CSE276C - Markov Chains

Henrik I. Christensen

Computer Science and Engineering
University of California, San Diego

November 2023

Introduction

- How do we model temporal "discrete" processes with associated uncertainty?
- Lots of examples in robotics
  - Executing a plan
  - Modeling traffic
  - Receiving packages for logistics
- Basic coverage of the underlying theory

Independent Trials

- A set of possible outcomes $X_1, X_2, \ldots$ is given
- Each outcome has an associated probability p_k
- The probability of a samples sequence is given by

$$P\{(X_{j0}, X_{j1}, \ldots, X_{jk})\} = p_{j0} p_{j1} \cdots p_{jk}$$

Markov Chains – Introduction

- The outcome of any trial dependent on the outcode of the directly precedings trial only
- **Conditional Probability** p_{jk}: given X_j has occured at some trial the probability of X_k at the next trial
- a_k is the probability of X_k at the initial trial
- I.e.:

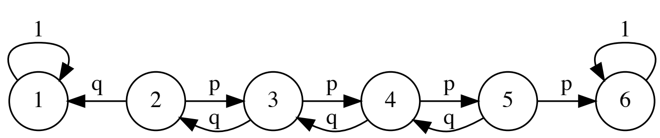
$$\begin{aligned} P\{(X_j, X_k)\} &= a_k p_{jk} \\ P\{(X_j, X_k, X_l)\} &= a_j p_{jk} p_{kl} \\ P\{(X_{j0}, X_{j1}, \ldots, X_{jk})\} &= a_{j0} p_{j1} \cdots p_{jk} \end{aligned}$$

Example – Random Walk

- How would you model a random walk?

Example – Random Walk

- How would you model a random walk?
- Events: $\{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$
- $p_{ij} = 0$ if $|j - k| > 1$
- $p_{ij} = \frac{1}{2}$ for $|i - j| = 1$



Formalizing things

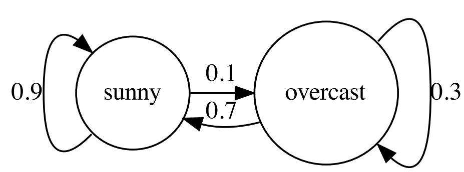
- The chain is in a **state** X_t at time t.
- The **state space** of a chain is the value X can take on, i.e., $S = \{1, 2, 3, 4, 5, 6\}$. Let the size of S be N (possibly infinite)
- A **trajectory** of a chain is the set of values of X over time, say $X_0, X_1, X_2, \ldots$ The trajectory is the "path" through a chain.
- The **Markov Property** implies that the future state/trajectory only depends upon the current state, i.e.:

$$P(X_{t+1} = s | X_t = s_t, X_{t-1} = s_{t-1}, \ldots, X_0 = s_0) = P(X_{t+1} = s | X_t = s_t)$$

- A sequence of discrete events random variables can be considered a Markov chain if it satisfies the above property

Transition matrix

- We have already seen multiple **transition diagrams** as shown below for San Diego weather



- We can capture the same information in a **transition matrix** – $(S \times S)$ that details the transitions between states

$$\begin{pmatrix} 0.9 & 0.1 \\ 0.7 & 0.3 \end{pmatrix}$$

- The transition matrix is one of the most important tools for analyzing Markov Chains

Transition Matrix

- The transition matrix is frequently denoted $P = (p_{ij})$
- In the transition matrix P:
  - the ROW represent NOW or FROM (X_t)
  - the COLUMNS present NEXT or TO (X_{t+1})
  - an entry (i,j) is the CONDITIONAL probability that NEXT (j) is happening given that NOW (i). Expressed as

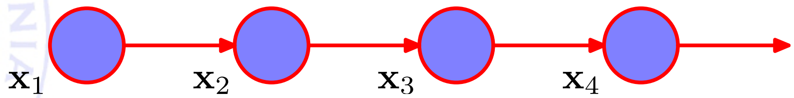
$$p_{ij} = P(X_{t+1}|X_t)$$

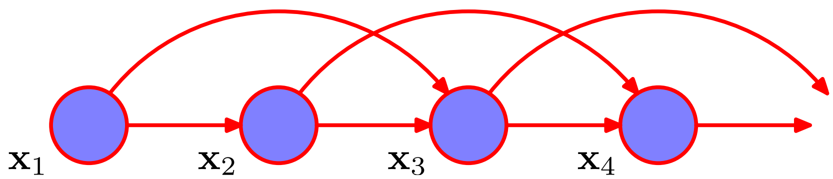
  - Square ($N \times N$) matrix
  - Rows sum to 1
  - Columns do not sum to 1

