Markov chains and hidden Markov models

Kairit Sirts

11.04.2014

Modeling sequential data

- Textual data
  - Speech recognition
  - Machine translation
  - Handwriting recognition
- Biological sequences: proteins, genes
- Stock market behaviour over time
- Robot locations over time
- Google PageRank: model sequences of links in internet
- Health conditions over time (insurance, cost-benefit analyses in medical care units)

Sequential processes

- Consider a system which can occupy one of N discrete **states** at each time t: $x_t \in \{1, \ldots, N\}$
- 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**:

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

Example: language modeling

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

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

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

Markov process

- For a **Markov process**, the next state depends only on the current state:

$$p(x_{t+1}|x_0, \ldots, x_t) = p(x_{t+1}|x_t)$$

- This is because we make the **Markov assumption** that **the future is independent of the past given the present**

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

- The probability of a whole sequence can be factored now as:

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

Language modeling with Markov process

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

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

- Advantage: much less parameters to estimate

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**:

$$Q = \begin{bmatrix} q_{11} & \dots & q_{1N} \\ \dots & \dots & \dots \\ q_{N_1} & \dots & q_{NN} \end{bmatrix},$$

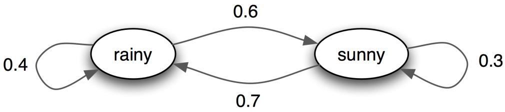
  where $q_{ij} = p(x_t = i | x_{t-1} = j)$

- Constraints on valid transition matrices:

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

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.



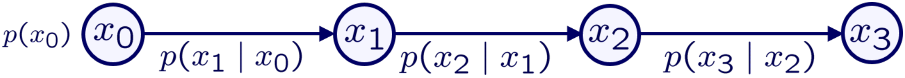
$$Q = \begin{bmatrix} 0.4 & 0.6 \\ 0.7 & 0.3 \end{bmatrix},$$

Quick intro into 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: $p(X) = \prod_i p(x_i|\text{Parents}_i)$
- A node's value conditional on its parents is independent of other ancestors

Markov chain as a graphical model

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



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

Training a Markov chain

- Assume we have training data in the form of sequences (for example lots of text)
- We 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 $p(\text{the}|\text{of})$ we have to divide with the number of times the word "of" occurs in the training data:

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

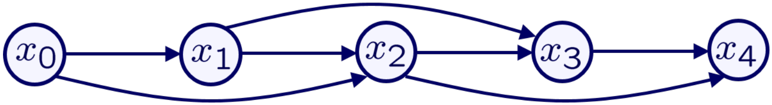
- In general, if N_{ij} is the number of times the value i is followed by the value j:

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

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 $p(x_0, x_1, \ldots, x_T) = \prod_t p(x_t)$
- We can also construct higher order Markov chains: a **second order** model operates with **trigrams**:

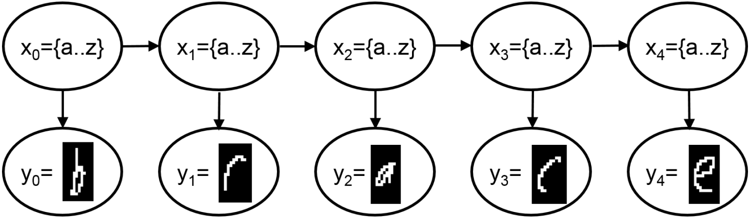
$$p(x_t|x_0, \ldots, x_{t-1}) = p(x_t|x_{t-2}, x_{t-1})$$



Hidden Markov models

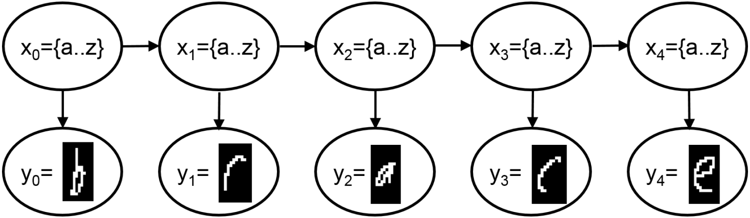
- Few realistic sequential processes directly satisfy the Markov assumption.
- Markov chains cannot capture long-range correlations between observations.
- Increasing the order leads the number of parameters to blow up
- This motivates the **hidden Markov models** (HMM)
- In HMM there is an underlying 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

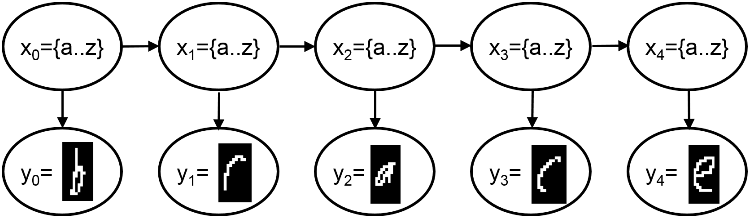
HMM specification



- There are three distributions:

$$p(x_0)$$
$$p(x_t|x_{t-1}), \qquad t = 1 \ldots T$$
$$p(y_t|x_t), \qquad t = 0 \ldots T$$

Joint distribution



- The joint distribution of the hidden sequence is:

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

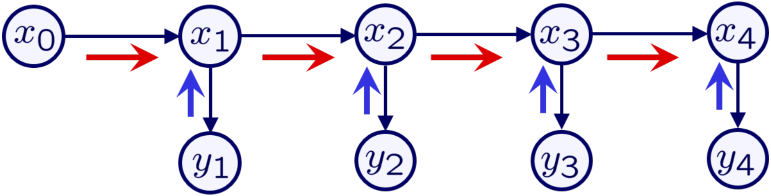
Inference with HMM

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

Filtering

- Computing $p(x_t|y_0,\ldots,y_t)$ 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

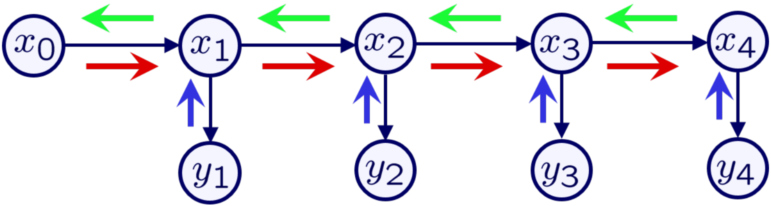


Input:

- 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 $p(x_t|y_{1:T})$ offline, 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:

$$p(x_t = j|y_{1:T}) \propto p(x_t = j, y_{t+1:T}|y_{1:t}) \propto p(x_t = j|y_{1:t})p(y_{t+1:T}|x_t = j)$$

- Compute the forward probabilities as before:

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

- Compute the backward probabilities:

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

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:

$$\hat{x} = \arg\max p(x_0, x_1, \ldots, x_t | y_1, \ldots, y_T)$$

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

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

- A backtracking procedure picks then the most likely sequence.

Learning HMM

- 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

$$\hat{q}_{ij} = \frac{n(i,j)}{\sum_k n(k,j)}$$

$$\tau_i = \{t | x_t = i\}$$

$$\hat{\theta}_i = \frac{1}{|\tau_i|} \sum_{t \in \tau_i} y_t$$

Learning HMM

- Typically we don't know the hidden state sequences
- Then we use EM algorithm that 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

More topics about HMM-s and Markov chains

- Continuous observations
- Everything is Gaussian: Kalman filters
- Whole field of random sampling methods (MCMC - Markov Chain Monte Carlo) are based on Marko chains
- Enable to draw random samples from very complicated distributions