University of Rome "La Sapienza"

Master in Artificial Intelligence and Robotics

Machine Learning

A.Y. 2018/2019

Prof. Luca Iocchi

Sapienza University of Rome, Italy
Master in Artificial Intelligence and Robotics
Machine Learning (2018/19)

11. Hidden Markov Models and Partially Observable MDPs

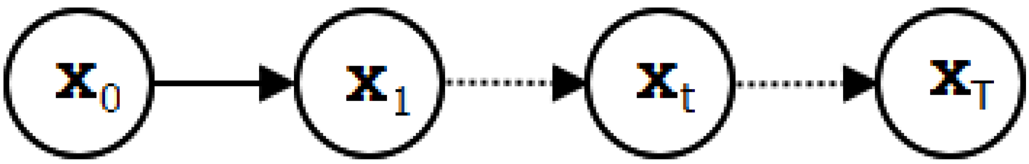
Luca Iocchi

Overview

- Hidden Markov Models (HMM)
- Learning in HMM
- Partially Observable Markov Decision Processes (POMDP)
- Policy trees
- Example: POMDP tiger proglem

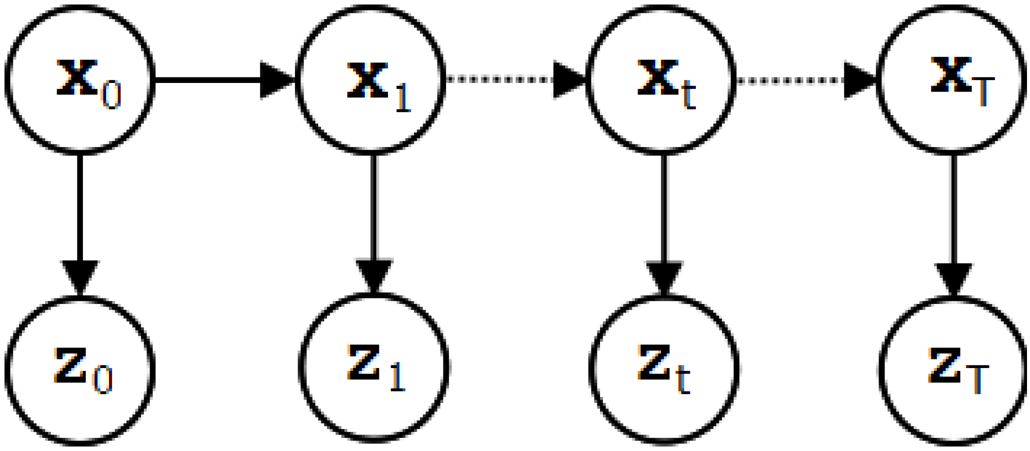
Markov Chain

Dynamic system evolving according to the Markov property.



Future evolution depends only on the current state $\mathbf{X}_t$

Hidden Markov Models (HMM)



- states $\mathbf{x}_t$ are **discrete** and **non-observable**,
- observations (emissions) $\mathbf{z}_t$ can be either discrete or continuous.
- controls $\mathbf{u}_t$ are not present (i.e., evolution is not controlled by our system),

Sapienza University of Rome, Italy - Machine Learning (2018/2019)

HMM representation

HMM $= \langle \mathbf{X}, \mathbf{Z}, \pi_0 \rangle$

- transition model: $P(\mathbf{x}_t | \mathbf{x}_{t-1})$
- observation model: $P(\mathbf{z}_t | \mathbf{x}_t)$
- initial distribution: π_0

State transition matrix $\mathbf{A} = \{A_{ij}\}$

$$A_{ij} \equiv P(\mathbf{x}_t = j | \mathbf{x}_{t-1} = i)$$

Observation model (discrete or continuous):