Initial state

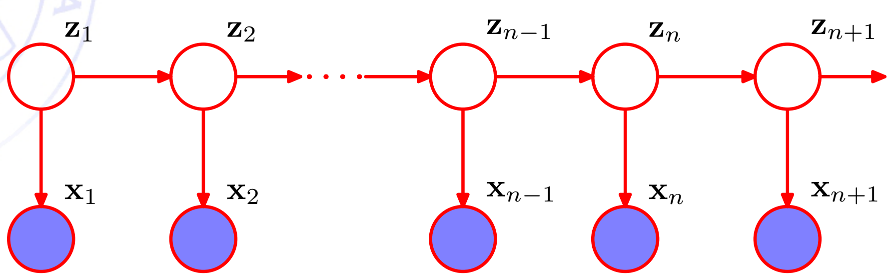
- The Markov chair also has an initial state X_0 which is a distribution over possible start states
- The initial distirbution is represented by the earlier mentioned a_i

Markov chains





Hidden Markov Model



Modelling of HMM's

- We can model the transition probabilities as a table

$$A_{jk} = p(z_{nk} = 1 | z_{n-1,j} = 1)$$

- The conditionals are then (with a 1-out-of-K coding)

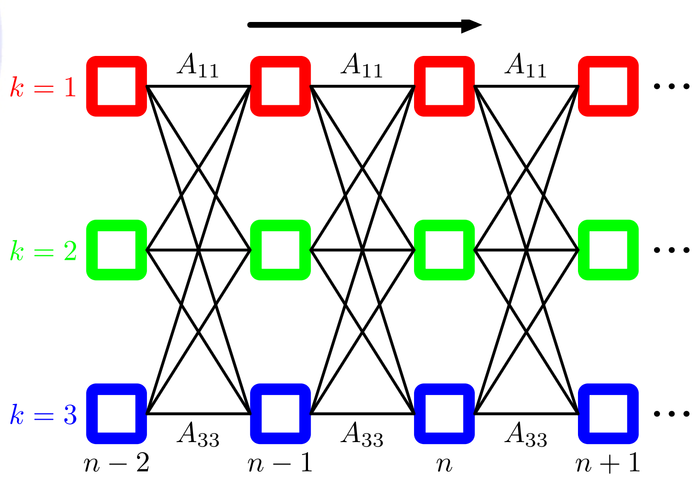
$$p(z_n | z_{n-1}, A) = \prod_{k=1}^{K} \prod_{j=1}^{K} A_{jk}^{z_{n-1,j} z_{nk}}$$

- The per element probability is expressed by $\pi_k = p(z_{1k} = 1)$

$$p(z_1 | \pi) = \prod_{k=1}^{K} \pi_k^{z_{1k}}$$

with $\sum_k \pi_k = 1$

Illustration of HMM



Maximum likelihood for the HMM

- If we observe a set of data $X = \{x_1, \ldots, x_N\}$ we can estimate the parameters using ML

$$p(X|\theta) = \prod_Z p(X, Z|\theta)$$

- I.e. summation of paths through lattice
- We can use EM as a strategy to find a solution
- E-Step: Estimation of $p(Z|X, \theta^{old})$
- M-Step: Maximize over θ to optimize

ML solution to HMM

- Define

$$\begin{aligned}
Q(\theta, \theta^{old}) &= \sum_Z p(Z|X, \theta^{old}) \ln p(X, Z|\theta) \\
\gamma(z_n) &= p(z_n|X, \theta^{old}) \\
\gamma(z_{nk}) &= E[z_{nk}] = \sum_z \gamma(z) z_{nk} \\
\xi(z_{n-1}, z_n) &= p(z_{n-1}, z_n|X, \theta^{old}) \\
\xi(z_{n-1,j}, z_{nk}) &= \sum_z \gamma(z) z_{n-1,j} z_{nk}
\end{aligned}$$

