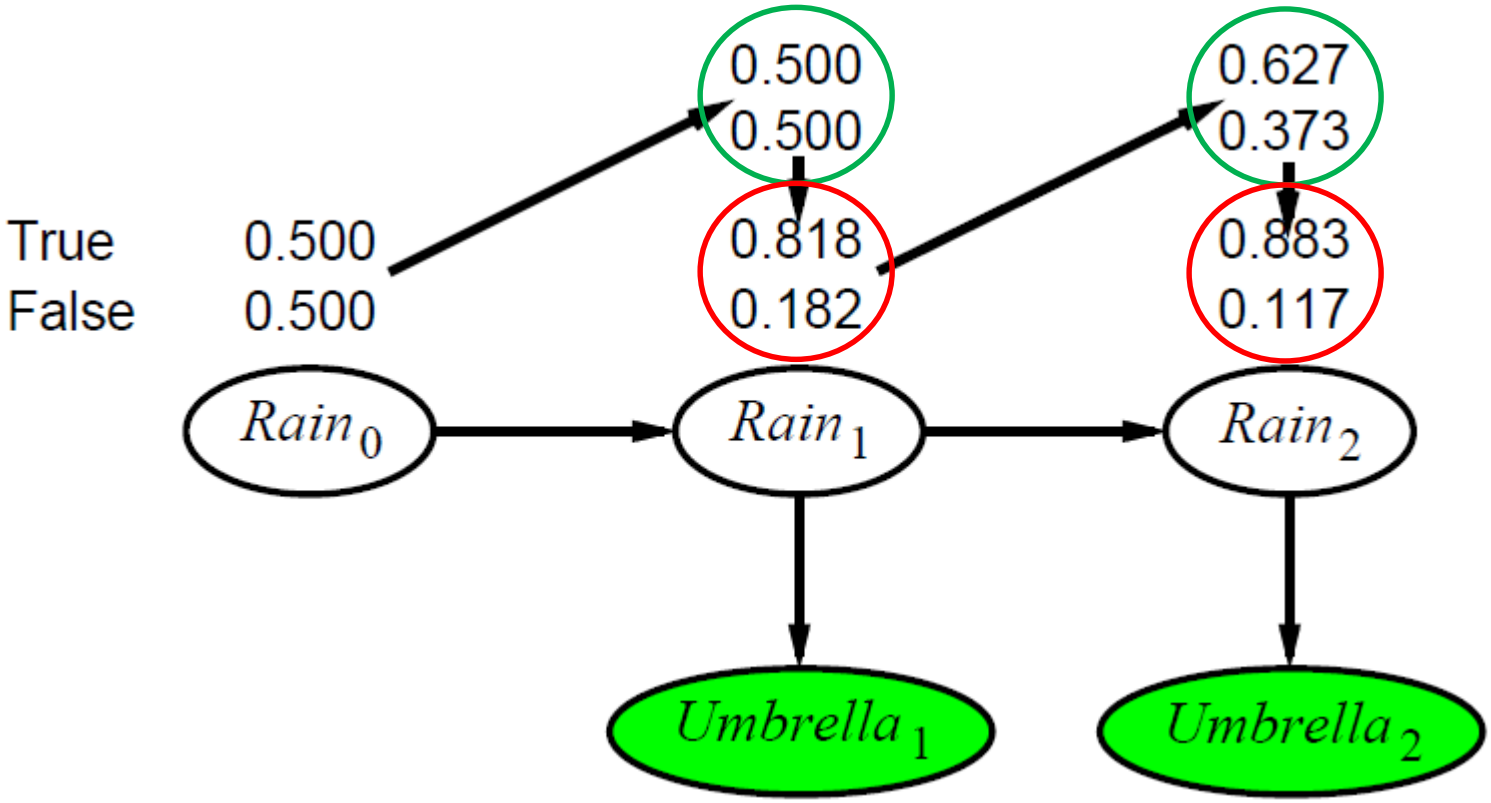
Prediction-Update



Filtering: weather example



Filtering: weather example



Matrix representation

UNIVERSITY OF LINCOLN

- Simpler way to work with HMM using linear algebra

Rain

$$X_t = \begin{bmatrix} True \\ False \end{bmatrix}$$

X_t

$$\mathbf{T} = \mathbf{P}(X_t \mid X_{t-1}) = \begin{bmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{bmatrix}$$ X_{t-1}

$$\mathbf{O}_t = \mathbf{P}(e_t = True \mid X_t) = \begin{bmatrix} 0.9 & 0 \\ 0 & 0.2 \end{bmatrix} \quad \text{or} \quad \mathbf{O}_t = \mathbf{P}(e_t = False \mid X_t) = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.8 \end{bmatrix}$$

