

Machine Learning, Lecture 12: Markov chains and hidden Markov models

S. Nõmm

¹Department of Computer Science, Tallinn University of Technology

23.04.2015

Modeling sequential data

- ▶ Speech recognition
- ▶ Machine translation
- ▶ Handwriting recognition
- ▶ Biological sequences
- ▶ Processes originating from the area of business and finance
- ▶ Robotics (location of the robot)
- ▶ Health monitoring

Sequential processes

- ▶ Consider a system with N discrete states. (Some times referred as the system which may occupy one of N states at each time instance t).
- ▶ The processes, in which the state evolution is random over time, are called stochastic processes.
- ▶ Any joint distribution over sequences of states can be factored according to the chain rule into a product of conditional distributions:

$$p(x_0, x_1, \dots, x_T) = p(x_0) \prod_{t=1}^T p(x_t \mid x_0, \dots, x_{t-1})$$

Example: language modeling

- ▶ What is the probability of a sentence: *The cat sat on the mat* ?
- ▶ According to the chain rule:

$$\begin{aligned} p(\text{The cat sat on the mat}) = & \\ p(\text{The}) \times & \\ p(\text{cat} \mid \text{The}) \times & \\ p(\text{sat} \mid \text{The cat}) \times & \\ p(\text{on} \mid \text{The cat sat}) \times & \\ p(\text{the} \mid \text{The cat sat on}) \times & \\ p(\text{mat} \mid \text{The cat sat on the}) \times & \end{aligned}$$

- ▶ Problem: infeasible amount of data necessary to learn all the statistics reliably.

Markov process

- ▶ Let us suppose that *the future is independent of the past given the present*.

$$p(x_{t-1}, x_{t+1} \mid x_t) = p(x_{t-1} \mid x_t) \cdot p(x_{t+1} \mid x_t)$$

referred as *Markov Assumption*

- ▶ The processes where the next step depends only on the current state:

$$p(x_{t+1} \mid x_0, \dots, x_t) = p(x_{t+1} \mid x_t)$$

are called *Markov processes*

- ▶ Combining the Markov assumption with the chain rule one gets the probability of the whole sequence as:

$$p(x_0, x_1, \dots, x_T) = p(x_0) \prod_{t=1}^T p(x_t \mid x_{t-1})$$

Language modeling with Markov process

- ▶ What is the probability of the sentence *The cat sat on the mat?*
- ▶ according to the Markov assumption and the chain rule:

$$\begin{aligned} p(\text{The cat sat on the mat}) = & \\ p(\text{The}) \times & \\ p(\text{cat} \mid \text{The}) \times & \\ p(\text{sat} \mid \text{cat}) \times & \\ p(\text{on} \mid \text{sat}) \times & \\ p(\text{the} \mid \text{on}) \times & \\ p(\text{mat} \mid \text{the}) \times & \end{aligned}$$

- ▶ Obviously one has to estimate much smaller number of the parameters.

Markov Chain

- ▶ The sequence generated by a Markov process is called the Markov chain
- ▶ Usually it is assumed that the Markov chain is time-invariant or stationary - this means that the probabilities $p(x_t | x_{t-1})$ do not depend on time.
- ▶ For example in language modeling the probability $p(\text{the} | \text{on})$ does not depend on the positions of these words in the sentence.
- ▶ This is an example of parameter tying since the parameter is shared by multiple variables

Markov model specification

- ▶ A stationary Markov model with N states can be described by an $N \times N$ transition matrix:

$$Q = \begin{bmatrix} q_{11} & \dots & q_{1N} \\ \dots & \dots & \dots \\ q_{N1} & \dots & q_{NN} \end{bmatrix}$$

where $q_{ij} = p(x_t = i \mid x_{t-1} = j)$

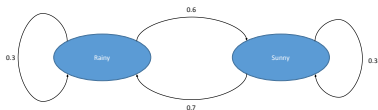
- ▶ Constraints on valid transition matrices:

$$q_{ij} \geq 0, \quad \sum_{i=1}^N q_{i,j} = 1, \text{ for all } j$$

State transition diagram

- ▶ State transition matrices can be visualized with a state transition diagram
- ▶ State transition diagram is a directed graph where arrows represent legal transitions.
- ▶ Drawing state transition diagrams is most useful when N is small and Q is sparse.

