Hidden Markov Models.

Petr Pošík

Czech Technical University in Prague
Faculty of Electrical Engineering
Dept. of Cybernetics

Markov Models

Reasoning over Time or Space

In areas like

- speech recognition,
- robot localization,
- medical monitoring,
- language modeling,
- DNA analysis,
- ...,

we want to *reason about a sequence* of observations.

Reasoning over Time or Space

In areas like

- speech recognition,
- robot localization,
- medical monitoring,
- language modeling,
- DNA analysis,
- . . . ,

we want to *reason about a sequence* of observations.

We need to **introduce time (or space)** into our models:

- A *static* world is modeled using a variable for each of its aspects which are of interest.
- A *changing* world is modeled using these variables *at each point in time*. The world is viewed as a sequence of *time slices*.
- Random variables form sequences in time or space.

Reasoning over Time or Space

In areas like

- speech recognition,
- robot localization,
- medical monitoring,
- language modeling,
- DNA analysis,
- . . . ,

we want to *reason about a sequence* of observations.

We need to **introduce time (or space)** into our models:

- A *static* world is modeled using a variable for each of its aspects which are of interest.
- A *changing* world is modeled using these variables *at each point in time*. The world is viewed as a sequence of *time slices*.
- Random variables form sequences in time or space.

Notation:

- X_t is the set of variables describing the world **state** at time t.
- X_a^b is the set of variables from X_a to X_b.
- E.g., X_1^t corresponds to variables $X_1, \ldots, X_t$.

Reasoning over Time or Space

In areas like

- speech recognition,
- robot localization,
- medical monitoring,
- language modeling,
- DNA analysis,
- $\ldots$,

we want to *reason about a sequence* of observations.

We need to **introduce time (or space)** into our models:

- A *static* world is modeled using a variable for each of its aspects which are of interest.
- A *changing* world is modeled using these variables *at each point in time*. The world is viewed as a sequence of *time slices*.
- Random variables form sequences in time or space.

Notation:

- X_t is the set of variables describing the world **state** at time t.
- X_a^b is the set of variables from X_a to X_b.
- E.g., X_1^t corresponds to variables $X_1, \ldots, X_t$.

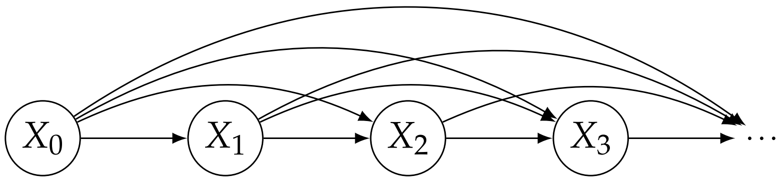
We need a way to specify joint distribution over a large number of random variables using assumptions suitable for the fields mentioned above.

Markov models

Transition model

- In general, it specifies the probability distribution over the current state X_t given all the previous states X_0^{t-1}:

$$P(X_t | X_0^{t-1})$$

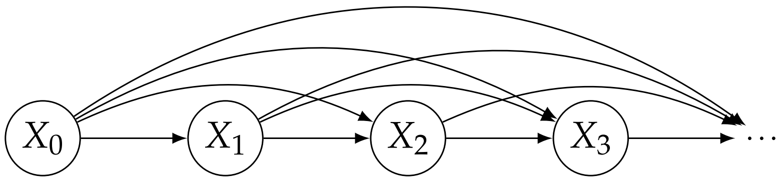


Markov models

Transition model

- In general, it specifies the probability distribution over the current state X_t given all the previous states X_0^{t-1}:

$$P(X_t|X_0^{t-1})$$



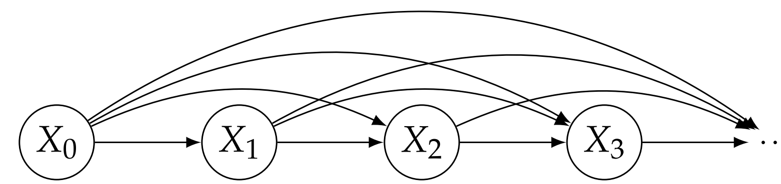
- Problem 1: X_0^{t-1} is unbounded in size as t increases.
- Solution: **Markov assumption** — the current state depends only on a *finite fixed number* of previous states. Such processes are called **Markov processes** or **Markov chains**.

Markov models

Transition model

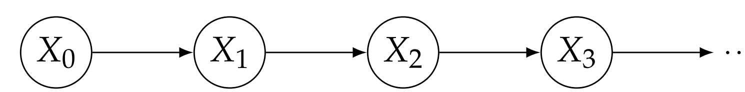
- In general, it specifies the probability distribution over the current state X_t given all the previous states X_0^{t-1}:

$$P(X_t|X_0^{t-1})$$



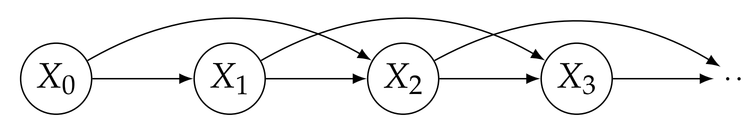
- Problem 1: X_0^{t-1} is unbounded in size as t increases.
- Solution: **Markov assumption** — the current state depends only on a *finite fixed number* of previous states. Such processes are called **Markov processes** or **Markov chains**.
- **First-order Markov process:**

$$P(X_t|X_0^{t-1}) = P(X_t|X_{t-1})$$



- **Second-order Markov process:**

$$P(X_t|X_0^{t-1}) = P(X_t|X_{t-1}^{t-2})$$

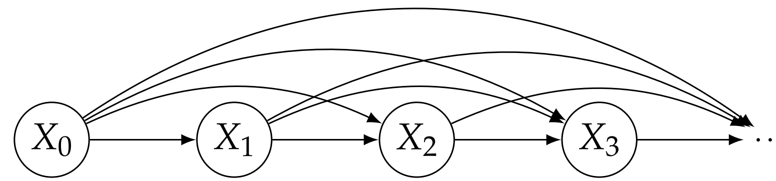


Markov models

Transition model

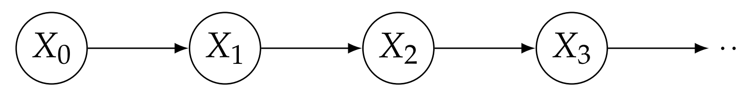
- In general, it specifies the probability distribution over the current state X_t given all the previous states X_0^{t-1}:

$$P(X_t|X_0^{t-1})$$



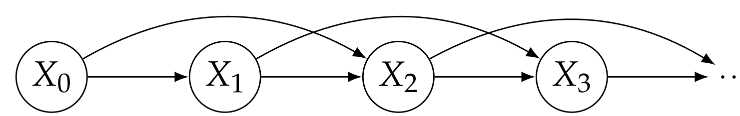
- Problem 1: X_0^{t-1} is unbounded in size as t increases.
- Solution: **Markov assumption** — the current state depends only on a *finite fixed number* of previous states. Such processes are called **Markov processes** or **Markov chains**.
- **First-order Markov process:**

$$P(X_t|X_0^{t-1}) = P(X_t|X_{t-1})$$



- **Second-order Markov process:**

$$P(X_t|X_0^{t-1}) = P(X_t|X_{t-1}^{t-2})$$



- Problem 2: Even with Markov assumption, there are infinitely many values of t. Do we have to specify a different distribution in each time step?
- Solution: assume a **stationary process**, i.e. the transition model does not change over time:

$$P(X_t|X_{t-k}^{t-1}) = P(X_{t'}|X_{t'-k}^{t'-1}) \qquad \text{for} \qquad t \neq t'.$$