ML solution to HMM

- The quantities can be computed

$$\pi_k = \frac{\gamma(z_{1k})}{\sum_{j=1}^{K} \gamma(z_{1j})}$$

$$A_{jk} = \frac{\sum_{n=2}^{N} \xi(z_{n-1,j}, z_{nk})}{\sum_{l=1}^{K}\sum_{n=2}^{N} \xi(z_{n-1,j}, z_{nl})}$$

- Assume $p(x|\phi_k) = N(x|\mu_k, \Sigma_k)$ so that

$$\mu_k = \frac{\sum_n \gamma(z_{nk}) x_n}{\sum_n \gamma(z_{nk})}$$

$$\Sigma_k = \frac{\sum_n \gamma(z_{nk})(x_n - \mu_k)(x_n - \mu_k)^T}{\sum_n \gamma(z_{nk})}$$

- How do we efficiently compute $\gamma(z_{nk})$?

Forward-Backward / Baum-Welch

- How can we efficiently compute $\gamma()$ and $\xi(.,.)$?
- Remember the HMM is a tree model
- Using message passing we can compute the model efficiently (remember earlier discussion?)
- We have two parts to the message passing forward and backward for any component
- We have

$$\gamma(z_n) = p(z_n|X) = \frac{P(X|z_n)p(z_n)}{p(X)}$$

- From earlier we have

$$\gamma(z_n) = \frac{\alpha(z_n)\beta(z_n)}{p(X)}$$

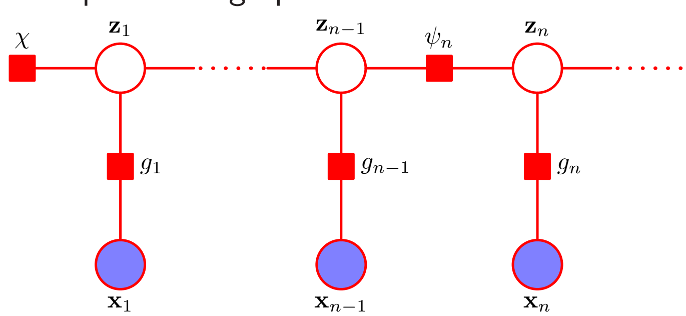
Forward-Backward

- We can then compute

$$
\begin{aligned}
\alpha(z_n) &= p(x_n|z_n)\sum_{z_{n-1}} \alpha(z_{n-1})p(z_n|z_{n-1}) \\
\beta(z_n) &= \sum_{z_{n+1}} \beta(z_{n+1})p(x_{n+1}|z_{n+1})p(z_{n+1}|z_n) \\
p(X) &= \sum_{z_n} \alpha(z_n)\beta(z_n) \\
\xi(z_{n-1}, z_n) &= \frac{\alpha(z_{n-1})p(x_n|z_n)p(z_n|z_{n-1})\beta(z_n)}{p(X)}
\end{aligned}
$$

Sum-product algorithms for the HMM

- Given the HMM is a tree structure
- Use of sum-product rule to compute marginals
- We can derive a simple factor graph for the tree



Sum-product algorithms for the HMM

- We can then compute the factors

$$
\begin{aligned}
h(z_1) &= p(z_1)p(x_1|z_1) \\
f_n(z_{n-1}, z_n) &= p(z_n|z_{n-1})p(x_n|z_n)
\end{aligned}
$$

- The update factors $\mu_{f_n \to z_n}(z_n)$ can be used to derive message passive with $\alpha(.)$ and $\beta(.)$

