Markov Models and Hidden Markov Models
Inference algorithms for HMMs
Parameter learning for HMMs
Structured hidden states

Models discussed so far: mainly classifiers predicting a categorical (class) variable $y \in \mathcal{Y}$

Often in applications: the hidden state is a structured variable.

Here: the hidden state is given by a **sequence** of categorical variables.

**Application examples:**

- text recognition (printed, handwritten, “in the wild”),
- speech recognition (single word recognition, continuous speech recognition, translation),
- robot self localisation.

Markov Models and Hidden Markov Models on chains: a class of generative probabilistic models for sequences of features and sequences of categorical variables.
Markov Models

Let \( s = (s_1, s_2, \ldots, s_n) \) denote a sequence of length \( n \) with elements from a finite set \( K \).

Any joint probability distribution on \( K^n \) can be written as

\[
p(s_1, s_2, \ldots, s_n) = p(s_1) p(s_2 | s_1) p(s_3 | s_2, s_1) \cdots p(s_n | s_1, \ldots, s_{n-1})
\]

**Definition 1** A joint p.d. on \( K^n \) is a Markov model if

\[
p(s) = p(s_1) p(s_2 | s_1) p(s_3 | s_2) \cdots p(s_n | s_{n-1}) = p(s_1) \prod_{i=2}^{n} p(s_i | s_{i-1})
\]

holds for any \( s = (s_1, s_2, \ldots, s_n) \).
Markov Models

Example 1 (Random walk on a graph)

- Let \( (V, E) \) be a directed graph. A random walk in \( (V, E) \) is described by a sequence \( s = (s_1, \ldots, s_t, \ldots) \) of visited nodes, i.e. \( s_t \in V \).
- The walker starts in node \( i \in V \) with probability \( p(s_1 = i) \).
- The edges of the graph are weighted by \( w : E \to \mathbb{R}_+ \), s.t.

\[
\sum_{j : (i,j) \in E} w_{ij} = 1 \quad \forall i \in V
\]

- If the current position of the walker is \( s_t = i \), it randomly chooses an outgoing edge with probability given by the weights and moves along this edge, i.e.

\[
p(s_{t+1} = j \mid s_t = i) = \begin{cases} 
    w_{ij} & \text{if } (i,j) \in E \\
    0 & \text{otherwise}
\end{cases}
\]

Questions: How does the distribution \( p(s_t) \) behave? Does it converge to some fix-point distribution for \( t \to \infty \)?
Algorithms: Computing the most probable sequence

How to compute the most probable sequence \( s^* \in \arg\max_{s \in K^n} p(s_1) \prod_{i=2}^{n} p(s_i | s_{i-1}) \)?

Take the logarithm of \( p(s) \): \( s^* \in \arg\max_{s \in K^n} \left[ g_1(s_1) + \sum_{i=2}^{n} g_i(s_{i-1}, s_i) \right] \)

and apply dynamic programming: Set \( \phi_1(s_1) \equiv g_1(s_1) \) and compute

\[
\phi_i(s_i) = \max_{s_{i-1} \in K} \left[ \phi_{i-1}(s_{i-1}) + g_i(s_{i-1}, s_i) \right].
\]

Finally, find \( s^*_n \in \arg\max_{s_n \in K} \phi_n(s_n) \) and back-track the solution. This corresponds to searching the best path in the graph
Algorithms: Computing marginal probabilities

How to compute marginal probabilities for the sequence element $s_i$ in position $i$

$$p(s_i) = \sum_{s_1 \in K} \cdots \sum_{s_i \in K} \cdots \sum_{s_n \in K} p(s_1) \prod_{i=2}^{n} p(s_i \mid s_{i-1})$$

Summation over the trailing variables is easily done because:

$$\sum_{s_n \in K} p(s_1) \cdots p(s_{n-1} \mid s_{n-2}) p(s_n \mid s_{n-1}) = p(s_1) \cdots p(s_{n-1} \mid s_{n-2})$$

The summation over the leading variables is done dynamically: Begin with $p(s_1)$ and compute

$$p(s_i) = \sum_{s_{i-1} \in K} p(s_i \mid s_{i-1}) p(s_{i-1})$$
Algorithms: Computing marginal probabilities

This computation is equivalent to a matrix vector multiplication: Consider the values \( p(s_i = k \mid s_{i-1} = k') \) as elements of a matrix \( P_{k'k}(i) \) and the values of \( p(s_i = k) \) as elements of a vector \( \pi_i \). Then the computation above reads as \( \pi_i = \pi_{i-1} P(i) \).