$$Q = \begin{bmatrix} 0.4 & 0.6 \\ 0.7 & 0.3 \end{bmatrix}$$



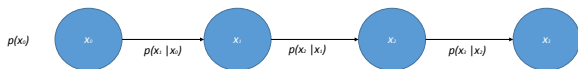
Graphical models

- ▶ A way of specifying conditional independencies
- ▶ *Directed graphical model*: DAG
- ▶ Nodes are random variables
- ▶ A node's distribution depends on its parents
- ▶ Joint distribution: $p(X) = \prod_i p(x_i \mid \text{Parents}_i)$
- ▶ A node's value conditional on its parents is independent of other ancestors

Markov chain as a graphical model

$$p(x_0, x_1, \dots, x_T) = p(x_0) \prod_{t=1}^T p(x_t | x_{t-1})$$

- ▶ Graph interpretation differs from state transition diagrams:
- ▶ Nodes represent state values at particular times
- ▶ Edges represent Markov properties



Markov chain training

- ▶ Let us assume that training data is given in the form of sequences
- ▶ One can count the number of occurrence of any two consecutive values
- ▶ For example, we can count how many times occurs the word pair "of the" in the training text.
- ▶ For obtaining the quantity $p(\text{the} \mid \text{of})$ we have to divide with the number of times the word "of" occurs in the training data:

$$p(\text{the} \mid \text{of}) = \frac{p(\text{of the})}{p(\text{of})} = \frac{\text{Count}(\text{of the})}{\text{Count}(\text{of})}$$

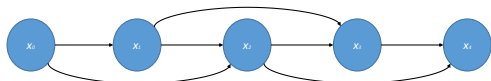
- ▶ In general, if $N_{i,j}$ is the number of times the value i is followed by the value j :

$$p(x_t = j \mid x_{t-1} = i) = \frac{p(x_{t-1} = i, x_t = j)}{p(x_{t-1} = i)} = \frac{N_{i,j}}{\sum_j N_{i,j}}$$

Markov chain order

- ▶ The Markov chain presented in previous slides is called *first-order* Markov model.
- ▶ It is also called *bigram* model (especially in language modelling)
- ▶ The marginal probabilities $p(x_t)$ are called *unigram* probabilities
- ▶ In the unigram model all the variables are independent
 $p(x_0, x_1, \dots, x_T) = \prod_t p(x_t)$
- ▶ One can also construct higher order Markov chains: a second order model operates with trigrams:

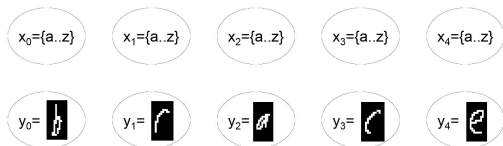
$$p(x_t \mid x_0, \dots, x_{t-1}) = p(x_t \mid x_{t-2}, x_{t-1})$$



Hidden Markov models

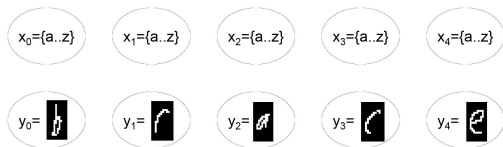
- ▶ Few realistic sequential processes directly satisfy the Markov assumption.
- ▶ Markov chains cannot capture long-range correlations between observations.
- ▶ Increasing the order leads the number of parameters to blow up
- ▶ This motivates the hidden Markov models (HMM)
- ▶ In HMM there is an underlying hidden process that can be modelled with a first-order Markov chain
- ▶ The data is the noisy observation of this process.

HMM: handwriting recognition



- ▶ We can only observe the handwritten character images
- ▶ The hidden process models the characters written

HMM specification



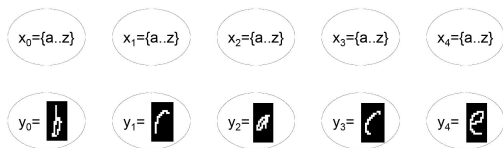
There are three distributions:

$$p(x_0)$$

$$p(x_t | x_{t-1}), \quad t = 1, \dots, T$$

$$p(y_t | x_t), \quad t = 1, \dots, T$$

Joint distribution



The joint distribution of the hidden sequence is:

$$p(x_0, \dots, x_T) \mid y_0, \dots, y_T \propto p(x_0) p(y_0 \mid x_0) \prod_{t=1}^T p(x_t \mid x_{t-1}) p(y_t \mid x_t)$$