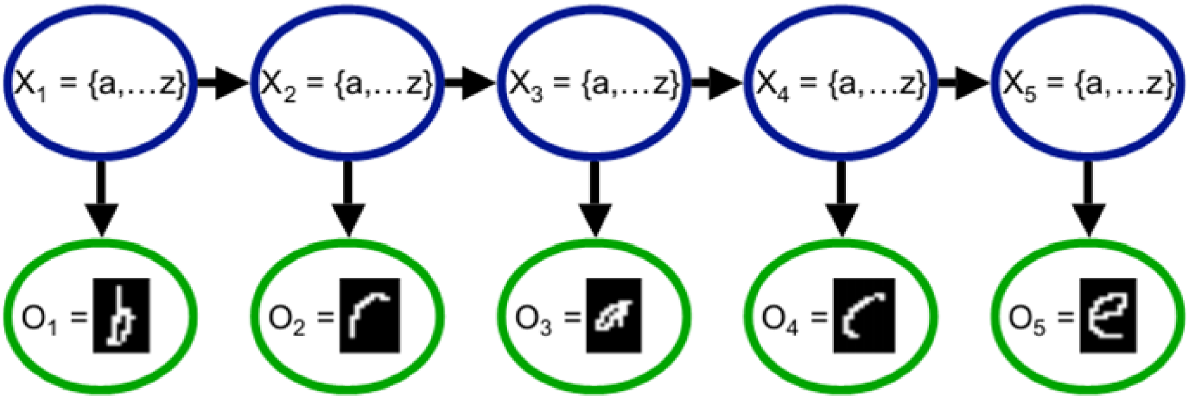
$$b_k(\mathbf{z}_t) \equiv P(\mathbf{z}_t | \mathbf{x}_t = k)$$

Initial probabilities:

$$\pi_0 = P(\mathbf{x}_0)$$

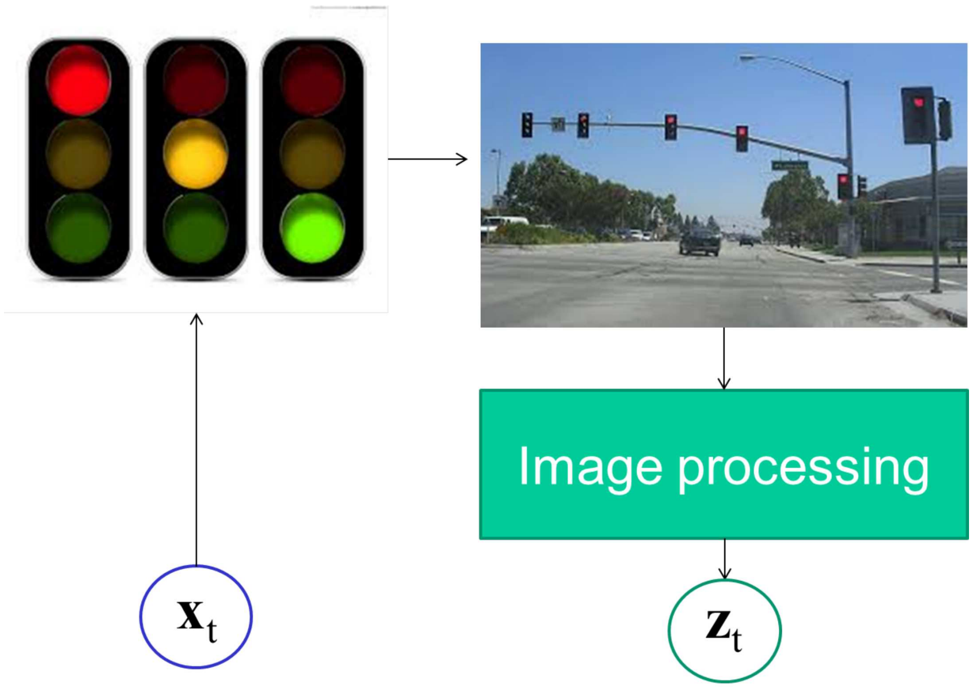
HMM examples of applications

Handwriting recognition



Similar structure for speech/gesture/activity recognition.

HMM examples of applications



HMM factorization

Application of chain rule on HMM:

$$P(\mathbf{x}_{0:T}, \mathbf{z}_{1:T}) = P(\mathbf{x}_0)P(\mathbf{z}_0|\mathbf{x}_0)P(\mathbf{x}_1|\mathbf{x}_0)P(\mathbf{z}_1|\mathbf{x}_1)P(\mathbf{x}_2|\mathbf{x}_1)P(\mathbf{z}_2|\mathbf{x}_2)\ldots$$

HMM inference

Given HMM $= \langle \mathbf{X}, \mathbf{Z}, \pi_0 \rangle$,

Filtering

$$P(\mathbf{x}_T = k|\mathbf{z}_{1:T}) = \frac{\alpha_T^k}{\sum_j \alpha_T^j}$$

Smoothing

$$P(\mathbf{x}_t = k|\mathbf{z}_{1:T}) = \frac{\alpha_t^k \beta_t^k}{\sum_j \alpha_t^j \beta_t^j}$$

