02471 Machine Learning for Signal Processing

Week 10: State-space models – the Hidden Markov Model

This exercise is based on S. Theodoridis: Machine Learning, A Bayesian and Optimization Perspective 2nd edition, section(s) 16.4–16.5. The objective of this exercise is to get better understanding of the Hidden Markov Model (HMM), both how to conduct inference and how to learn parameters. It will also use the EM algorithm from last week.

A complete derivation and implementation of the learning algorithm is beyond one exercise, so we will carry out the general principles and then use library functions for the training of the HMM. It should be stressed that the library functions we are using are learning parameters using the *Baum–Welch* algorithm, and we will derive the first parts of that algorithm as well.

Overview

The exercise have the following structure:

10.1 will demonstrate how to compute probabilities for the HMM where the model parameters are known. We will derive the first part (the forward pass) of the efficient *forward-backward* algorithm (also called the Sum-Product algorithm for the general case).

10.2 will formulate the likelihood for HMM and derive the EM update rules.

10.3 will train a simple HMM model on simulated data and investigate the stability of the EM parameter estimation.

10.4 will present a naive and simplified example using COVID data, where we use HMM to estimate the epidemic state based on the number of observed hospitalizations. *Disclaimer: these results should in no way be seen as a realistic epidemic estimation, but only serve as a toy example.*

The exercises can be carried out in any order.

Notation

- The book is unfortunately not consistent with the notation of random variables in the section describing HMMs (section 16.5). E.g, in sec 16.5, the book writes $P(x = 1)$, but in section 2.3, the (correct) notation $P(\mathrm{x} = 1)$ is used. The exercise uses the books notation. For the purpose of this exercise, do not be too concerned about this unfortunate notation ambiguity, the important aspect is the understanding of the model and derivations.

Code

The code can be found in the .m and .py files named in the same way as exercises, ie. the code for exercise 10.x.y is in the file 10_x_y.m (or .py).

For coding exercises that requires implementation we will usually write `complete this line` where the implementing should be done.

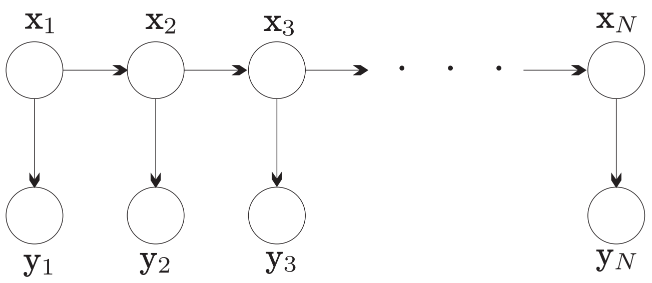
Solutions

The solution is provided for all derivation exercises, and often hints are provided at the end of the document. If you get stuck, take a look at the hints, and if you are still stuck, take a look in the solution to see the approach being taken. Then try to do it on your own.

Solutions are also provided for some coding exercises. If you get stuck, take a look at the solution, and then try to implement it on your own.

The Hidden Markov model

It is very important, for this exercise, to keep the graphical model for the HMM in mind, which is illustrated as:



Here, $\mathbf{x}_i$ is a discrete one-dimensional state variable with K possible values. There is, in general, no assumption on distribution of $\mathbf{y}_i$, but for this exercise we assume that $\mathbf{y}_i$ is a one-dimensional discrete variable.

10.1 Probabilities in HMM

This exercise is based on section 16.5.1, so be sure to read that before carrying out the exercise.

We will work with a 2-state model, with 4 different conditions. This scenario can mimic a number of real-life situations, e.g. gene sequencing (is a gene expressed or not), bio-markers, (is there a condition present or not) etc, but to simplify the calculations and the model, we assume that the observation variable y_n is one-dimensional, and only use a few states and conditions.

For discrete random variables, the distributions can simply be described as a matrix of appropriate dimension (see sec 2.2.2), so e.g $P(x_1 = \text{state } k) = P_k$, and then $P_k \in [0,1]^K$, $\sum_{k=1}^{K} P_k = 1$.

Assume the following distributions for the entirety of this exercise.

$$P_k = \begin{bmatrix} 0.7 \\ 0.3 \end{bmatrix}$$

$$P_{ij} = \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix}$$

$$P(y|k) = \begin{bmatrix} 0.2 & 0.6 & 0.1 & 0.1 \\ 0.1 & 0.2 & 0.5 & 0.2 \end{bmatrix}$$

Exercise 10.1.1

Use the sum-rule to find an expression for $P(y_1)$ and then calculate $P(y_1 = 2)$.

Exercise 10.1.2

Inspect the code for this exercise and complete the code that implements the calculation of $P(y_1)$

Exercise 10.1.3

Use the sum-rule to show that an expression for $P(y_1, y_2)$ is

$$P(y_1, y_2) = \sum_{i=1}^{K} \sum_{j=1}^{K} P(y_2|x_2 = j) P(y_1|x_1 = i) P(x_2 = j|x_1 = i) P(x_1 = i)$$

and then calculate $P(y_1 = 2, y_2 = 3)$.

