Lecture 24: Hidden Markov Models

GU4241/GR5241 Statistical Machine Learning

Linxi Liu
April 18, 2017

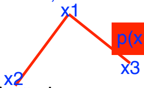
Graphical Model Notation

Conditional independence

Given random variables X, Y, Z, we say that X is **conditionally independent of Y given Z** if

$$P(x|y,z) = P(x|z)$$

high dimensional data, there are a lot of the nodes in the graph;
if can model the conditional independence
if there are is no edge, then the corresponding variable are independent
eg: given x1, x2 and x3 are independent



p(x,y,lz) = p(xlz) * p(ylz)

Notation:

$$X \perp\!\!\!\perp_Z Y$$

In words: Once $Z = z$ is known, the outcome of Y does not provide additional information about X.

we also need to know the direction of the edge and the cost

Graphical models: Idea

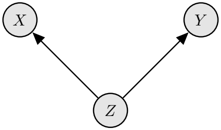
A graphical model represents the dependence structure within a set of random variables by a directed graph. Roughly speaking:

- Each random variable is represented by vertex.
- If Y depends on X, we draw an edge $X \to Y$.

A Simple Example

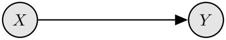
A simple example

The graphical model for $X \perp\!\!\!\perp_Z Y$ looks like this:

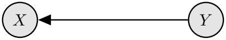


Important

- X and Y are *not* independent, independence holds only conditionally on Z.
- In other words: If we do not observe Z, X and Y are dependent, and we have to change the graph:



or



Graphical Model Notation

Factorizing a joint distribution

The joint probability of random variables $X_1, \ldots, X_m$ can always be factorized as

$$P(x_1, \ldots, x_m) = P(x_m | x_1, \ldots, x_{m-1}) P(x_{m-1} | x_1, \ldots, x_{m-2}) \cdots P(x_1) \ .$$

Note that we can re-arrange the variables in any order. If there are conditional independencies, we can remove some variables from the conditionals:

$$P(x_1, \ldots, x_m) = P(x_m | \mathcal{X}_m) P(x_{m-1} | \mathcal{X}_{m-1}) \cdots P(x_1) \ ,$$

where $\mathcal{X}_i$ is the subset of $X_1, \ldots, X_m$ on which X_i depends.

Definition

Let $X_1, \ldots, X_m$ be random variables. A **(directed) graphical model** represents a factorization of joint distribution $P(x_1, \ldots, x_m)$ as follows:

- Add one vertex for each variable X_i.
- For each variable X_i, add and edge from each variable $X_j \in \mathcal{X}_i$ to X_i.

Graphical Model Notation

Lack of uniqueness

The factorization is usually not unique, since e.g.

$$P(x, y) = P(x|y)P(y) = P(y|x)(x) \ .$$

That means the direction of edges is not generally determined.

Remark

- If we use a graphical model to *define* a model or visualize a model, we decide on the direction of the edges.
- Estimating the direction of edges from data is a very difficult (and very important) problem. This is the subject of a research field called **causal inference** or **causality**.

Overview

z1 > z2 >z3 ….zn we can not observe hidden state
above is generated by markove process
based on the hidden state according to certain dist
x1, x2 , x3, …..xn
(z1 and x1 are independent
but we can not use markov chain to calculate
Our observation x1, x2 …xn)

Motivation

We have already used Markov models to model sequential data. Various important types of sequence data (speech etc) have long-range dependencies that a Markov model does not capture well.

Hidden Markov model

there are two sequences: the hidden state

- A hidden Markov model is a latent variable model in which a sequence of latent (or "hidden") variables is generated by a Markov chain.
- These models can generate sequences of *observations* with long-range dependencies, but the *explanatory* variables (the latent variables) are Markovian.
- It turns out that this is exactly the right way to model dependence for a variety of important problems, including speech recognition, handwriting recognition, and parsing problems in genetics.