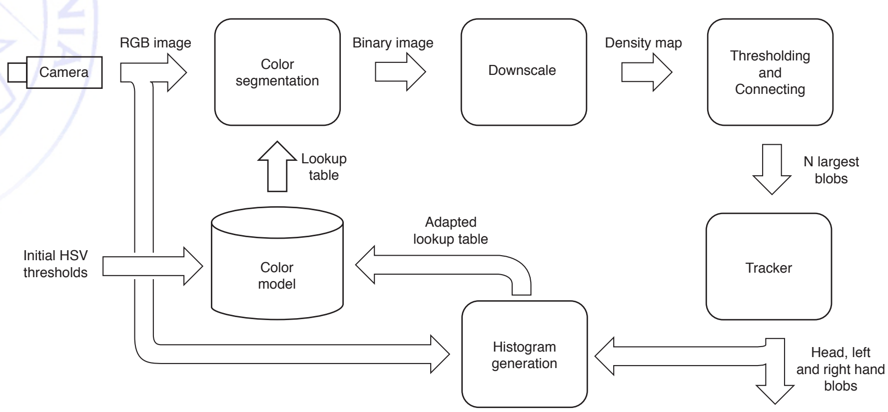
Viterbi Algorithm

- Using the message passing framework it is possible to determine the most likely solution (ie best recognition)
- Intuitively
- Keep only track of the most likely / probably path through the graph
- At any time there are only K possible paths to maintain
- Basically a greedy evaluation of the best solution

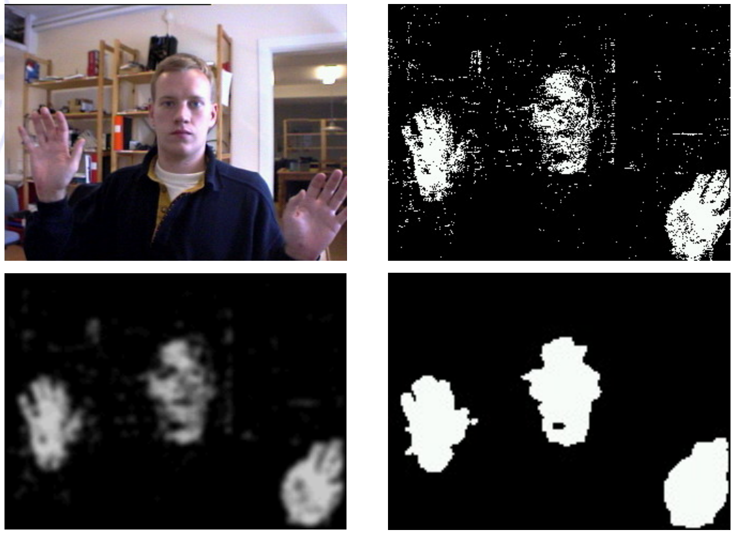
Small example of gesture tracking

- Tracking of hands using an HMM to interpret track
- Pre-process images to generate tracks
    - Color segmentation
    - Track regions using Kalman Filter
    - Interpret tracks using HMM

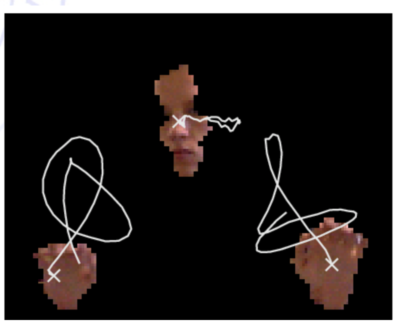
Pre-process architecture

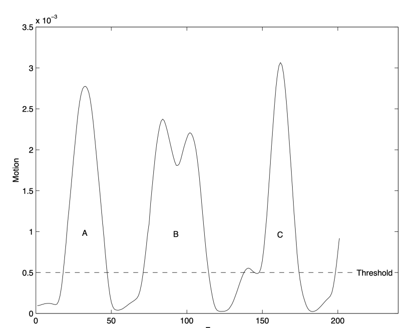


Basic idea

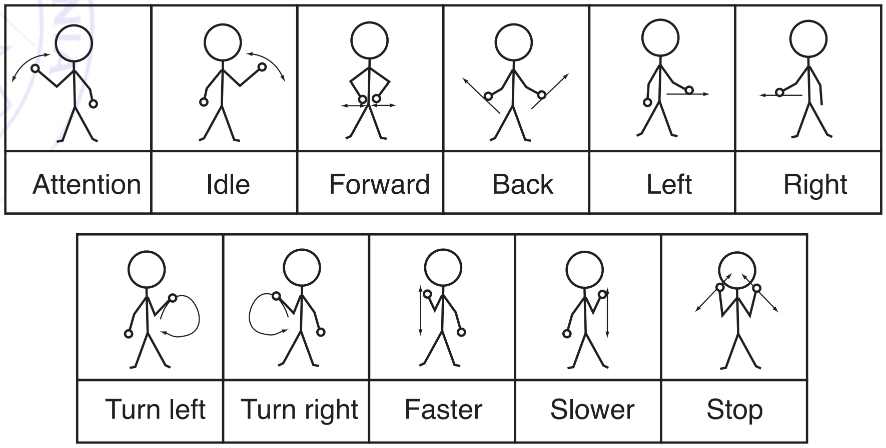


Tracking





Motion Patterns



Evaluation

- Acquired 2230 image sequences
- Covering 5 people in a normal living room
- 1115 used for training
- 1115 sequences were used for evaluation
- Capture of position and velocity data