Joint distribution of a Markov model

Assuming a stationary first-order Markov chain,

$$X_0 \longrightarrow X_1 \longrightarrow X_2 \longrightarrow X_3 \longrightarrow \cdots$$

the **MC joint distribution** is factorized as

$$P(X_0^T) = P(X_0) \prod_{t=1}^{T} P(X_t | X_{t-1}).$$

Joint distribution of a Markov model

Assuming a stationary first-order Markov chain,

$$X_0 \longrightarrow X_1 \longrightarrow X_2 \longrightarrow X_3 \longrightarrow \cdots$$

the **MC joint distribution** is factorized as

$$P(X_0^T) = P(X_0) \prod_{t=1}^{T} P(X_t | X_{t-1}).$$

This factorization is possible due to the following assumptions:

$$X_t \perp\!\!\!\perp X_0^{t-2} | X_{t-1}$$

- Past X are conditionally independent of future X given present X.
- In many cases, these assumptions are reasonable.
- They simplify things a lot: we can do reasoning in *polynomial time and space*!

Joint distribution of a Markov model

Assuming a stationary first-order Markov chain,

$$X_0 \to X_1 \to X_2 \to X_3 \to \cdots$$

the **MC joint distribution** is factorized as

$$P(X_0^T) = P(X_0)\prod_{t=1}^{T} P(X_t|X_{t-1}).$$

This factorization is possible due to the following assumptions:

$$X_t \perp\!\!\!\perp X_0^{t-2}|X_{t-1}$$

- Past X are conditionally independent of future X given present X.
- In many cases, these assumptions are reasonable.
- They simplify things a lot: we can do reasoning in *polynomial time and space*!

Just a *growing* Bayesian network with a very simple structure.

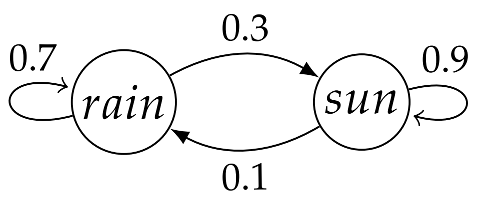
MC Example

- States: $X = \{rain, sun\} = \{r, s\}$
- Initial distribution: sun 100%
- Transition model: $P(X_t|X_{t-1})$

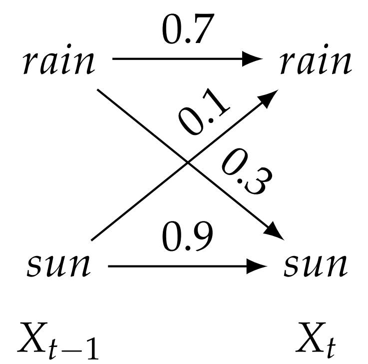
As a conditional prob. table:

X_{t-1}	X_t	$P(X_t\|X_{t-1})$
sun	sun	0.9
sun	$rain$	0.1
$rain$	sun	0.3
$rain$	$rain$	0.7

As a state transition diagram (automaton):



As a state trellis:



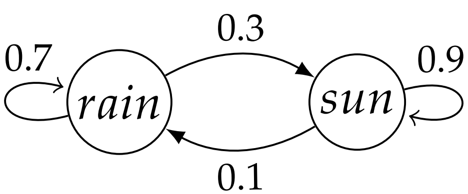
MC Example

- States: $X = \{rain, sun\} = \{r, s\}$
- Initial distribution: sun 100%
- Transition model: $P(X_t | X_{t-1})$

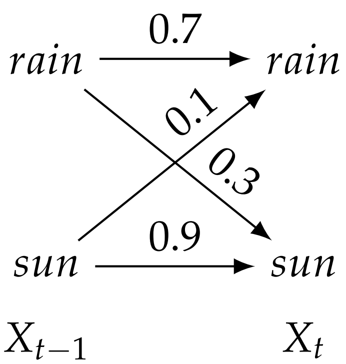
As a conditional prob. table:

X_{t-1}	X_t	$P(X_t \mid X_{t-1})$
sun	sun	0.9
sun	$rain$	0.1
$rain$	sun	0.3
$rain$	$rain$	0.7

As a state transition diagram (automaton):



As a state trellis:



What is the weather distribution after one step, i.e. $P(X_1)$ given $P(X_0 = s) = 1$?

$$
\begin{aligned}
P(X_1 = s) &= P(X_1 = s | X_0 = s) P(X_0 = s) + P(X_1 = s | X_0 = r) P(X_0 = r) = \\
&= \sum_{x_0} P(X_1 = s | x_0) P(x_0) = \\
&= 0.9 \cdot 1 + 0.3 \cdot 0 = 0.9
\end{aligned}
$$

Prediction

A **mini-forward** algorithm:

- What is $P(X_t)$ on some day t? $P(X_0)$ and $P(X_t|X_{t-1})$ is known.

$$P(X_t) = \sum_{x_{t-1}} P(X_t, x_{t-1}) = \\ = \sum_{x_{t-1}} \underbrace{P(X_t|x_{t-1})}_{\text{Step forward}} \underbrace{P(x_{t-1})}_{\text{Recursion}}$$

- $P(X_t|x_{t-1})$ is known from the transition model.
- $P(x_{t-1})$ is either known from $P(X_0)$ or from previous step of forward simulation.

Prediction

A **mini-forward** algorithm:

- What is $P(X_t)$ on some day t? $P(X_0)$ and $P(X_t|X_{t-1})$ is known.

$$P(X_t) = \sum_{x_{t-1}} P(X_t, x_{t-1}) = \\ = \sum_{x_{t-1}} \underbrace{P(X_t|x_{t-1})}_{\text{Step forward}} \underbrace{P(x_{t-1})}_{\text{Recursion}}$$

- $P(X_t|x_{t-1})$ is known from the transition model.
- $P(x_{t-1})$ is either known from $P(X_0)$ or from previous step of forward simulation.

Example run for our example starting from *sun*:

t	$P(X_t = s)$	$P(X_t = r)$
0	1	0
1	0.90	0.10
2	0.84	0.16
3	0.804	0.196
$\vdots$	$\vdots$	$\vdots$
∞	0.75	0.25

starting from *rain*:

t	$P(X_t = s)$	$P(X_t = r)$
0	0	1
1	0.3	0.7
2	0.48	0.52
3	0.588	0.412
$\vdots$	$\vdots$	$\vdots$
∞	0.75	0.25

Prediction

A **mini-forward** algorithm:

- What is $P(X_t)$ on some day t? $P(X_0)$ and $P(X_t|X_{t-1})$ is known.

$$P(X_t) = \sum_{x_{t-1}} P(X_t, x_{t-1}) = $$
$$= \sum_{x_{t-1}} \underbrace{P(X_t|x_{t-1})}_{\text{Step forward}} \underbrace{P(x_{t-1})}_{\text{Recursion}}$$

- $P(X_t|x_{t-1})$ is known from the transition model.
- $P(x_{t-1})$ is either known from $P(X_0)$ or from previous step of forward simulation.