Hidden Markov Models

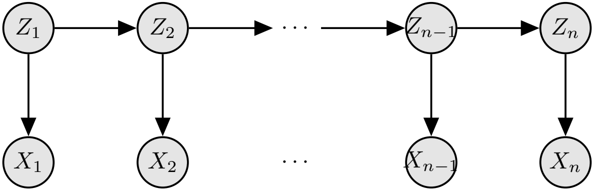
Definition

A (**discrete**) **hidden Markov model (HMM)** consists of:

- A stationary Markov chain $(Q_{\text{init}}, \mathbf{q})$ with states $\{1, \ldots, K\}$, initial distribution Q_{init} and transition matrix $\mathbf{q}$.
- A (discrete) **emission distribution**, given by a conditional probability $P(x|z)$.

The model generates a sequence $X_1, X_2, \ldots$ by:

1. Sampling a sequence $Z_1, Z_2, \ldots$ from the Markov chain $(Q_{\text{init}}, \mathbf{q})$.
2. Sampling a sequence $X_1, X_2, \ldots$ by independently sampling $X_i \sim P(\,.\,|Z_i)$.



Hidden Markov Models

Definition

A (**discrete**) **hidden Markov model (HMM)** consists of:

- A stationary Markov chain $(Q_{\text{init}}, \mathbf{q})$ with states $\{1, \ldots, K\}$, initial distribution Q_{init} and transition matrix $\mathbf{q}$.
- A (discrete) **emission distribution**, given by a conditional probability $P(x|z)$.

The model generates a sequence $X_1, X_2, \ldots$ by:

1. Sampling a sequence $Z_1, Z_2, \ldots$ from the Markov chain $(Q_{\text{init}}, \mathbf{q})$.
2. Sampling a sequence $X_1, X_2, \ldots$ by independently sampling $X_i \sim P(\,.\,|Z_i)$.

In a **continuous HMM**, the variables X_i have continuous distributions, and $P(x|z)$ is substituted by a density $p(x|z)$. The Markov chain still has finite state space $[K]$.

Notation

We will see a lot of sequences, so we use the "programming" notation

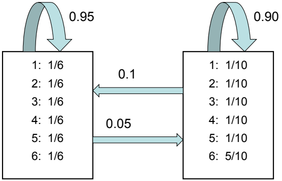
$$x_{1:n} := (x_1, \ldots, x_n)$$

Example: Dishonest Casino

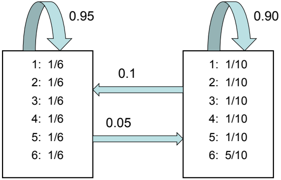
This example is used in most textbooks and is very simple, but it is useful to understand the conditional independence structure.

Problem

- We consider two dice (one fair, one loaded).
- At each roll, we either keep the current dice, or switch to the other one with a certain probability.
- A roll of the chosen dice is then observed.



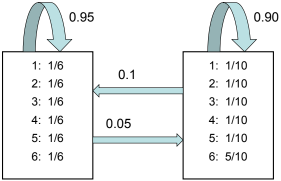
Example: Dishonest Casino



HMM

- States: $Z_n \in \{\text{fair}, \text{loaded}\}$.
- Sample space: $\mathbf{X} = \{1, \ldots, 6\}$.
- Transition matrix: $\mathbf{q} = \begin{pmatrix} 0.95 & 0.05 \\ 0.10 & 0.90 \end{pmatrix}$
- Emission probabilities:
  $P(x|z = \text{fair}) = (1/6, 1/6, 1/6, 1/6, 1/6, 1/6)$
  $P(x|z = \text{loaded}) = (1/10, 1/10, 1/10, 1/10, 1/10, 5/10)$

Example: Dishonest Casino



Conditional independence

- Given the state (=which dice), the outcomes are independent.
- If we do not know the current state, observations are dependent!
- For example: If we observe sequence of sixes, we are more likely to be in state "loaded" than "fair", which increases the probability of the next observation being a six.

HMM: Estimation Problems

Filtering problem

- **Given:** Model and observations, i.e. :
  1. Transition matrix $\mathbf{q}$ and emission distribution $P(\,.\,|z)$.
  2. Observed sequence $x_{1:N} = (x_1, \ldots, x_N)$.
- **Estimate:** Probability of each hidden variable, i.e. $Q(Z_n = k|x_{1:n})$

Variant: **Smoothing problem**, in which we estimate $Q(Z_n = k|x_{1:N})$ instead.