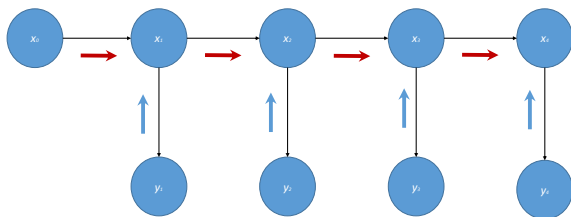
Inference with HMM

- ▶ Compute marginal probabilities of hidden variables
- ▶ Filtering - compute the belief states $p(x_t | y_0, \dots, y_t)$ online
- ▶ Smoothing - compute the probabilities $(x_t | y_0, \dots, y_T)$ offline using all the evidence
- ▶ Find the most likely sequence of hidden variables - Viterbi decoding

Filtering

- ▶ Computing $p(x_t | y_0, \dots, y_t)$ is called filtering, because it reduces noise in comparison to computing just $p(x_t | y_t)$.
- ▶ Filtering is done using forward algorithm
- ▶ Forward algorithm uses dynamic programming - this means the algorithm is recursive but we reuse the already done computations.

Forward algorithm

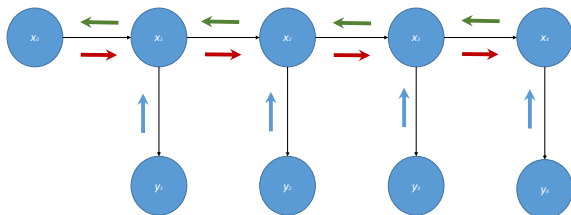


Input:

- ▶ Transition matrix
- ▶ Initial state distribution
- ▶ Observation matrix containing probabilities $p(y_t | x_t)$
- ▶ Compute the forward probabilities:

$$\alpha_t(x_t) = p(x_t | y_{1:t}) = \frac{1}{Z_t} p(y_t | x_t) \sum_{x_{t-1}} p(x_t | x_{t-1}) \alpha_{t-1}(x_{t-1})$$

Smoothing



- ▶ Smoothing computes the marginal probabilities $p(x_t | y_{1:T})$ off line, using all the evidence
- ▶ It is called smoothing, because conditioning on the past and future data the uncertainty will be significantly reduced.
- ▶ Smoothing is performed using forward-backward algorithm.

Forward-backward algorithm

- ▶ Break the chain into past and future:

$$\begin{aligned} p(x_t = j \mid y_{1:T}) &\propto p(x_t = j, y_{t+1:T} \mid y_{1:t}) \\ &\propto p(x_t = j \mid y_{1:t})p(y_{t+1:T} \mid x_t = j) \end{aligned}$$

- ▶ Compute the forward probabilities as before:

$$\alpha_t(x_t) = p(x_t = j \mid y_{1:t})$$

- ▶ Compute the backward probabilities:

$$\beta_t(x_t) = \frac{1}{Z_t} \sum_{x_{t+1}} p(x_{t+1} \mid x_t) p(y_{t+1} \mid x_{t+1}) \beta_{t+1}(x_{t+1})$$

Optimal state estimation

- ▶ Compute the smoothed posterior marginal probabilities

$$p(x_t | y_{1:T}) \propto \alpha_t(x_t)\beta_t(x_t)$$

- ▶ Probabilities measure the posterior confidence in the true hidden states
- ▶ Takes account both the past and the future

Optimal sequence estimation

- ▶ Viterbi algorithm computes

$$\hat{x} = \arg \max p(x_0, \dots, x_T \mid y_1, \dots, y_T)$$

- ▶ Using dynamic programming it finds recursively the probability of the most likely state sequence ending with each x_t :

$$\begin{aligned} \gamma_t(x_t) &= \max_{x_1, \dots, x_{t-1}} p(x_1, \dots, x_t \mid y_{1:t}) \\ &\propto p(y_t \mid x_t) \left[\max_{x_{t-1}} p(x_t \mid x_{t-1}) \gamma_{t-1}(x_{t-1}) \right] \end{aligned}$$

- ▶ A backtracking procedure picks then the most likely sequence.

Learning HMM

- ▶ Let us suppose the latent state sequence is available during training
- ▶ Then the transition matrix, observation matrix and initial state distribution can be estimated by normalized counts

$$\hat{q}_{i,j} = \frac{n(i,j)}{\sum_k n(k,j)}$$
$$\tau_i = \{t \mid x_t = i\}$$
$$\hat{\theta}_i = \frac{1}{|\tau_i|} \sum_{t \in \tau_i} y_t$$

Learning HMM

- ▶ Typically one don't know the hidden state sequences
- ▶ EM algorithm is used, it iteratively maximizes the lower bound on the true data likelihood
- ▶ E-step: Use current parameters to estimate the state using forward-backward
- ▶ M-step: Update the parameters using weighted averages