Example run for our example starting from *sun*:

t	$P(X_t = s)$	$P(X_t = r)$
0	1	0
1	0.90	0.10
2	0.84	0.16
3	0.804	0.196
$\vdots$	$\vdots$	$\vdots$
∞	0.75	0.25

starting from *rain*:

t	$P(X_t = s)$	$P(X_t = r)$
0	0	1
1	0.3	0.7
2	0.48	0.52
3	0.588	0.412
$\vdots$	$\vdots$	$\vdots$
∞	0.75	0.25

In both cases we end up in the **stationary distribution** of the MC.

Stationary distribution

Informally, for most chains:

- Influence of initial distribution decreases with time.
- The limiting distribution is independent of the initial one.
- The limiting distribution $P_\infty(X)$ is called **stationary distribution** and it satisfies

$$P_\infty(X) = P_{\infty+1}(X) = \sum_x P(X|x)P_\infty(x)$$

Stationary distribution

Informally, for most chains:

- Influence of initial distribution decreases with time.
- The limiting distribution is independent of the initial one.
- The limiting distribution $P_\infty(X)$ is called **stationary distribution** and it satisfies

$$P_\infty(X) = P_{\infty+1}(X) = \sum_x P(X|x)P_\infty(x)$$

More formally:

- MC is called **regular** if there is a finite positive integer m such that after m time-steps, every state has a nonzero chance of being occupied, no matter what the initial state is.
- For a regular MC, a unique stationary distribution exists.

Stationary distribution

Informally, for most chains:

- Influence of initial distribution decreases with time.
- The limiting distribution is independent of the initial one.
- The limiting distribution $P_\infty(X)$ is called **stationary distribution** and it satisfies

$$P_\infty(X) = P_{\infty+1}(X) = \sum_x P(X|x)P_\infty(x)$$

More formally:

- MC is called **regular** if there is a finite positive integer m such that after m time-steps, every state has a nonzero chance of being occupied, no matter what the initial state is.
- For a regular MC, a unique stationary distribution exists.

Stationary distribution for the weather example:

$$\begin{aligned} P_\infty(s) &= P(s|s)P_\infty(s) + P(s|r)P_\infty(r) \\ P_\infty(r) &= P(r|s)P_\infty(s) + P(r|r)P_\infty(r) \end{aligned}$$

$$\begin{aligned} P_\infty(s) &= 0.9P_\infty(s) + 0.3P_\infty(r) \\ P_\infty(r) &= 0.1P_\infty(s) + 0.7P_\infty(r) \end{aligned}$$

$$\begin{aligned} P_\infty(s) &= 3P_\infty(r) \\ P_\infty(r) &= \frac{1}{3}P_\infty(s) \end{aligned}$$

Stationary distribution

Informally, for most chains:

- Influence of initial distribution decreases with time.
- The limiting distribution is independent of the initial one.
- The limiting distribution $P_\infty(X)$ is called **stationary distribution** and it satisfies

$$P_\infty(X) = P_{\infty+1}(X) = \sum_x P(X|x)P_\infty(x)$$

More formally:

- MC is called **regular** if there is a finite positive integer m such that after m time-steps, every state has a nonzero chance of being occupied, no matter what the initial state is.
- For a regular MC, a unique stationary distribution exists.

Stationary distribution for the weather example:

$$\begin{aligned} P_\infty(s) &= P(s|s)P_\infty(s) + P(s|r)P_\infty(r) \\ P_\infty(r) &= P(r|s)P_\infty(s) + P(r|r)P_\infty(r) \end{aligned}$$

$$\begin{aligned} P_\infty(s) &= 0.9P_\infty(s) + 0.3P_\infty(r) \\ P_\infty(r) &= 0.1P_\infty(s) + 0.7P_\infty(r) \end{aligned}$$

$$\begin{aligned} P_\infty(s) &= 3P_\infty(r) \\ P_\infty(r) &= \frac{1}{3}P_\infty(s) \end{aligned}$$

Two equations saying the same thing. But we know that $P_\infty(s) + P_\infty(r) = 1$, thus $P_\infty(s) = 0.75$ and $P_\infty(r) = 0.25$

Google PageRank

- The most famous and successful application of stationary distribution.
- Problem: How to order web pages mentioning the query phrases? How to compute relevance/importance of the result?
- Idea: Good pages are referenced more often; a random surfer spends more time on highly reachable pages.
- Each web page is a state.
- Random surfer clicks on a randomly chosen link on a web page, but with a small probability goes to a random page.
- This defines a MC. Its stationary distribution gives the importance of individual pages.
- In 1997, this was revolutionary and Google quickly surpassed the other search engines (Altavista, Yahoo, . . .).
- Nowadays, all search engines use link analysis along with many other factors (rank getting less important over time).

Hidden Markov Models

From Markov Chains to Hidden Markov Models

- MCs are not that useful in practice. They assume all the state variables are observable.
- In real world, some variables are observable, some are not (they are hidden).
- At any time slice t, the world is described by (X_t, E_t) where
  - X_t are the hidden state variables, and
  - E_t are the observable variables (evidence, effects).
- In general, the probability distribution over possible current states and observations given the past states and observations is

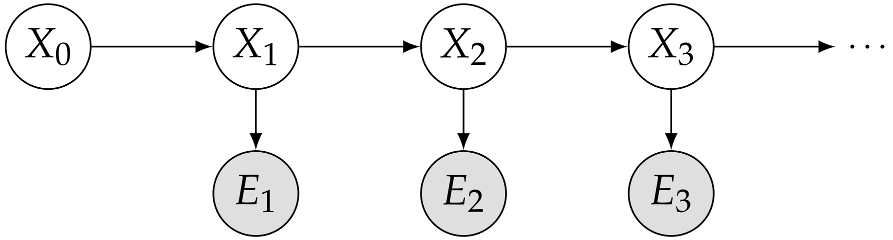
$$P(X_t, E_t | X_0^{t-1}, E_1^{t-1})$$

- Assumption: past observations E_1^{t-1} have no effect on the current state X_t and obs. E_t given the past states X_1^{t-1}. Using the first-order Markov assumption, then

$$P(X_t, E_t | X_0^{t-1}, E_1^{t-1}) = P(X_t, E_t | X_{t-1})$$

- Assumption: E_t is independent of X_{t-1} given X_t, then

$$P(X_t, E_t | X_{t-1}) = P(X_t | X_{t-1}) P(E_t | X_t)$$



Hidden Markov Model

HMM is defined by

- the initial state distribution $P(X_0)$,
- the transition model $P(X_t|X_{t-1})$, and
- the emission (sensor) model $P(E_t|X_t)$.
- It defines the following factorization of the joint distribution

$$P(X_0^T, E_1^T) = \underbrace{P(X_0)}_{\text{Init. state}} \prod_{t=1}^{T} \underbrace{P(X_t|X_{t-1})}_{\text{Transition model}} \underbrace{P(E_t|X_t)}_{\text{Sensor model}}$$