Forward step

Forward iterative steps to compute

$$\alpha_t^k \equiv P(\mathbf{x}_t = k, \mathbf{z}_{1:t})$$

- For each state k do:
  - $\alpha_0^k = \pi_0 b_k(\mathbf{z}_0)$
- For each time $t = 1, \ldots, T$ do:
  - For each state k do:
    - $\alpha_t^k = b_k(\mathbf{z}_t) \sum_j \alpha_{t-1}^j A_{jk}$

Backward step

Backward iterative steps to compute

$$\beta_t^k \equiv P(\mathbf{z}_{t+1:T} | \mathbf{x}_t = k)$$

- For each state k do:
  - $\beta_T^k = 1$
- For each time $t = T-1, \ldots, 1$ do:
  - For each state k do:
    - $\beta_t^k = \sum_j \beta_{t+1}^j A_{kj} b_j(\mathbf{z}_{t+1})$

Learning in HMM

Given output sequences, determine maximum likelihood estimate of the parameters of the HMM (*transition and emission probabilities*).

Case 1: states can be observed at training time

Transition and observation models can be estimated with statistical analysis

$$A_{ij} = \frac{|\{i \rightarrow j \ transitions\}|}{|\{i \rightarrow * \ transitions\}|}$$

$$b_k(v) = \frac{|\{observe \ v \wedge state \ k\}|}{|\{observe \ * \wedge state \ k\}|}$$

Sapienza University of Rome, Italy - Machine Learning (2018/2019)

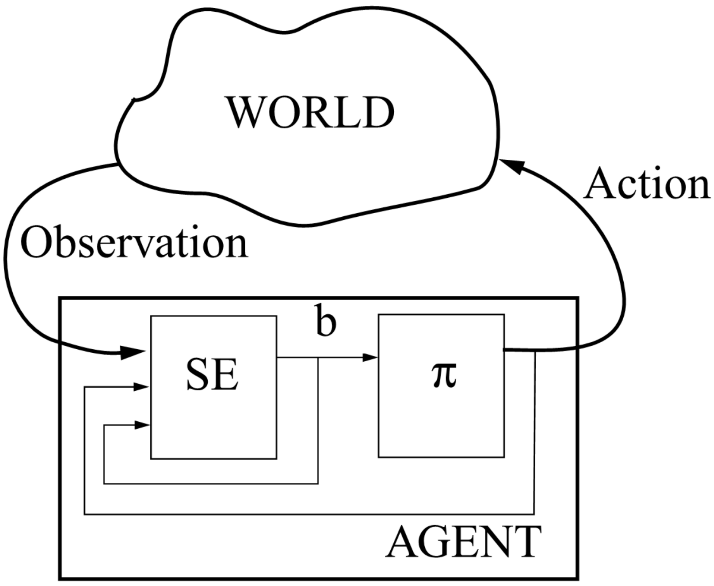
Learning in HMM

Case 2: states cannot be observed at training time

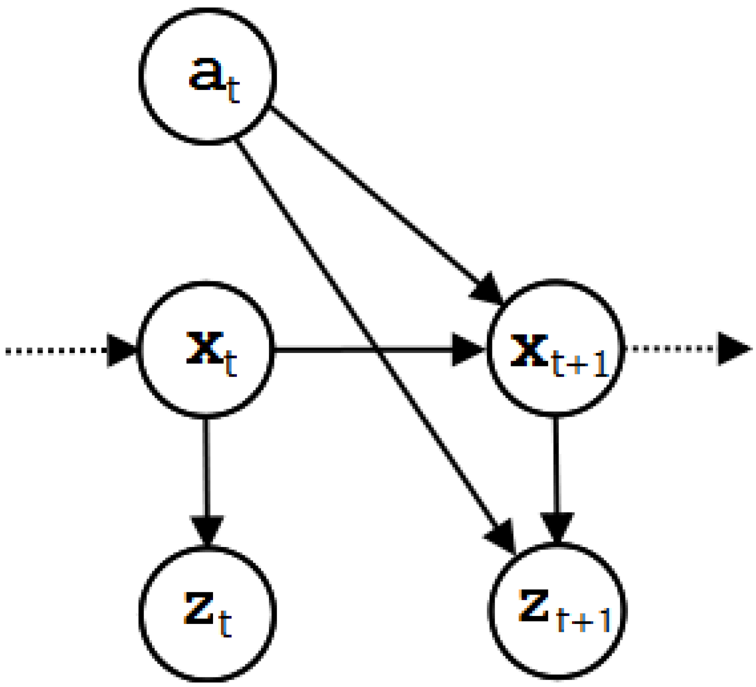
Compute a **local** maximum likelihood with an Expectation-Maximization (EM) method (e.g., Baum-Welch algorithm).