Umbrella

No umbrella

Matrix representation

UNIVERSITY OF LINCOLN

$f_t = P(x_t \mid e_1, e_2, \ldots, e_t)$

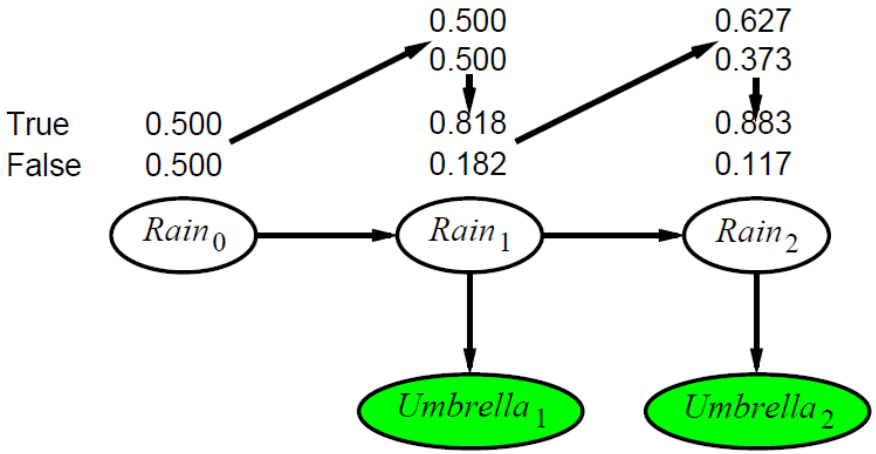
$= \alpha_t P(e_t \mid x_t) \Sigma_{x_{t-1}} [P(x_t \mid x_{t-1}) f_{t-1}]$

$= \mathbf{FORWARD}(f_{t-1}, e_t)$

- The same equation can be rewritten as

$$\mathbf{f}_t = \alpha_t \mathbf{O}_t \mathbf{T}^T \mathbf{f}_{t-1}$$

Matrix representation



$$\mathbf{f}_t = \alpha_t \mathbf{O}_t \mathbf{T}^T \mathbf{f}_{t-1}$$

in case $e_1 = True$

$$\mathbf{f}_1 = \alpha_t \begin{bmatrix} 0.9 & 0 \\ 0 & 0.2 \end{bmatrix} \begin{bmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} = \alpha_t \begin{bmatrix} 0.45 \\ 0.1 \end{bmatrix} = \begin{bmatrix} 0.818 \\ 0.182 \end{bmatrix}$$

in case $e_2 = True$

$$\mathbf{f}_2 = \alpha_t \begin{bmatrix} 0.9 & 0 \\ 0 & 0.2 \end{bmatrix} \begin{bmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{bmatrix} \begin{bmatrix} 0.818 \\ 0.182 \end{bmatrix} = \alpha_t \begin{bmatrix} 0.565 \\ 0.075 \end{bmatrix} = \begin{bmatrix} 0.883 \\ 0.117 \end{bmatrix}$$

Observation sequences

UNIVERSITY OF LINCOLN

- The probability of a sequence of observations $\{e_1, e_2, e_3, \ldots, e_t\}$ is equal to the sum of the probabilities of the sequence for each possible ending state x_t

- More in general, given $E_t = \{e_1, \ldots, e_t\}$

$$P(E_t) = \Sigma_{x_t} P(E_t, x_t)$$

Observation sequences

UNIVERSITY OF LINCOLN

- Let's call $s_t = P(E_t, x_t)$. Then

$$
\begin{aligned}
s_t &= \Sigma_{x_{t-1}} P(E_t, x_{t-1}, x_t) \\
&= \Sigma_{x_{t-1}} P(E_{t-1}, e_t, x_{t-1}, x_t) \\
&= \Sigma_{x_{t-1}} P(e_t \mid \cancel{E_{t-1}, x_{t-1}}, x_t)\, P(E_{t-1}, x_{t-1}, x_t) \\
&= P(e_t \mid x_t)\, \Sigma_{x_{t-1}} P(x_t \mid \cancel{E_{t-1}}, x_{t-1})\, P(E_{t-1}, x_{t-1}) \\
&= P(e_t \mid x_t)\, \Sigma_{x_{t-1}} P(x_t \mid x_{t-1})\, s_{t-1}
\end{aligned}
$$