Hidden Markov Model

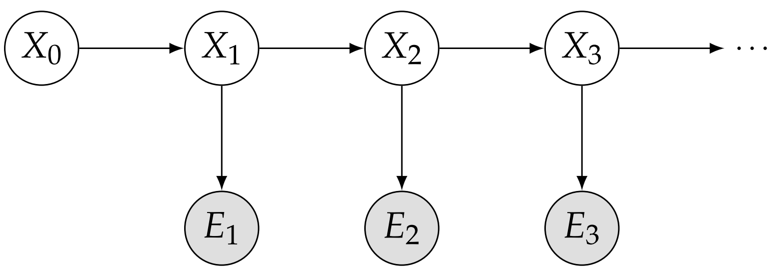
HMM is defined by

- the initial state distribution $P(X_0)$,
- the transition model $P(X_t|X_{t-1})$, and
- the emission (sensor) model $P(E_t|X_t)$.
- It defines the following factorization of the joint distribution

$$P(X_0^T, E_1^T) = \underbrace{P(X_0)}_{\text{Init. state}} \prod_{t=1}^{T} \underbrace{P(X_t|X_{t-1})}_{\text{Transition model}} \underbrace{P(E_t|X_t)}_{\text{Sensor model}}$$

Independence assumptions:

$$\begin{aligned} X_2 &\perp\!\!\!\perp X_0, E_1 | X_1 \\ E_2 &\perp\!\!\!\perp X_0, X_1, E_1 | X_2 \\ X_3 &\perp\!\!\!\perp X_0, X_1, E_1, E_2 | X_2 \\ E_3 &\perp\!\!\!\perp X_0, X_1, E_1, X_2, E_2 | X_3 \\ &\ldots \end{aligned}$$



HMM Examples

- Speech recognition: E – acoustic signals, X – phonemes
- Machine translation: E – words in source lang., X – translation options
- Handwriting recognition: E – pen movements, X – (parts of) characters
- EKG and EEG analysis: E – signals, X – signal characteristics
- DNA sequence analysis:
  - E – responses from molecular markers, $X = \{A, C, G, T\}$
  - $E = \{A, C, G, T\}$, X – subsequences with interesting interpretations
- Robot tracking: E – sensor measurements, X – positions on a map
- Recognition in images with special arrangement, e.g. car registration labels: E – images of columns of the registration label, X – characters forming the label

Weather-Umbrella Domain

Suppose you are in a situation with no chance of learning what the weather is today.

- You may be a hard working Ph.D. student locked in your no-windows lab for several days.
- Or you may be a soldier guarding a military base hidden a few hundred meters underneath the Earth surface.

The only indication of the weather outside is your boss (or supervisor) coming to his office each day, and bringing an umbrella or not.

Weather-Umbrella Domain

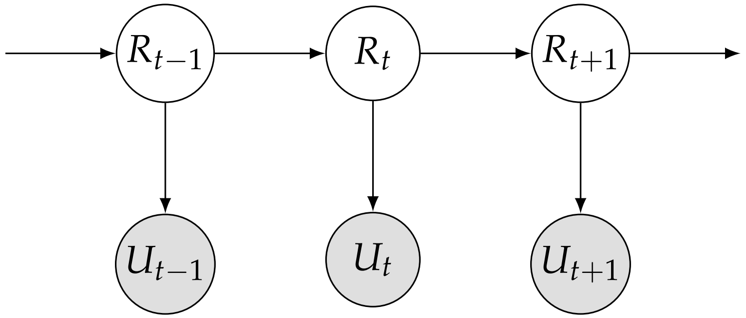
Suppose you are in a situation with no chance of learning what the weather is today.

- You may be a hard working Ph.D. student locked in your no-windows lab for several days.
- Or you may be a soldier guarding a military base hidden a few hundred meters underneath the Earth surface.

The only indication of the weather outside is your boss (or supervisor) coming to his office each day, and bringing an umbrella or not.

Random variables:

- R_t: Is it raining on day t?
- U_t: Did your boss bring an umbrella?



Transition model:

R_{t-1}	R_t	$P(R_t \mid R_{t-1})$
t	t	0.7
t	f	0.3
f	t	0.3
f	f	0.7

Emission model:

R_t	U_t	$P(U_t \mid R_t)$
t	t	0.9
t	f	0.1
f	t	0.2
f	f	0.8

HMM tasks

Filtering:

- computing the posterior distribution over *the current state* given all the previous evidence, i.e.
- $P(X_t|e_1^t)$.
- AKA state estimation, or tracking.
- *Forward* algorithm.

HMM tasks

Filtering:

- computing the posterior distribution over *the current state* given all the previous evidence, i.e.
- $P(X_t|e_1^t)$.
- AKA state estimation, or tracking.
- *Forward* algorithm.

Prediction:

- computing the posterior distribution over the future state given all the previous evidence, i.e.
- $P(X_{t+k}|e_1^t)$ for some $k > 0$.
- The same *"mini-forward"* algorithm as in case of Markov Chain.

HMM tasks

Filtering:

- computing the posterior distribution over *the current state* given all the previous evidence, i.e.
- $P(X_t|e_1^t)$.
- AKA state estimation, or tracking.
- *Forward* algorithm.

Prediction:

- computing the posterior distribution over the future state given all the previous evidence, i.e.
- $P(X_{t+k}|e_1^t)$ for some $k > 0$.
- The same *"mini-forward"* algorithm as in case of Markov Chain.

Smoothing:

- computing the posterior distribution over the past state given all the evidence, i.e.
- $P(X_k|e_1^t)$ for some $k \in (0, t)$
- It estimates the state better than filtering because more evidence is available.
- *Forward-backward* algorithm.

HMM tasks (cont.)

Recognition or **evaluation of statistical model:**

- Compute the likelihood of an HMM, i.e. the probability of observing the data given the HMM parameters,
- $P(e_1^t|\theta)$.
- If several HMMs are given, the most likely model can be chosen (as a class label).
- Uses *forward* algorithm.

HMM tasks (cont.)

Recognition or **evaluation of statistical model:**

- Compute the likelihood of an HMM, i.e. the probability of observing the data given the HMM parameters,
- $P(e_1^t|\theta)$.
- If several HMMs are given, the most likely model can be chosen (as a class label).
- Uses *forward* algorithm.

Most likely explanation:

- given a sequence of observations, find the sequence of states that has most likely generated those observations, i.e.
- $\arg\max_{x_1^t} P(x_1^t|e_1^t)$.
- *Viterbi* algorithm (dynamic programming).
- Useful in speech recognition, in reconstruction of bit strings transmitted over a noisy channel, etc.

HMM tasks (cont.)

Recognition or **evaluation of statistical model:**

- Compute the likelihood of an HMM, i.e. the probability of observing the data given the HMM parameters,
- $P(e_1^t|\theta)$.
- If several HMMs are given, the most likely model can be chosen (as a class label).
- Uses *forward* algorithm.