POMDP agent

Combines decision making of MDP and non-observability of HMM.



POMDP graphical model



POMDP representation

$$POMDP = \langle \mathbf{X}, \mathbf{A}, \mathbf{Z}, \delta, r, o \rangle$$

- $\mathbf{X}$ is a set of states
- $\mathbf{A}$ is a set of actions
- $\mathbf{Z}$ is a set of observations
- $P(\mathbf{x}_0)$ is a probability distribution of the initial state
- $\delta(\mathbf{x}, a, \mathbf{x}') = P(\mathbf{x}'|\mathbf{x}, a)$ is a probability distribution over transitions
- $r(\mathbf{x}, a)$ is a reward function
- $o(\mathbf{x}', a, \mathbf{z}') = P(\mathbf{z}'|\mathbf{x}', a)$ is a probability distribution over observations

Example: tiger problem

Two closed doors hide a treasure and a tiger.

- $\mathbf{X} = \{s_L, s_R\}$
- $\mathbf{A} = \{Open_L, Open_R, Listen\}$
- $\mathbf{Z} = \{t_L, t_R\}$
- $P(\mathbf{x}_0) =< 0.5, 0.5 >$
- $\delta(\mathbf{x}, a, \mathbf{x}')$ *Listen* does not change state, *Open* actions restart the situation with 0.5 probability between s_L, s_R
- $r(\mathbf{x}, a) = 10$ if opening the treasure door, -100 if opening the tiger door, -1 if listening
- $o(\mathbf{x}', a, \mathbf{z}') = 0.85$ correct perception, 0.15 wrong perception

Solution concept for POMDP

Solution: *policy*, but we do not know the states!

Option 1: map from history of observations to actions
- histories are too long!

Option 2: belief state
- probability distribution over the current state

Sapienza University of Rome, Italy - Machine Learning (2018/2019)

Belief MDP

Belief $b(\mathbf{x})$ = probability distribution over the states.

POMDP can be described as an MDP in the belief states, but belief states are infinite.

- $\mathbf{B}$ is a set of belief states
- $\mathbf{A}$ is a set of actions
- $\tau(b, a, b')$ is a probability distribution over transitions
- $\rho(b, a, b')$ is a reward function

Policy: $\pi : \mathbf{B} \mapsto \mathbf{A}$

Computing Belief States

Given current belief state b, action a and observation $\mathbf{z}'$ observed after execution of a, compute the next belief state $b'(\mathbf{x}')$

$$\begin{aligned} b'(\mathbf{x}') &\equiv SE(b, a, \mathbf{z}') \equiv P(\mathbf{x}'|b, a, \mathbf{z}') \\ &= \frac{P(\mathbf{z}'|\mathbf{x}', b, a)P(\mathbf{x}'|b, a)}{P(\mathbf{z}'|b, a)} \\ &= \frac{P(\mathbf{z}'|\mathbf{x}', a)\sum_{\mathbf{x}\in\mathbf{X}} P(\mathbf{x}'|b, a, \mathbf{x})P(\mathbf{x}|b, a)}{P(\mathbf{z}'|b, a)} \\ &= \frac{o(\mathbf{x}', a, \mathbf{z}')\sum_{\mathbf{x}\in\mathbf{X}} \delta(\mathbf{x}, a, \mathbf{x}')b(\mathbf{x})}{P(\mathbf{z}'|b, a)} \end{aligned}$$

Sapienza University of Rome, Italy - Machine Learning (2018/2019)

Belief MDP transition and reward functions

Transition function

$$\tau(b, a, b') = P(b'|b, a) = \sum_{\mathbf{z}\in\mathbf{Z}} P(b'|b, a, \mathbf{z})P(\mathbf{z}|b, a)$$

$$P(b'|b, a, \mathbf{z}) = 1 if\ b' = SE(a, b, \mathbf{z}),\ 0\ otherwise$$

Reward function

$$\rho(b, a) = \sum_{\mathbf{x}\in\mathbf{X}} b(\mathbf{x})r(\mathbf{x}, a)$$

Value function in POMDP

$$V(b) = \max_{a \in \mathbf{A}} [\rho(b,a) + \gamma \sum_{b'} (\tau(b,a,b') V(b'))]$$