- … and we end up again with a recursive algorithm (like filtering, without normalization)

Most Likely Explanation

UNIVERSITY OF LINCOLN

- Suppose we have the following sequence of umbrella observations (emissions)

  $[e_1, e_2, e_3, e_4, e_5] = [true, true, false, true, true]$

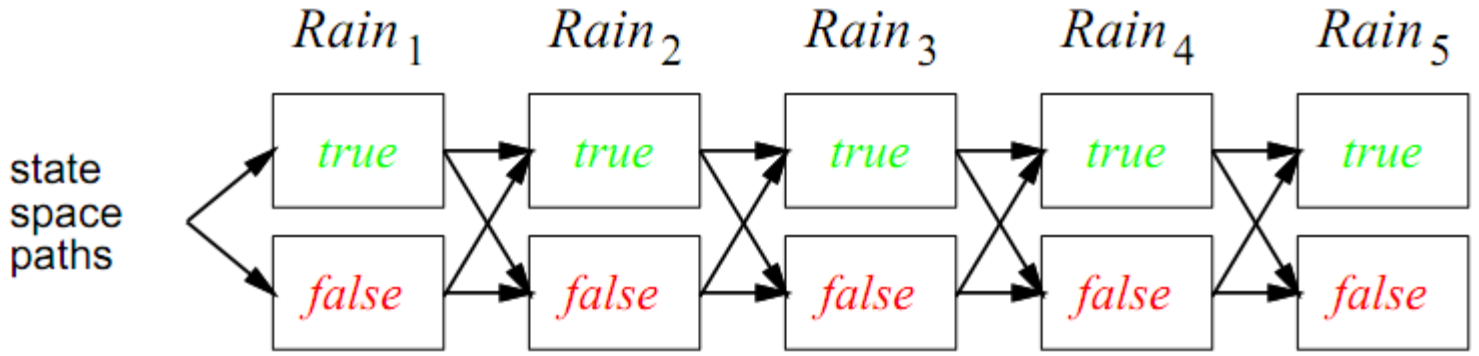
- There are $\mathbf{2^5}$ possible weather (hidden) state sequences generating such observations

- What is the most likely sequence of states explaining our observations? (without enumerating all the possible sequences)

$$\max_{x_1 \ldots x_t} P(x_1, \ldots, x_t \mid e_1, \ldots, e_t) = ?$$

Viterbi Algorithm

UNIVERSITY OF LINCOLN



- The likelihood of any path is the product of the transition and the observation probabilities along that path
- E.g., the most likely path reaching x_5 = *True* (i.e. $Rain_5$ = *True*) will be the most likely path to reach **some** state x_4, followed by a transition to x_5
- The state x_4 part of the path to x_5 = *True* is the one that maximises the path's likelihood
- The same for $x_3 \rightarrow x_4$, for $x_2 \rightarrow x_3$, etc. $\Rightarrow$ recursive algorithm!!

Viterbi Algorithm

UNIVERSITY OF LINCOLN

- Similarly to **FORWARD** in filtering, we can compute for each possible state at time *t*

$m_t = \max_{x_1 \ldots x_{t-1}} P(x_1, \ldots x_{t-1}, x_t \mid e_1, e_2, \ldots, e_t)$

$= P(e_t \mid x_t) \max_{x_{t-1}} [P(x_t \mid x_{t-1}) \max_{x_1 \ldots x_{t-2}} P(x_1, \ldots x_{t-2}, x_{t-1} \mid e_1, e_2, \ldots, e_{t-1})]$