Remark 1

- Notice that the preferred direction (from first to last) in the Definition 1 of a Markov model is only apparent. By computing the marginal probabilities \( p(s_i) \) and by using \( p(s_i \mid s_{i-1})p(s_{i-1}) = p(s_{i-1}, s_i) = p(s_{i-1} \mid s_i)p(s_i) \), we can rewrite the model in reverse order.

- A Markov model is called homogeneous if the transition probabilities \( p(s_i = k \mid s_{i-1} = k') \) do not depend on the position \( i \) in the sequence. In this case the formula \( \pi_i = \pi_1 P^{i-1} \) holds for the computation of the marginal probabilities.
Suppose we are given i.i.d. training data \( \mathcal{T}_m = \{ s^j \in K^n \mid j = 1, \ldots, m \} \) and want to estimate the parameters of the Markov model by the maximum likelihood estimate. This is very easy:

- Denote by \( \alpha(s_{i-1} = \ell, s_i = k) \) the fraction of sequences in \( \mathcal{T}_m \) for which \( s_{i-1} = \ell \) and \( s_i = k \).

- The estimates for the conditional probabilities are then given by
  \[
  p(s_i = k \mid s_{i-1} = \ell) = \frac{\alpha(s_{i-1} = \ell, s_i = k)}{\sum_k \alpha(s_{i-1} = \ell, s_i = k)}.
  \]
Hidden Markov Models

- Let \( s = (s_1, s_2, \ldots, s_n) \) denote a sequence of hidden states from a finite set \( K \).
- Let \( x = (x_1, x_2, \ldots, x_n) \) denote a sequence of features from some feature space \( \mathcal{X} \).

**Definition 2** A joint p.d. on \( \mathcal{X}^n \times K^n \) is a Hidden Markov model if

(a) the prior p.d. \( p(s) \) for the sequences of hidden states is a Markov model, and
(b) the conditional distribution \( p(x \mid s) \) for the feature sequence is independent, i.e.

\[
p(x \mid s) = \prod_{i=1}^{n} p(x_i \mid s_i).
\]

**Example 2** (Text recognition, OCR)

- \( x = (x_1, x_2, \ldots, x_n) \) – sequence of images with characters,
- \( s = (s_1, s_2, \ldots, s_n) \) – sequence of alphabetic characters,
- \( p(s_i \mid s_{i-1}) \) – language model,
- \( p(x_i \mid s_i) \) – appearance model for characters.
Hidden Markov Models

\[ p(x_1 | s_1) \quad p(s_2 | s_1) \quad p(x_2 | s_2) \quad p(s_3 | s_2) \quad p(x_3 | s_3) \quad p(s_4 | s_3) \quad p(x_4 | s_4) \]
(1) Find the most probable sequence of hidden states given the sequence of features:

\[ s^* \in \arg \max_{s \in K^n} p(s_1) \prod_{i=2}^{n} p(s_i \mid s_{i-1}) \prod_{i=1}^{n} p(x_i \mid s_i) \]

Take the logarithm, redefine the \( g \)-s and apply dynamic programming as before for Markov models.

(2) Compute marginal probabilities for hidden states given the sequence of features:

This is now more complicated, because we need to sum over the leading and trailing hidden state variables. Do this by dynamic matrix-vector multiplication from the left and from the right

\[ \phi_i(s_i) = \sum_{s_{i-1}} p(x_i \mid s_i) p(s_i \mid s_{i-1}) \phi_{i-1}(s_{i-1}) \]
\[ \psi_i(s_i) = \sum_{s_{i+1}} p(x_{i+1} \mid s_{i+1}) p(s_{i+1} \mid s_i) \psi_{i+1}(s_{i+1}) \]
Algorithms for HMMs

The (posterior) marginal probabilities are then obtained from

\[ p(s_i \mid \mathbf{x}) \sim \phi_i(s_i) \psi_i(s_i) \]

The computational complexity is \( O(nK^2) \).

(3) Learning the model parameters from training data:

Given i.i.d. training data \( \mathcal{T}_m = \{(\mathbf{x}^j, s^j) \in \mathcal{X}^n \times K^n \mid j = 1, \ldots, m\} \), estimate the parameters of the HMM by the maximum likelihood estimator.

This is done by simple “counting” as before for Markov models.