Most likely explanation:

- given a sequence of observations, find the sequence of states that has most likely generated those observations, i.e.
- $\arg\max_{x_1^t} P(x_1^t|e_1^t)$.
- *Viterbi* algorithm (dynamic programming).
- Useful in speech recognition, in reconstruction of bit strings transmitted over a noisy channel, etc.

HMM Learning:

- Given the HMM structure, learn the transition and sensor models from observations.
- *Baum-Welch* algorithm, an instance of EM algorithm.
- Requires smoothing, learning with filtering can fail to converge correctly.

Filtering

Recursive estimation:

- Any useful filtering algorithm must *maintain and update* a current state estimate (as opposed to estimating the current state from the whole evidence sequence each time), i.e.
- we want to find a function u such that

$$P(X_t|e_1^t) = u(P(X_{t-1}|e_1^{t-1}), e_t)$$

Filtering

Recursive estimation:

- Any useful filtering algorithm must *maintain and update* a current state estimate (as opposed to estimating the current state from the whole evidence sequence each time), i.e.
- we want to find a function u such that

$$P(X_t|e_1^t) = u(P(X_{t-1}|e_1^{t-1}), e_t)$$

This process will have 2 parts:

1. Predict the current state at t from the filtered estimate of state at $t-1$.
2. Update the prediction with new evidence at t.

Filtering

Recursive estimation:

- Any useful filtering algorithm must *maintain and update* a current state estimate (as opposed to estimating the current state from the whole evidence sequence each time), i.e.
- we want to find a function u such that

$$P(X_t|e_1^t) = u(P(X_{t-1}|e_1^{t-1}), e_t)$$

This process will have 2 parts:

1. Predict the current state at t from the filtered estimate of state at $t-1$.
2. Update the prediction with new evidence at t.

$$\begin{aligned} P(X_t|e_1^t) &= P(X_t|e_1^{t-1}, e_t) = && \text{(split the evidence sequence)} \\ &= \alpha P(e_t|X_t, e_1^{t-1}) P(X_t|e_1^{t-1}) = && \text{(from Bayes rule)} \\ &= \alpha P(e_t|X_t) P(X_t|e_1^{t-1}) && \text{(using Markov assumption)} \end{aligned}$$

where

- α is a normalization constant,
- $P(e_t|X_t)$ is the update by evidence (known from sensor model), and
- $P(X_t|e_1^{t-1})$ is the 1-step prediction. How to compute it?

Filtering (cont.)

1-step prediction:

$$
\begin{aligned}
P(X_t|e_1^{t-1}) &= \sum_{x_{t-1}} P(X_t, x_{t-1}|e_1^{t-1}) = && \text{(as a sum over previous states)} \\
&= \sum_{x_{t-1}} P(X_t|x_{t-1}, e_1^{t-1}) P(x_{t-1}|e_1^{t-1}) = && \text{(introduce conditioning on previous state)} \\
&= \sum_{x_{t-1}} P(X_t|x_{t-1}) P(x_{t-1}|e_1^{t-1}), && \text{(using Markov assumption)}
\end{aligned}
$$

where

- $P(X_t|x_{t-1})$ is known from transition model, and
- $P(x_{t-1}|e_1^{t-1})$ is the filtered estimate at previous step.

Filtering (cont.)

1-step prediction:

$$\begin{aligned} P(X_t|e_1^{t-1}) &= \sum_{x_{t-1}} P(X_t, x_{t-1}|e_1^{t-1}) = && \text{(as a sum over previous states)} \\ &= \sum_{x_{t-1}} P(X_t|x_{t-1}, e_1^{t-1}) P(x_{t-1}|e_1^{t-1}) = && \text{(introduce conditioning on previous state)} \\ &= \sum_{x_{t-1}} P(X_t|x_{t-1}) P(x_{t-1}|e_1^{t-1}), && \text{(using Markov assumption)} \end{aligned}$$

where

- $P(X_t|x_{t-1})$ is known from transition model, and
- $P(x_{t-1}|e_1^{t-1})$ is the filtered estimate at previous step.

All together:

$$\underbrace{P(X_t|e_1^t)}_{\text{new estimate}} = \alpha \underbrace{P(e_t|X_t)}_{\text{sensor model}} \sum_{x_{t-1}} \underbrace{P(X_t|x_{t-1})}_{\text{transition model}} \underbrace{P(x_{t-1}|e_1^{t-1})}_{\text{previous estimate}}$$

Online belief updates

$$\underbrace{P(X_t|e_1^t)}_{\text{new estimate}} = \alpha \underbrace{P(e_t|X_t)}_{\text{sensor model}} \sum_{x_{t-1}} \underbrace{P(X_t|x_{t-1})}_{\text{transition model}} \underbrace{P(x_{t-1}|e_1^{t-1})}_{\text{previous estimate}}$$

- At every moment, we have a belief distribution over the states, $B(X)$.
- Initially, it is our prior distribution $B(X) = P(X_0)$.
- The above update equation may be split into 2 parts:
  1. **Update for time step**:
     $$B(X) \leftarrow \sum_{x'} P(X|x') \cdot B(X)$$
  2. **Update for a new evidence**:
     $$B(X) \leftarrow \alpha P(e|X) \cdot B(X),$$
     where α is a normalization constant.
- If you update for time step several times without evidence, it is a prediction several steps ahead.
- If you update for evidence several times without a time step, you incorporate multiple measurements.
- The *forward* algorithm does both updates at once and *does not normalize*!

Forward algorithm

$$\underbrace{P(X_t|e_1^t)}_{\text{new estimate}} = \alpha \underbrace{P(e_t|X_t)}_{\text{sensor model}} \sum_{x_{t-1}} \underbrace{P(X_t|x_{t-1})}_{\text{transition model}} \underbrace{P(x_{t-1}|e_1^{t-1})}_{\text{previous estimate}}$$

Forward message: a filtered estimate of state at time t given the evidence e_1^t, i.e.

$$f_t(X_t) \stackrel{\text{def}}{=} P(X_t|e_1^t).$$

Then

$$f_t(X_t) = \alpha P(e_t|X_t) \sum_{x_{t-1}} P(X_t|x_{t-1}) f_{t-1}(x_{t-1}),$$

i.e.

$$f_t = \alpha \cdot \text{FORWARD-UPDATE}(f_{t-1}, e_t)$$

where

- the FORWARD-UPDATE function implements the update equation above (without the normalization), and
- the recursion is initialized with $f_0(X_0) = P(X_0)$.

Umbrella example

Day 0:

- No observations, just prior belief: $P(R_0) = (0.5, 0.5)$.

Umbrella example

Day 0:

- No observations, just prior belief: $P(R_0) = (0.5, 0.5)$.

Day 1: 1st observation $U_1 = true$

- Prediction: $P(R_1) =$

Umbrella example

Day 0:

- No observations, just prior belief: $P(R_0) = (0.5, 0.5)$.

Day 1: 1st observation $U_1 = true$

- Prediction: $P(R_1) = \sum_{r_0} P(R_1|r_0)P(r_0) = (0.7, 0.3) \cdot 0.5 + (0.7, 0.3) \cdot 0.5 = (0.5, 0.5)$

Umbrella example

Day 0:

- No observations, just prior belief: $P(R_0) = (0.5, 0.5)$.