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

$= \mathbf{MAXPATH}(m_{t-1}, e_t)$

- Note the analogy with the **FORWARD** case

$f_t = \alpha_t P(e_t \mid x_t) \Sigma_{x_{t-1}} [P(x_t \mid x_{t-1}) f_{t-1}]$

$= \mathbf{FORWARD}(f_{t-1}, e_t)$

Viterbi Algorithm

UNIVERSITY OF LINCOLN

- **MAXPATH** applies for $t \geq 2$

- For $t = 1$, it's the same as "filtering"

$$m_1 = P(x_1 \mid e_1) = \alpha_1 P(e_1 \mid x_1) P(x_1)$$

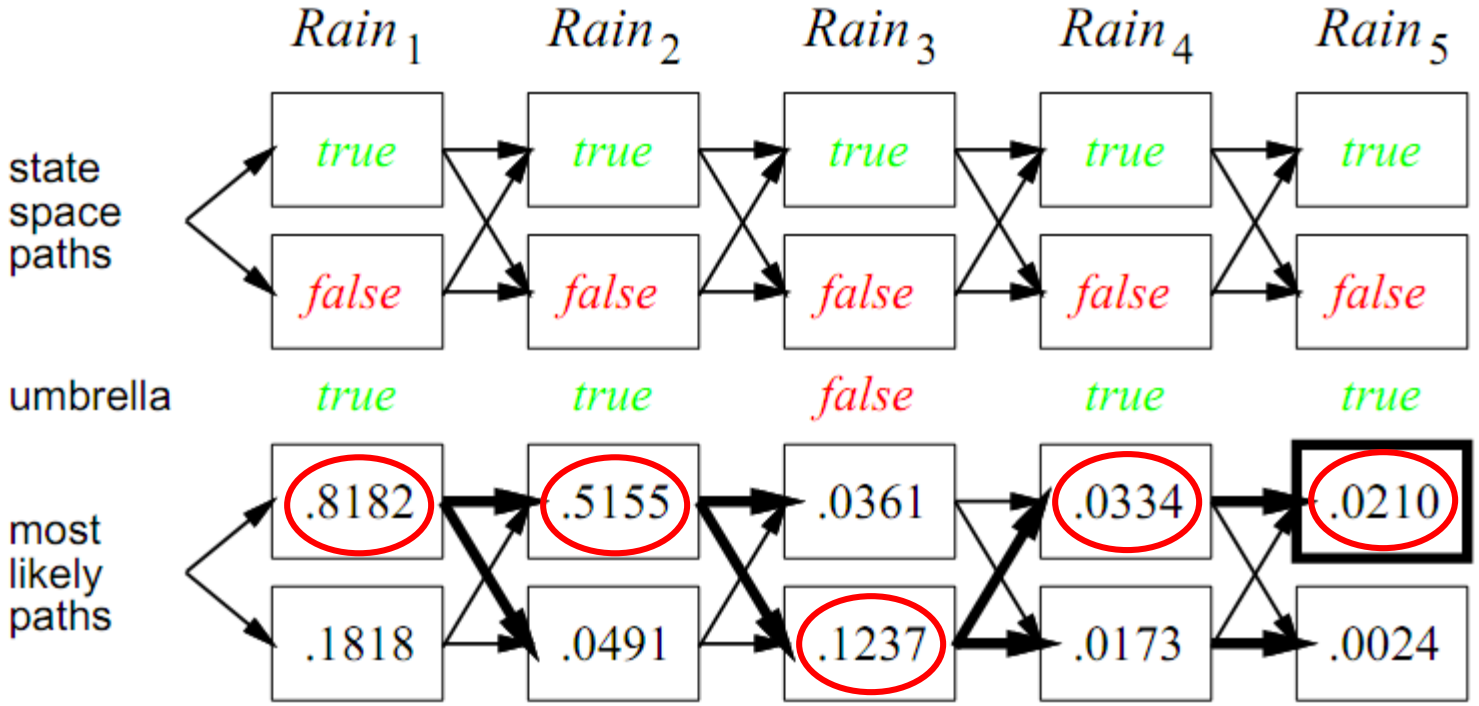
- where α_1 is a normalisation factor and $P(x_1)$ is the initial state, e.g. [0.5 0.5]'

- Note there are no more normalisations for $t \geq 2$

Viterbi Algorithm

UNIVERSITY OF LINCOLN

- As with **FORWARD**, we compute all the state probabilities (left to right) using **MAXPATH**, storing for each state the best state leading to it (bold arrows below)
- The most likely sequence then goes backwards through the bold arrows from the best final state



Conclusions

UNIVERSITY OF LINCOLN

- Suggested reading
  - Chapter 15 of Russell & Norvig
- Next week
  - No lecture!
- Any question?