Rec Rates	Position	Velocity	Combined
Result [%]	96.6	88.7	99.5

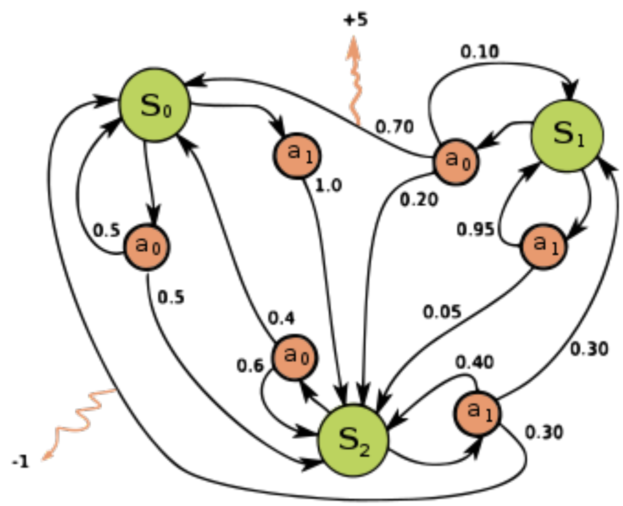
Example Timing

Phase	Time/frame[ms]
Image Transfer	4.3
Segmentation	0.6
Density Est	2.1
Connect Comp	2.1
Kalman Filter	0.3
HMM	21.0
Total	30.4

Markov Decision Processes

- Not all processes are passive.
- In some cases you can introduce actions that drive changes in states
- In robotics a popular class of such problems are Markov Decision Processes (MDP)
- Consider a 4-tuple
  - (S, A, P_a, R_a) where
  - S is the set of possible states
  - A is the set of possible actions, term the action space
  - $P_a(s, s') = P(X_{t+1} = s' | X_t = s, a_t = a)$ is the probability action a in state s will result in reaching state s' at time t+1
  - $R_a(s, s')$ is an immediate reward received from transition from s to s' due to action a

MDP example



MDP Objective

- The goal of the MDP is to find a good policy for a decision maker
- The policy $\pi(s)$ specifies the optimal action in each state and the resulting execution is a Markov chain
- Objective is to choose a policy π that maximizes the cummulative reward, typically with a discount factor, i.e.

$$E\left[\sum_t \gamma^t R_{a_t}(s_t, s+_{t+1})\right]$$

- where γ is a discount factor $0 \leq \gamma \leq 1$ typically close to 1. A lower discount factor will encourage actions earlier

Algorithms

- The MDP can be solved using linear programming
  - The optimal policy can be found using value function iteration.
    1. Update Value: $V(s) = \sum_{s'} P_\pi(s,s')(R_\pi(s,s') + \gamma V(s'))$
    2. Policy update: $\pi(s) = \arg\max_a \left\{ \sum_{s'} P_\pi(s,s')(R_\pi(s,s') + \gamma V(s')) \right\}$
- If the reward function is unknown this is an RL problem
  - $Q(s,a) = \sum_{s'} P_a(s,s')(R_a(s,s') + \gamma V(s'))$
  - Collecting (s, a, s') triplets allow optimization and estimation of Q (so also as Q-learning)

Summary

- Many types of time sequences can be described as a Markov chain or a Hidden Markov Model (HMM)
- The underlying theory is simple to understand
- You can describe the model as a graph / tree structure
- It is possible to capture the model with a transition matrix and an initial distirbution
- It is possible to learn / adapt the probabilities over time
- Widely used for temporal processes such as gestures, pose analysis, navigation, ...
- Numerous toolkits available for analysis and learning models.