Day 1: 1st observation $U_1 = true$

- Prediction: $P(R_1) = \sum_{r_0} P(R_1|r_0)P(r_0) = (0.7, 0.3) \cdot 0.5 + (0.7, 0.3) \cdot 0.5 = (0.5, 0.5)$
- Update by evidence and normalize:
  $P(R_1|u_1) =$

Umbrella example

Day 0:

- No observations, just prior belief: $P(R_0) = (0.5, 0.5)$.

Day 1: 1st observation $U_1 = \mathit{true}$

- Prediction: $P(R_1) = \sum_{r_0} P(R_1|r_0)P(r_0) = (0.7, 0.3) \cdot 0.5 + (0.7, 0.3) \cdot 0.5 = (0.5, 0.5)$
- Update by evidence and normalize:
  $P(R_1|u_1) = \alpha P(u_1|R_1)P(R_1) = \alpha(0.9, 0.2) \cdot (0.5, 0.5) = \alpha(0.45, 0.1) = (0.818, 0.182)$

Umbrella example

Day 0:

- No observations, just prior belief: $P(R_0) = (0.5, 0.5)$.

Day 1: 1st observation $U_1 = true$

- Prediction: $P(R_1) = \sum_{r_0} P(R_1|r_0)P(r_0) = (0.7, 0.3) \cdot 0.5 + (0.7, 0.3) \cdot 0.5 = (0.5, 0.5)$
- Update by evidence and normalize:
  $P(R_1|u_1) = \alpha P(u_1|R_1)P(R_1) = \alpha(0.9, 0.2) \cdot (0.5, 0.5) = \alpha(0.45, 0.1) = (0.818, 0.182)$

Day 2: 2nd observation $U_2 = true$

- Prediction: $P(R_2|u_1) = \sum_{r_1} P(R_2|r_1)P(r_1|u_1) = (0.7, 0.3) \cdot 0.818 + (0.3, 0.7) \cdot 0.182 = (0.627, 0.373)$

Umbrella example

Day 0:

- No observations, just prior belief: $P(R_0) = (0.5, 0.5)$.

Day 1: 1st observation $U_1 = true$

- Prediction: $P(R_1) = \sum_{r_0} P(R_1|r_0)P(r_0) = (0.7, 0.3) \cdot 0.5 + (0.7, 0.3) \cdot 0.5 = (0.5, 0.5)$
- Update by evidence and normalize:
  $P(R_1|u_1) = \alpha P(u_1|R_1)P(R_1) = \alpha(0.9, 0.2) \cdot (0.5, 0.5) = \alpha(0.45, 0.1) = (0.818, 0.182)$

Day 2: 2nd observation $U_2 = true$

- Prediction: $P(R_2|u_1) = \sum_{r_1} P(R_2|r_1)P(r_1|u_1) = (0.7, 0.3) \cdot 0.818 + (0.3, 0.7) \cdot 0.182 = (0.627, 0.373)$
- Update by evidence and normalize:
  $P(R_2|u_1, u_2) = \alpha P(u_2|R_2)P(R_2|u_1) = \alpha(0.9, 0.2) \cdot (0.627, 0.373) = (0.883, 0.117)$

Umbrella example

Day 0:

- No observations, just prior belief: $P(R_0) = (0.5, 0.5)$.

Day 1: 1st observation $U_1 = true$

- Prediction: $P(R_1) = \sum_{r_0} P(R_1|r_0)P(r_0) = (0.7, 0.3) \cdot 0.5 + (0.7, 0.3) \cdot 0.5 = (0.5, 0.5)$
- Update by evidence and normalize:
  $P(R_1|u_1) = \alpha P(u_1|R_1)P(R_1) = \alpha(0.9, 0.2) \cdot (0.5, 0.5) = \alpha(0.45, 0.1) = (0.818, 0.182)$

Day 2: 2nd observation $U_2 = true$

- Prediction: $P(R_2|u_1) = \sum_{r_1} P(R_2|r_1)P(r_1|u_1) = (0.7, 0.3) \cdot 0.818 + (0.3, 0.7) \cdot 0.182 = (0.627, 0.373)$
- Update by evidence and normalize:
  $P(R_2|u_1, u_2) = \alpha P(u_2|R_2)P(R_2|u_1) = \alpha(0.9, 0.2) \cdot (0.627, 0.373) = (0.883, 0.117)$

Probability of rain increased, because rain tends to persist.

Prediction

- Filtering contains 1-step prediction.
- General prediction in HMM is like filtering without adding a new evidence:

$$P(X_{t+k+1}|e_1^t) = \sum_{x_{t+k}} P(X_{t+k+1}|x_{t+k})P(x_{t+k}|e_1^t)$$

- It involves the transition model only.
- From the time slice we have our last evidence, it is just a Markov chain over hidden states:
  - Use filtering to compute $P(X_t|e_1^t)$. This is the initial state of MC.
  - Use mini-forward algorithm to predict further in time.
- By predicting further in the future, we recover the stationary distribution of the Markov chain given by the transition model.

Model evaluation

- Compute the likelihood of the evidence sequence given the HMM parameters, i.e. $P(e_1^t)$.
- Useful for assesssing which of several HMMs could have generated the observation.

Model evaluation

- Compute the likelihood of the evidence sequence given the HMM parameters, i.e. $P(e_1^t)$.
- Useful for assesssing which of several HMMs could have generated the observation.

Likelihood message:

- Similarly to forward message, we can define a likelihood message as

$$l_t(X_t) \stackrel{\text{def}}{=} P(X_t, e_1^t)$$

- It can be shown that the forward algorithm can be used to update the likelihood message as well:

$$l_t(X_t) = \text{FORWARD-UPDATE}(l_{t-1}(X_{t-1}), e_t)$$

- The **likelihood** of e_1^t is then obtained by summing out X_t:

$$L_t = P(e_1^t) = \sum_{x_t} l_t(x_t)$$

- l_t is a probability of longer and longer evidence sequence as time goes by, resulting in numbers close to 0 $\Rightarrow$ underflow problems. (
- When forward updates are used with the forward message f_t, these issues are prevented, because f_t is rescaled in each time step to form a proper prob. distribution.