Exercise 10.1.4

Inspect the code for this exercise and complete the code that implements the calculation of $P(y_1, y_2)$. What is the number of arithmetic operations (multiplications and additions) taken?

Exercise 10.1.5

The number of operations that is needed in the naive implementation is infeasible, so we will derive a more efficient expression. We'll introduce the more compact notation

$$P(y_1, y_2, \cdots, y_n, x_n) := P(y_{[1:n]}, x_n)$$
$$\sum_{i=1}^{K} P(y_1|x_1 = i) P(x_1 = i) := \sum_{x_i} P(y_1|x_1) P(x_1)$$

Using this notation, we can write

$$P(y_2, y_1) = \sum_{x_2} P(y_1, y_2, x_2)$$

Show that

$$P(y_1, y_2, x_2) = P(y_2|x_2) \sum_{x_1} P(x_2|x_1) P(y_1, x_1)$$

Exercise 10.1.6

Define $\alpha(x_n) := P(y_{[1:n]}, x_n)$ that is, the joint distribution over observations up until time n and the latent variable at time n (i.e $\alpha(x_{n-1}) := P(y_{[1:n-1]}, x_{n-1})$ and so on).

Show that we get the recursive formula:

$$P(y_{[1:n]}, x_n) = P(y_n|x_n) \sum_{x_{n-1}} P(x_n|x_{n-1}) \alpha(x_{n-1})$$

Exercise 10.1.7

We can exploit these result further using Bayes formula. Show that

$$P(x_n|y_{[1:n]}) = \frac{\alpha(x_n)}{P(y_{[1:n]})}$$
$$P(y_{[1:n]}) = \sum_{x_n} \alpha(x_n)$$

Additionally, identify the number of operations required to calculate $P(y_{[1:n]})$ using this scheme.

This is an important result, since this formula calculations the probability of being in a specific state, given all observations up until time n.

Exercise 10.1.8

Inspect the code for this exercise and complete the code that implements the calculation of $P(x_n|y_{[1:n]})$.

10.2 HMM model formulation and EM updates

This exercise is based on section 16.5.2, so be sure to read that before carrying out the exercise.

As a reminder, the EM algorithm consist of the following steps:

1. Specification of the complete log likelihood, $\ln p(\mathcal{X}, \mathcal{X}^l)$ (the model).
2. Derive $\mathcal{Q}(\boldsymbol{\xi}, \boldsymbol{\xi}^{(j)}) = \mathbb{E}\big[\ln p(\mathcal{X}, \mathcal{X}^l; \boldsymbol{\xi})^{(j)})\big]$.
3. Maximize $\mathcal{Q}(\boldsymbol{\xi}, \boldsymbol{\xi}^{(j)})$ in order to get $\boldsymbol{\xi}^{(j+1)}$.

where $\mathcal{X}$ denotes the set of observations, $\mathcal{X}^l$ denotes the set of latent random variables, and $\boldsymbol{\xi}$ is a vector of distribution parameters.

For HMM, we have $\mathcal{X} = Y$, $\mathcal{X}^l = X$, and to follow the same notation as the book, we set $\boldsymbol{\xi} = \Theta$. As a first step, we need to specify $p(X, Y)$.

Keep the graphical model of the HMM in mind when doing this exercise.

Exercise 10.2.1

We encode the state variable using using one-hot encoding (or one-of-K), and create a vector $\boldsymbol{x}_n \in \{0,\, 1\}^K$ for the n'th timestep, where $x_{n,k} = 1$ when the system is in state k and all other elements of the vector are 0.

Demonstrate, using an example, that this notation means that we can write

$$P(\boldsymbol{x}_1) = \prod_{k=1}^{K} P_k^{x_{1,k}}$$

and using the structure of the graph of the HMM model, that the likelihood becomes

$$
\begin{aligned}
p(Y, X) &= P(\boldsymbol{x}_1)p(\boldsymbol{y}_1|\boldsymbol{x}_1)\prod_{n=2}^{N} P(\boldsymbol{x}_n|\boldsymbol{x}_{n-1})p(\boldsymbol{y}_n|\boldsymbol{x}_n) \\
P(\boldsymbol{x}_n|\boldsymbol{x}_{n-1}) &= \prod_{i=1}^{K}\prod_{j=1}^{K} P_{ij}^{x_{n-1,j}x_{n,i}} \\
p(\boldsymbol{y}_n|\boldsymbol{x}_n) &= \prod_{k=1}^{K}(p(\boldsymbol{y}_n|k;\boldsymbol{\theta}_k))^{x_{n,k}}
\end{aligned}
$$