Replacing $\tau(b,a,b')$ and $\rho(b,a)$ and considering that $P(b'|b,a,\mathbf{z}) = 1$, if $b' = SE(a,b,\mathbf{z}) = b^a_{\mathbf{z}}$, and 0 otherwise

$$V(b) = \max_{a \in \mathbf{A}} [\sum_{\mathbf{x} \in \mathbf{X}} b(\mathbf{x}) r(\mathbf{x},a) + \gamma \sum_{\mathbf{z} \in \mathbf{Z}} P(\mathbf{z}|b,a) V(b^a_{\mathbf{z}})]$$

Sapienza University of Rome, Italy - Machine Learning (2018/2019)

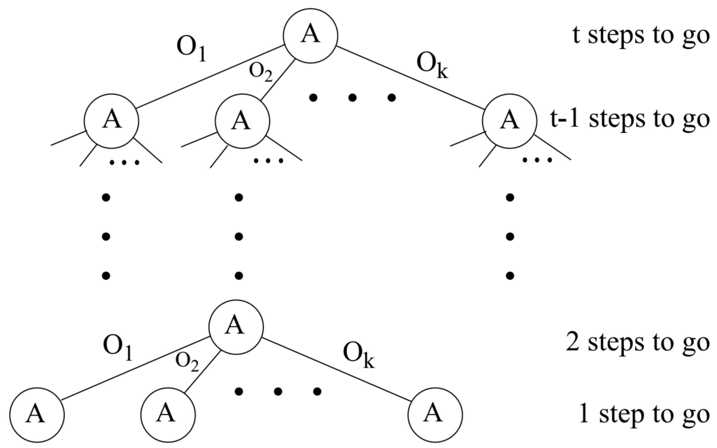
Value iteration for belief MDP

- Discretize the distributions $b(\mathbf{x})$
- Apply value iteration on the discretized belief MDP

A similar method can be devised for any MDP solving technique.

Solution concept in POMDP

Policy trees



Value function for tiger problem

One-step policies: $\pi_1 = Open_L$, $\pi_2 = Open_R$, $\pi_3 = Listen$

$$\alpha_{\pi_1} = \langle -100, 10 \rangle$$
$$\alpha_{\pi_2} = \langle 10, -100 \rangle$$
$$\alpha_{\pi_3} = \langle -1, -1 \rangle$$

One-step optimal value function:

$$V^{(1)}(b) = \max_{\pi} b\, \alpha_{\pi}$$

Value function for tiger problem

Two-step policies:
$\pi_1 = Listen; (t_L : Listen, t_R : Listen) \rightarrow \alpha_{\pi_1} = \langle -2, -2 \rangle$
$\pi_2 = Listen; (t_L : Open_R, t_R : Open_L) \rightarrow \alpha_{\pi_2} = \langle -7.5, -7.5 \rangle$
$\pi_3 = Open_L; (t_L : Open_L, t_R : Open_L) \rightarrow \alpha_{\pi_3} = \langle -145, -35 \rangle$
$\pi_4 = Open_L; (t_L : Listen, t_R : Listen) \rightarrow \alpha_{\pi_4} = \langle -101, 9 \rangle$
$\pi_5 = Open_R; (t_L : Listen, t_R : Listen) \rightarrow \alpha_{\pi_5} = \langle 9, -101 \rangle$
... and many others

Two-step optimal value function:

$$V^{(2)}(b) = \max_{\pi} b\, \alpha_{\pi}$$

Value function for tiger problem

Three-step policies:
$\pi_1 = Listen; Listen; (t_L, t_L : Open_R, t_R, t_R : Open_L, t_L, t_R \text{ or } t_R, t_L : Listen)$
... and many many others ...

Three-step optimal value function:

$$V^{(3)}(b) = \max_{\pi} b\, \alpha_{\pi}$$

References

Leslie Pack Kaelbling, Michael L. Littman, Anthony R. Cassandra.
Planning and acting in partially observable stochastic domains.
Artificial Intelligence, vol. 101, issues 12, 1998, pages 99134.