Smoothing

Compute the distribution over past state given evidence up to present:

$$P(X_k|e_1^t) \text{ for some } k < t.$$

Smoothing

Compute the distribution over past state given evidence up to present:

$$P(X_k|e_1^t) \text{ for some } k < t.$$

Let's factorize the distribution as follows:

$$\begin{aligned} P(X_k|e_1^t) &= P(X_t|e_1^k, e_{k+1}^t) = && \text{(split the evidence sequence)} \\ &= \alpha P(e_{k+1}^t|X_k, e_1^k) P(X_k|e_1^k) = && \text{(from Bayes rule)} \\ &= \alpha \underbrace{P(e_{k+1}^t|X_k)}_{?} \underbrace{P(X_k|e_1^k)}_{\text{filtering, forward}} && \text{(using Markov assumption)} \end{aligned}$$

Smoothing

Compute the distribution over past state given evidence up to present:

$$P(X_k|e_1^t) \text{ for some } k < t.$$

Let's factorize the distribution as follows:

$$\begin{aligned}
P(X_k|e_1^t) &= P(X_t|e_1^k, e_{k+1}^t) = && \text{(split the evidence sequence)} \\
&= \alpha P(e_{k+1}^t|X_k, e_1^k) P(X_k|e_1^k) = && \text{(from Bayes rule)} \\
&= \alpha \underbrace{P(e_{k+1}^t|X_k)}_{?} \underbrace{P(X_k|e_1^k)}_{\text{filtering, forward}} && \text{(using Markov assumption)}
\end{aligned}$$

$$\begin{aligned}
P(e_{k+1}^t|X_k) &= \sum_{x_{k+1}} P(e_{k+1}^t|X_k, x_{k+1}) P(x_{k+1}|X_k) = && \text{(condition on } X_{k+1}\text{)} \\
&= \sum_{x_{k+1}} P(e_{k+1}^t|x_{k+1}) P(x_{k+1}|X_k) = && \text{(using Markov assumption)} \\
&= \sum_{x_{k+1}} P(e_{k+1}, e_{k+2}^t|x_{k+1}) P(x_{k+1}|X_k) = && \text{(split evidence sequence)} \\
&= \sum_{x_{k+1}} \underbrace{P(e_{k+1}|x_{k+1})}_{\text{sensor model}} \underbrace{P(e_{k+2}^t|x_{k+1})}_{\text{recursion}} \underbrace{P(x_{k+1}|X_k)}_{\text{transition model}} && \text{(using cond. independence)}
\end{aligned}$$

Smoothing (cont.)

$$P(e^t_{k+1}|X_k) = \sum_{x_{k+1}} \underbrace{P(e_{k+1}|x_{k+1})}_{\text{sensor model}} \underbrace{P(e^t_{k+2}|x_{k+1})}_{\text{recursion}} \underbrace{P(x_{k+1}|X_k)}_{\text{transition model}}$$

Backward message:

$$b_k(X_k) \stackrel{\text{def}}{=} P(e^t_{k+1}|X_k)$$

Then

$$b_k(X_k) = \sum_{x_{k+1}} P(e_{k+1}|x_{k+1}) b_{k+1} P(x_{k+1}|X_k)$$

i.e.

$$b_k = \text{BACKWARD-UPDATE}(b_{k+1}, e_{k+1})$$

where

- the BACKWARD-UPDATE function implements the update equation above, and
- the recursion is initialized by $b_t = P(e^t_{t+1}|X_t) = P(\emptyset|X_t) = \mathbf{1}$.

Smoothing (cont.)

The whole smoothing algorithm can then be expressed as

$$P(X_k|e_1^t) = \alpha P(e_{k+1}^t|X_k)P(X_k|e_1^k) = \alpha f_k \times b_k,$$

where

- $\times$ denotes element-wise multiplication.
- Both f_k and b_k can be computed by recursion in time:
  - f_k by a forward recursion from 1 to k,
  - b_k by a backward recursion from t to $k+1$.

Smoothing (cont.)

The whole smoothing algorithm can then be expressed as

$$P(X_k|e_1^t) = \alpha P(e_{k+1}^t|X_k)P(X_k|e_1^k) = \alpha f_k \times b_k,$$

where

- $\times$ denotes element-wise multiplication.
- Both f_k and b_k can be computed by recursion in time:
  - f_k by a forward recursion from 1 to k,
  - b_k by a backward recursion from t to $k+1$.

Smoothing the whole sequence of hidden states:

- Can be computed efficiently by
- a forward pass, computing and *storing* all the filtered estimates f_k for $k = 1 \rightarrow t$, followed by
- a backward pass, using the stored f_ks and computing b_ks on the fly for $k = t \rightarrow 1$.

Umbrella example: Smoothing

Filtering with uniform prior and observations $U_1 = true$ and $U_2 = true$:

- Day 0: No observations, just prior belief: $P(R_0) = (0.5, 0.5)$.
- Day 1: Observation $U_1 = true$: $P(R_1|u_1) = (0.818, 0.182)$
- Day 2: Observation $U_2 = true$: $P(R_2|u_1, u_2) = (0.883, 0.117)$

Umbrella example: Smoothing

Filtering with uniform prior and observations $U_1 = true$ and $U_2 = true$:

- Day 0: No observations, just prior belief: $P(R_0) = (0.5, 0.5)$.
- Day 1: Observation $U_1 = true$: $P(R_1|u_1) = (0.818, 0.182)$
- Day 2: Observation $U_2 = true$: $P(R_2|u_1, u_2) = (0.883, 0.117)$

Filtering versus smoothing:

- Filtering estimates $P(R_t)$ by using evidence up to time t, i.e. $P(R_1)$ is estimated by $P(R_1|u_1)$, i.e. it ignores future observation u_2.
- At $t = 2$, we have a new observation u_2 which also brings some information about R_1. We can thus update the distribution about past state by future evidence by computing $P(R_1|u_1, u_2)$.

Umbrella example: Smoothing

Filtering with uniform prior and observations $U_1 = true$ and $U_2 = true$:

- Day 0: No observations, just prior belief: $P(R_0) = (0.5, 0.5)$.
- Day 1: Observation $U_1 = true$: $P(R_1|u_1) = (0.818, 0.182)$
- Day 2: Observation $U_2 = true$: $P(R_2|u_1, u_2) = (0.883, 0.117)$

Filtering versus smoothing:

- Filtering estimates $P(R_t)$ by using evidence up to time t, i.e. $P(R_1)$ is estimated by $P(R_1|u_1)$, i.e. it ignores future observation u_2.
- At $t = 2$, we have a new observation u_2 which also brings some information about R_1. We can thus update the distribution about past state by future evidence by computing $P(R_1|u_1, u_2)$.

Smoothing:

$$P(R_1|u_1, u_2) = \alpha P(R_1|u_1)P(u_2|R_1)$$

- The first term is known from the forward pass.
- The second term can be computed by the backward recursion:

$$P(u_2|R_1) = \sum_{r_2} P(u_2|r_2)P(\emptyset|r_2)P(r_2|R_1) = 0.9 \cdot 1 \cdot (0.7, 0.3) + 0.2 \cdot 1 \cdot (0.3, 0.7) = (0.69, 0.41).$$

- Substituting back to the smoothing equation above:

$$P(R_1|u_1, u_2) = \alpha(0.818, 0.182) \times (0.69, 0.41) \doteq (0.883, 0.117).$$

Forward-backward algorithm

Algorithm 1: FORWARD-BACKWARD(e_1^t, P_0) returns a vector of prob. distributions

Input : e_1^t – a vector of evidence values for steps $1, \dots, t$
P_0 – the prior distribution on the initial state

Local : f_0^t – a vector of forward messages for steps $0, \dots, t$
b – the backward message, initially all 1s
s_1^t – a vector of smoothed estimates for steps $1, \dots, t$

Output: a vector of prob. distributions, i.e. the smoothed estimates s_1^t

begin
  $f_0 \leftarrow P_0$
  for $i = 1$ *to* t **do**
    $f_i \leftarrow$ `FORWARD-UPDATE(`f_{i-1}, e_i`)`
  for $i = t$ *downto* 1 **do**
    $s_i \leftarrow$ `NORMALIZE(`$f_i \times b$`)`
    $b \leftarrow$ `BACKWARD-UPDATE(`b, e_i`)`
  return s_1^t

Most likely sequence

Weather-Umbrella example problem:

- Assume that the observation sequence over 5 days is $u_1^5 = (true, true, false, true, true)$.
- What is the weather sequence most likely to explain these observations?