Exercise 10.2.2

Use the previous result to obtain the following expression for $\mathcal{Q}(\Theta, \Theta^{(j)})$

$$
\begin{aligned}
\mathcal{Q}(\Theta, \Theta^{(j)}) &= \mathbb{E}[\ln p(X, Y; \Theta)] \\
&= \sum_{k=1}^{K}\mathbb{E}[\mathrm{x}_{1,k}]\ln P_k + \sum_{n=2}^{N}\sum_{i=1}^{K}\sum_{j=1}^{K}\mathbb{E}[\mathrm{x}_{n-1,j}\mathrm{x}_{n,i}]\ln P_{ij} \\
&\quad + \sum_{n=1}^{N}\sum_{k=1}^{K}\mathbb{E}[\mathrm{x}_{n,k}]\ln p(y_n \mid k; \boldsymbol{\theta}_k)
\end{aligned}
$$

Exercise 10.2.3

Let us define two helper functions

$$
\begin{aligned}
\gamma(\boldsymbol{x}_n) &:= P(\boldsymbol{x}_n|Y) \\
\xi(\boldsymbol{x}_{n-1}, \boldsymbol{x}_n) &:= P(\boldsymbol{x}_{n-1}, \boldsymbol{x}_n|Y)
\end{aligned}
$$

Use these helper functions to obtain the final expression for the Expectation step:

$$
\begin{aligned}
\mathcal{Q}(\Theta, \Theta^{(j)}) &= \sum_{k=1}^{K}\gamma\big(x_{1,k} = 1; \Theta^{(j)}\big)\ln P_k \\
&\quad + \sum_{n=2}^{N}\sum_{i=1}^{K}\sum_{j=1}^{K}\xi\big(x_{n-1,j} = 1, x_{n,i} = 1; \Theta^{(j)}\big)\ln P_{ij} \\
&\quad + \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma\big(x_{n,k} = 1; \Theta^{(j)}\big)\ln p(y_n \mid k; \boldsymbol{\theta}_k)
\end{aligned}
$$

Exercise 10.2.4

In the Maximization step, we maximize $\mathcal{Q}(\Theta, \Theta^{(j)})$ with respect to the parameters, which in this case are P_k, P_{ij} and $\boldsymbol{\theta}_k$. We will only derive an update for P_{ij}, as the derivation for P_k is similar, and the derivation for $\boldsymbol{\theta}_k$ is not something that depends on the HMM as such, but the nature of $\mathbf{y}_i$ as this models our observations. Common choices are the Multinomial distribution (discrete), Gaussian distribution, or a Gaussian mixture model.

Use Lagrange multiplier to specify the Lagrangian, and then show that the update rule for P_{ij} is

$$P_{ij}^{(j+1)} = \frac{\sum_{n=2}^{N} \xi\big(x_{n-1,j} = 1, x_{n,i} = 1; \Theta^{(j)}\big)}{\sum_{n=2}^{N} \sum_{k=1}^{K} \xi(x_{n-1,j} = 1, x_{n,k} = 1; \Theta^{(j)})}$$

Remark

We are still missing some ingredients to have a fully fledged learning algorithm. We need to

1. Derive $P(\boldsymbol{x}_n|Y) := \gamma(x_n)$ instead of $P(\boldsymbol{x}_n|Y_{[1:n]})$ (which was derived in ex 10.1).
2. Derive $P(\boldsymbol{x}_{n-1}, \boldsymbol{x}_n|Y) := \xi(\boldsymbol{x}_{n-1}, \boldsymbol{x}_n)$.
3. Derive $P_k^{(j+1)}$.
4. Derive $\boldsymbol{\theta}_k^{(j+1)}$.

Step number 1–2 is solved in a similar fashion as the forward pass (and requires the backward pass). Step 3 is derived using Lagrange multipliers in the same manner as deriving $P_{ij}^{(t+1)}$. And finally, step 4 requires us to specify the emission distribution, and the derivation is similar as last week.

As mention in the introduction, to carry out *all* of these steps are quite lengthy both in terms of coding and derivations, so the practical examples in this exercise are using built-in library functions.

10.3 Convergence of parameter estimation using EM

In this exercise we will create a "teacher" model and use that to generate a sequence of length N. Then we will train M models to investigate the convergence properties of the EM algorithm. Inspect and run the code for this exercise. What steps is carried out? Make a simplified flow-chart of the operations being taken. Why does the EM fail to find good models (sometimes) for small sequences even though we initialize with correct parameters?

Modify the code to create learning curves and error bars when the EM is randomly initialized and initialized with the teacher model parameters.

10.4 HMM for COVID-19

Inspect and run the code for this exercise. What steps does the code carry out? Make a simplified flow-chart of the operations being taken. Make sure to identify the preprocessing steps, the number of hidden steps and the type of state emission distribution used. Modify the code to run multiple models and use the best. Experiment with other data representations (or data sources) to find better results.

HINTS

Exercise 10.1.1

$P(y_1 = 2) = 0.17$

Exercise 10.1.3

$P(y_1 = 2, y_2 = 3) = 0.084$

Exercise 10.2.3

Since $\mathrm{x}_{1,k}$ is a binary variable, the expectation is equal to the probability $\mathbb{E}[\mathrm{x}_{1,k}]$.

Exercise 10.2.4

The constraint we have is

$$\sum_{i=1} P_{ij} = 1 \quad \forall j$$