Decoding problem

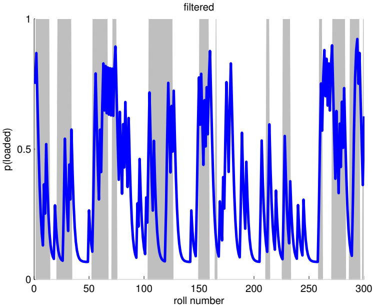
- **Given:** Model ($\mathbf{q}$ and $P(\,.\,|z)$) and observed sequence $x_{1:N}$.
- **Estimate:** Maximum likelihood estimates $\hat{z}_{1:N} = (\hat{z}_1, \ldots, \hat{z}_N)$ of hidden states.

Learning problem

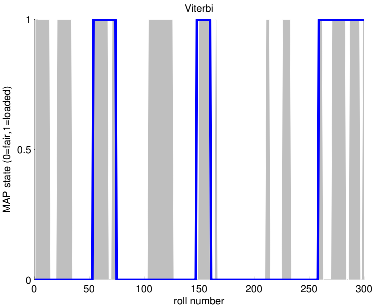
- **Given:** Observed sequence $x_{1:N}$.
- **Estimate:** Model (i.e. $\mathbf{q}$ and $P(\,.\,|z)$).

Examples

Before we look at the details, here are examples for the dishonest casino.



Filtering result.



Decoding result.

Gray bars: Loaded dice used.
Blue: Probability $P(Z_n = \text{loaded}|x_{1:N})$

Gray bars: Loaded dice used.
Blue: Most probable state Z_n.

Probabilities of Hidden States

The first estimation problem we consider is to estimate the probabilities $Q(z_n|x_{1:n})$.

Idea

We could use Bayes' equation (recall: $P(a|b) = \frac{P(b|a)P(a)}{P(b)}$) to write:

$$Q(k|x_n) = \frac{P(x_n|k)Q(Z_n = k)}{\sum_{j=1}^{K} P(x_n|k)Q(Z_n = k)} \,.$$

Since we know the Markov chain $(Q_{\text{init}}, \mathbf{q})$, we can compute Q, and the emission probabilities $P(x_n|k)$ are given.

Filtering

The drawback of the solution above is that it throws away all information about the past. We get a better estimate of Z_n by taking $x_1, \ldots, x_{n-1}$ into account. Reducing the uncertainty in Z_n using $x_1, \ldots, x_{n-1}$ is called **filtering**.

Filtering

Filtering problem

Our task is to estimate the probabilities $Q(z_n|x_{1:n})$. Since the sequence has length n and each Z_i can take K possible values, this is a $N \times K$-matrix $\hat{Q}$, with entries

$$\hat{Q}_{nk} := Q(Z_n = k|x_{1:n}) \ .$$

Decomposition using Bayes' equation

We can use Bayes' equation (recall: $P(a|b) = \frac{P(b|a)P(a)}{P(b)}$) to write:

$$Q(z_n|x_{1:n}) = Q(z_n|x_n, x_{1:(n-1)}) = \underbrace{\frac{\overbrace{P(x_n|z_n, x_{1:(n-1)})}^{\text{This is the emission probability } P(x_n|z_n) \text{ (conditional independence!)}}\overbrace{Q(z_n|x_{1:(n-1)})}^{\text{This is the crucial term}}}{\sum_{z_n=1}^{K} P(x_n|z_n, x_{1:(n-1)})Q(z_n|x_{1:(n-1)})}}_{\text{Normalization}}$$

Filtering

Reduction to previous step

The crucial idea is that we can use the results computed for step $n-1$ to compute those for step n:

$$Q(Z_n = k | x_{1:(n-1)}) = \sum_{l=1}^{K} \underbrace{Q(Z_n = k | Z_{n-1} = l)}_{= \, q_{lk} \text{ (transition matrix)}} \underbrace{Q(Z_{n-1} = l | x_{1:(n-1)})}_{= \, \hat{Q}_{(n-1)l}}$$

Summary

In short, we can compute the numerator in the Bayes equation as

$$a_{nk} := P(x_n | z_n) \sum_{l=1}^{K} q_{lk} \hat{Q}_{(n-1)l} \ .$$

The normalization term is

$$\sum_{z_n=1}^{K} \Big(P(x_n | z_n) \sum_{l=1}^{K} q_{lk} \hat{Q}_{(n-1)l} \Big) = \sum_{j=1}^{K} a_{nj} \ .$$

Filtering

Solution to the filtering problem: The forward algorithm

Given is a sequence $(x_1, \ldots, x_N)$.