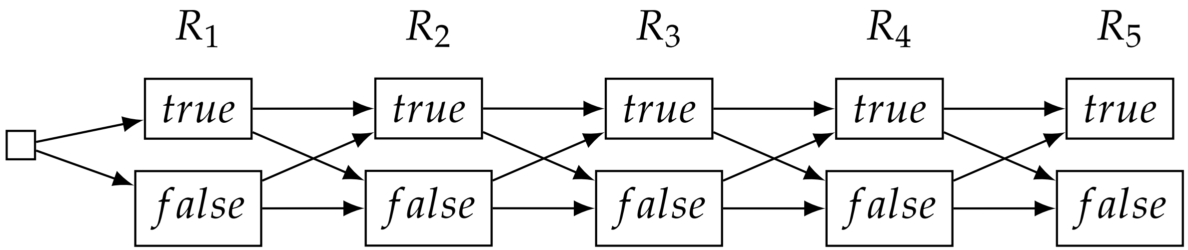
Most likely sequence

Weather-Umbrella example problem:

- Assume that the observation sequence over 5 days is $u_1^5 = (true, true, false, true, true)$.
- What is the weather sequence most likely to explain these observations?

Possible approaches:

- Approach 1: Enumeration of all possible sequences.
  - View each sequence as a possible path through the state trellis graph:



  - There are 2 possible states for each of the 5 days, that is $2^5 = 32$ different state sequences r_1^5.
  - *Enumerate* and evaluate them by computing $P(r_1^t, e_1^t)$, and choose the one with the largest probability.
  - Intractable for longer sequences/larger state spaces. Can it be done more efficiently?

Most likely sequence (cont.)

- Approach 2: Sequence of most likely states?
  - Use smoothing to find a posterior distribution of rain $P(R_k|u_1^t)$ for all time steps.
  - Then construct a sequence of most likely states

$$(\arg\max_{r_1} P(r_1|u_1^t), \ldots, \arg\max_{r_t} P(r_t|u_1^t)).$$

  - But this is *not the same* as the most likely sequence

$$\arg\max_{r_1^t} P(r_1^t|u_1^t)$$

- Approach 3: Find $\arg\max_{r_1^t} P(r_1^t|u_1^t)$ using a recursive algorithm:
  - The likelihood of any path is the product of the transition probabilities along the path and the probabilities of the given observations at each state.
  - The most likely path to certain state x_t consists of the most likely path to some state x_{t-1} followed by a transition to x_t. The state x_{t-1} that will become part of the path to x_t is the one which maximizes the likelihood of that path.
  - Let's define a recursive relationship between most likely path to each state x_{t-1} and most likely path to each state x_t.

Viterbi algorithm

A dynamic programming approach to finding most likely sequence of states.

- We want to find $\arg\max_{x_1^t} P(x_1^t | e_1^t)$.
- Note that $\arg\max_{x_1^t} P(x_1^t | e_1^t) = \arg\max_{x_1^t} P(x_1^t, e_1^t)$. Let's work with the joint.
- Let's define the max message:

$$\begin{aligned} m_t(X_t) &\overset{\text{def}}{=} \max_{x_1^{t-1}} P(x_1^{t-1}, X_t, e_1^t) = \\ &= \max_{x_1^{t-2}, x_{t-1}} P(e_t | X_t) P(X_t | x_{t-1}) P(x_1^{t-1}, e_1^{t-1}) = \\ &= P(e_t | X_t) \max_{x_{t-1}} P(X_t | x_{t-1}) \max_{x_1^{t-2}} P(x_1^{t-1}, e_1^{t-1}) = \\ &= P(e_t | X_t) \max_{x_{t-1}} P(X_t | x_{t-1}) m_{t-1}(x_{t-1}) \text{ for } t \geq 2. \end{aligned}$$

- The recursion is initialized by $m_1 = P(X_1, e_1) = \text{FORWARD-UPDATE}(P(X_0), e_1)$.
- At the end, we have the probability of the most likely sequence reaching *each final state*.
- The construction of the most likely sequence starts in the final state with the largest probability, and runs backwards.
- The algorithm needs to store for each x_t its "best" predecesor x_{t-1}.

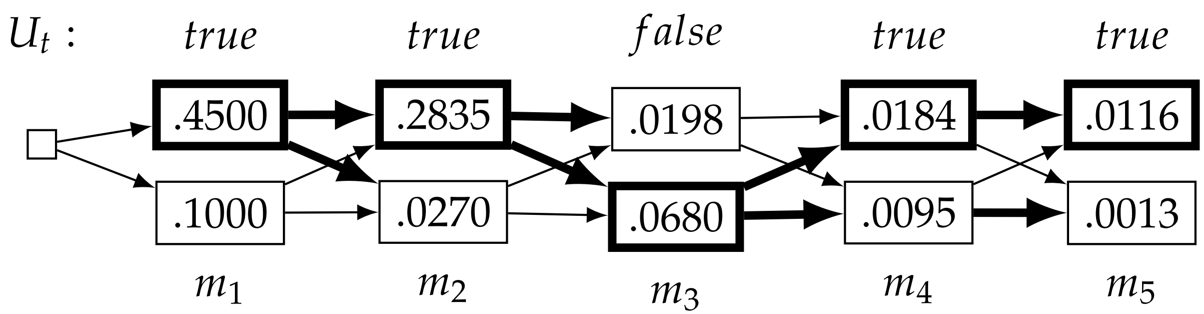
Viterbi algorithm: example

Weather-Umbrella example:

- After applying

$$m_1 = P(X_1, e_1) = \text{FORWARD-UPDATE}(P(X_0), e_1) \text{and}$$
$$m_t = P(e_t|X_t) \max_{x_{t-1}} P(X_t|x_{t-1}) m_{t-1} \text{ for } t \geq 2,$$

  we have the following:



- The most likely sequence is constructed by
  - starting in the last state with the highest probability, and
  - following the bold arrows backwards.

Note:

- The probabilities for sequences of increasing length decrease towards 0, they can underflow.
- To remedy this, we can use the log-sum-exp approach.

Summary

Competencies

After this lecture, a student shall be able to . . .

- define Markov Chain (MC), describe assumptions used in MCs;
- show the factorization of joint probability distribution used by 1st-order MC;
- understand and implement the mini-forward algorithm for prediction;
- explain the notion of the stationary distribution of a MC, describe its features, compute it analytically for simple cases;
- define Hidden Markov Model (HMM), describe assumptions used in HMM;
- explain the factorization of the joint probability distribution of states and observations implied by HMM;
- define the main inference tasks related to HMMs;
- explain the principles of forward, forward-backward, and Viterbi algorithms, implement them, and know when to apply them;
- compute a few steps of the above algorithms by hand for simple cases;
- describe issues that can arise in practice when using the above algorithms.