For $n = 1, \ldots, N$, compute

$$a_{nk} := P(x_n|z_n) \sum_{l=1}^{K} q_{lk} \hat{Q}_{(n-1)l} \ ,$$

and

$$\hat{Q}_{nk} = \frac{a_{nk}}{\sum_{j=1}^{K} a_{nj}} \ .$$

This method is called the **forward algorithm**.

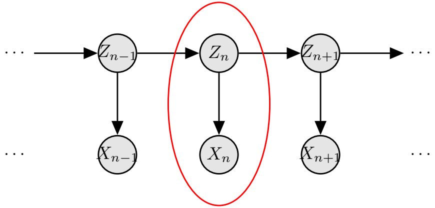
HMMs and Mixture Models

Parametric emission model

We usually define the emission probabilities $P(x_n|z_n)$ using a parametric model $P(x|\theta)$ (e.g. a multinomial or Gaussian model). Then

$$P(x_n|Z_n = k) := P(x_n|\theta_k) \ ,$$

i.e. the emission distribution of each state k is defined by a parameter value θ_k.



Relation to mixture models

If we just consider a *single* pair (Z_n, X_n), this defines a finite mixture with K clusters:

$$\pi(x_n) = \sum_{k=1}^{K} c_k P(x_n|\theta_k) = \sum_{k=1}^{K} Q(Z_n = k) P(x_n|\theta_k)$$

EM for HMMs

Recall: EM for mixtures

E-step	M-step
Soft assignments $\mathbb{E}[M_{ik}] = \mathsf{Pr}(m_i = k)$	cluster weights c_k component parameters θ_k

EM for HMMs

Recall: EM for mixtures

E-step	M-step
Soft assignments $\mathbb{E}[M_{ik}] = \text{Pr}(m_i = k)$	cluster weights c_k component parameters θ_k

HMM case

- For mixtures, $\text{Pr}\{m_i = k\} = c_k$. In HMMs, the analogous probability $\text{Pr}\{Z_n = k\}$ is determined by the transition probabilities.
- The analogue of the soft assignments a_{ik} computed for mixtures are state probabilities

$$b_{nk} = Q(Z_n = k|\theta, x_{1:N}) \; .$$

- Additionally, we have to estimate the transition matrix $\mathbf{q}$ of the Markov chain.

EM for HMMs

EM for HMMs

E-step	M-step
Transition probabilities q_{kj}	component parameters θ_k
State probabilities b_{nk}	

HMM case

- For mixtures, $\Pr\{m_i = k\} = c_k$. In HMMs, the analogous probability $\Pr\{Z_n = k\}$ is determined by the transition probabilities.
- The analogue of the soft assignments a_{ik} computed for mixtures are state probabilities

$$b_{nk} = Q(Z_n = k|\theta, x_{1:N}) \ .$$

- Additionally, we have to estimate the transition matrix $\mathbf{q}$ of the Markov chain.

EM for HMMs

M-step

The M-step works exactly as for mixture models. E.g. for Gaussian emission distributions with parameters μ_k and σ_k^2,

State probabilities substituted for assignment probabilities

$$\mu_k = \frac{\sum_{n=1}^N b_{nk} x_n}{\sum_{n=1}^N b_{nk}} \qquad \text{and} \qquad \sigma_k^2 = \frac{\sum_{n=1}^N b_{nk}(x_n - \mu_k)^2}{\sum_{n=1}^N b_{nk}}$$

E-step

- Computing the state probabilities is a filtering problem:

  $$b_{nk}^{\text{new}} = Q(Z_n = k | \theta^{\text{old}}, x_{1:n}) \; .$$

  The forward-backward algorithm assumes the emission probabilities are known, so we use the emission parameters θ^{old} computed during the previous M-step.

- Estimating the transition probabilities is essentially a filtering-type problem for *pairs* of states and can also be solved recursively, but we will skip the details since the equations are quite lengthy.

Application: Speech Recognition

Problem

Given speech in form of a sound signal, determine the words that have been spoken.

Method

- Words are broken down into small sound units (called *phonemes*). The states in the HMM represent phonemes.
- The incoming sound signal is transformed into a sequence of vectors (feature extraction). Each vector x_n is indexed by a time step n.
- The sequence $x_{1:N}$ of feature vectors is the observed data in the HMM.

Phoneme Models

Phoneme

A **phoneme** is defined as the smallest unit of sound in a language that distinguishes between distinct meanings. English uses about 50 phonemes.

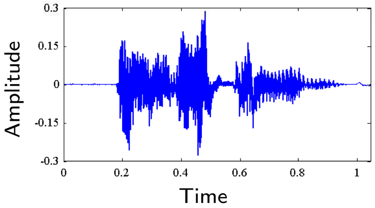
Example

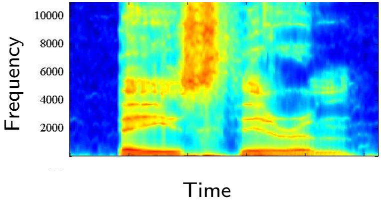
Zero	Z IH R OW	Six	S IH K S
One	W AH N	Seven	S EH V AX N
Two	T UW	Eight	EY T
Three	TH R IY	Nine	N AY N
Four	F OW R	Oh	OW
Five	F AY V		

Subphonemes

Phonemes can be further broken down into subphonemes. The standard in speech processing is to represent a phoneme by three subphonemes ("triphons").

Preprocessing Speech

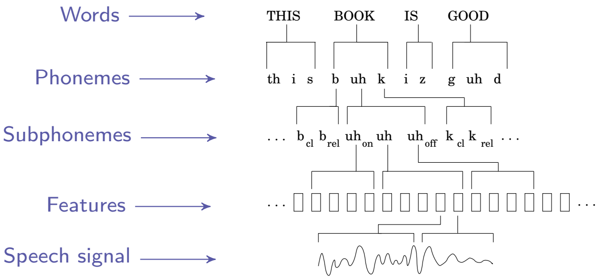




Feature extraction

- A speech signal is measured as amplitude over time.
- The signal is typically transformed into various types of features, including (windowed) Fourier- or cosine-transforms and so-called "cepstral features".
- Each of these transforms is a scalar function of time. All function values for the different transforms at time t are collected in a vector, which is the feature vector (at time t).

Layers in Phoneme Models



HMM speech recognition

- Training: The HMM parameters (emission parameters and transition probabilities) are estimated from data, often using both supervised and unsupervised techniques.
- Recognition: Given a language signal (= observation sequence $x_{1:N}$, estimate the corresponding sequence of subphonemes (= states $z_{1:N}$). This is a decoding problem.

Speaker Adaptation

Factory model

Training requires a lot of data; software is typically shipped with a model trained on a large corpus (i.e. the HMM parameters are set to "factory settings").

The adaptation problem

- The factory model represents an average speaker. Recognition rates can be improved drastically by adapting to the specific speaker using the software.
- Before using the software, the user is presented with a few sentences and asked to read them out, which provides labelled training data.

Speaker adaptation

- Transition probabilities are properties of the language. Differences between speakers (pronounciation) are reflected by the emission parameters θ_k.
- Emission probabilities in speech are typically multi-dimensional Gaussians, so we have to adapt means and covariance matrices.
- The arguably most widely used method is **maximum likelihood linear regression** (**MLLR**), which uses a regression technique to make small changes to the covariance matrices.

Further Reading

More details on HMMs

If you feel enthusiastic, the following books provide more background:

- David Barber's "Bayesian reasoning and machine learning" (available online; see class homepage).
- Chris Bishop's "Pattern recognition and machine learning".
- Many books on speech, e.g. Rabiner's classic "Fundamentals of speech recognition".

HTK

If you would like to try out speech recognition software, have a look at the HTK (**H**MM **T**ool**k**it) package, which is the de-facto standard in speech research. HTK implements both HMMs for recognition and routines for feature extraction.