Note on Spectral learning for Hidden Markov Models (HMMs)

Wen Sun

1 Problem Definition

HMM Consider a discrete hidden Markov model with m latent states and n observations where we assume $n \geq m$. We will denote s as the latent state and o as the observation. We denote $T \in \mathbb{R}^{m \times m}$ as the Markovian transition in latent states, i.e., $T[i, j] = P(s' = i | s = j)$ – the probability of the next latent state s' being i given the current latent state s being j. We denote $O \in \mathbb{R}^{n \times m}$ as the observation matrix, where $O[i, j] = P(o = i | s = j)$ denotes the probability of observing i given latent state being j. Finally, we define the initial latent state distribution as $\mu \in \Delta(m)$. Note that μ, T, O together fully specify the HMM.

Learning setting We are given a dataset of i.i.d trajectories (consisting of observations only) with length 3, i.e., $\mathcal{D} = \{o_1^i, o_2^i, o_3^i\}_{i=1}^N$, where each trajectory is sampled as follows: $s_1 \sim \mu, o_1 \sim O(\cdot|s_1), s_2 \sim T(\cdot|s_1), o_2 \sim O(\cdot|s_2), s_3 \sim T(\cdot|s_2), o_3 \sim O(\cdot|s_3)$.

Our goal is to learn to compute the following quantities: $P(o_1, \ldots, o_t)$, for any observation trajectory $o_1, o_2, \ldots, o_t$ for $t \in \mathbb{Z}^+$,

Challenge The unknown parameters of the HMM are T and O (for simplicity, let us assume μ is given). To learn them, we can use Maximum Likelihood estimation, i.e., searching for T and O which maximizes the likelihood of training dataset $\mathcal{D}$. This is straightforward to do. However, due to the existence of latent states, the log-likelihood objective is non-convex. While in practice, one can use a gradient ascent style approach or a classic algorithm called Expectation-Maximization (EM), these algorithms cannot guarantee to find global optimality.

Is there an algorithm that can find the global optimal solution?

Notations For a given matrix $A \in \mathbb{R}^{n \times m}$, we will write $A[i, :] \in \mathbb{R}^{1 \times m}$ as the i-th row of the matrix, and $A[:, i] \in \mathbb{R}^n$ as the i-th column. We denote span(A) as the column span of the matrix A.

2 Spectral Learning for HMMs

In this algorithm, we explain the spectral learning algorithm from Hsu et al. (2012).

2.1 Observable operator

Let us define observable operator $A_o \in \mathbb{R}^{m \times m}$, for all $o \in [n]$.

$$A_o = T \text{diag}\left(O[o|1], \ldots, O[o|m]\right)$$

In other words, $A_o[i, j] := P(s' = i, o | s = j) = O(o|s = j)T(s' = i|s = j)$, i.e., the probability of seeing observation o at the current step, and transiting to i in the next step, conditioned on being at latent state j at the current step.

The following lemma shows that we can write $P(o_1, \ldots, o_t)$ using these observable operators.

Lemma 1. *For any $t \in \mathbb{Z}^+$, and any observations $o_1, \ldots, o_t$, we have:*

$$P(o_1, \ldots, o_t) = \mathbf{1}^\top A_{o_t} A_{o_{t-1}} \ldots A_{o_1} \mu$$

where we denote $\mathbf{1}$ as a m-dim vector consisting of all ones.

Proof. Let us consider $P(s_2, o_1)$. It is not hard to verify that

$$P(s_2, o_1) = \sum_{s_1} P(s_1) O(o_1|s_2) T(s_2|s_1) = \mu^\top A_{o_1}[s_2, :]^\top.$$

We can continue to compute $P(s_3, o_2, o_1)$, which is:

$$\begin{aligned} P(s_3, o_2, o_1) &= \sum_{s_2} P(s_3, s_2, o_2, o_1) = \sum_{s_2} P(s_2, o_1) P(s_3, o_2|s_2, o_1) \\ &= \sum_{s_2} P(s_2, o_1) T(s_3|s_2) O(o_2|s_2) \\ &= \sum_{s_2} \left(\mu^\top A_{o_1}[s_2, :]^\top \right) A_{o_2}[s_3, s_2] = \mu^\top A_{o_1}^\top (A_{o_2}[s_3, \cdot])^\top. \end{aligned}$$

Repeat the above process, we can have:

$$P(o_1, o_2, \ldots, o_t, s_{t+1}) = \mu^\top A_{o_1}^\top \ldots (A_{o_t}[s_{t+1}, :])^\top.$$

Note that $P(o_1, \ldots, o_t) = \sum_{i \in [m]} P(o_1, \ldots, o_t, s_{t+1} = i) \cdot 1$, we must have:

$$P(o_1, \ldots, o_t) = \mu^\top A_{o_1}^\top \ldots A_{o_t}^\top \mathbf{1}.$$

This concludes the proof. □

2.2 Learning equivalent representations of observable operators

So far, the observable operators A_o still consist of unknown transition T and O which both depend on latent states. What we are going to do next is to find a full-rank matrix $C \in \mathbb{R}^{m \times m}$ (thus invertible), and re-write $P(o_1, \ldots, o_t)$ as:

$$P(o_1, o_2, \ldots, o_t) = \mathbf{1}^\top C^{-1} C A_{o_t} C^{-1} C A_{o_{t-1}} C^{-1} C \ldots C^{-1} C A_{o_1} C^{-1} C \mu.$$

We denote

$$B_o = C A_o C^{-1},\ b_\infty^\top = \mathbf{1}^\top C^{-1},\ b_1 = C\mu,$$

and we will show that with a properly designed C, we can directly learn B_o, b_∞, b_1 from the dataset that only contains observations.

We first state the following key assumption.

Assumption 2. *Assume $n \geq m$, i.e., more observations than latent states, and T and O are full column rank.*

Remark 3. *The condition where $n \geq m$ and O are full rank is typically called 1-observability. This roughly means that knowing the distribution of the next observation is equivalent to knowing the current latent state distribution. More formally, denote $p \in \Delta(m)$ as the distribution over the current latent states. Then the distribution $q \in \Delta(n)$ of the current observation can be written as $q = Op$. Since O is full rank, we have $O^\dagger O = I$, which means that $p = O^\dagger q$. Thus it means that even though we do not know the distribution q since latent states are never observed, if we know the distribution of the observation q (which can be estimated accurately from observable data), this is equivalent to knowing p up to a linear transformation.*

We will define the following observable vectors/matrices:

$$P_1[i] = P(o_1 = i); P_{2,1}[i,j] = P(o_2 = i, o_1 = j), P_{3,o,1}[i,j] = P(o_3 = i, o_2 = o, o_1 = j), \forall o \in [m]. \quad (1)$$

Note that these matrices can be directly estimated using the dataset $\mathcal{D}$, for example, $P(o_2 = i, o_1 = j)$ can be estimated by $\sum_{i=1}^{N} \mathbf{1}(o_2^i = i, o_1^i = j)/N$. We know that when $N \to \infty$, these estimations converge to the exact probabilities (finite sample error is also easily computable like we did for quantifying the transition error in learning tabular MDPs). The key message is that these quantities do not depend on latent states.

Denote the SVD of $P_{2,1}$ as $P_{2,1} = U\Sigma V^\top$ where $U \in \mathbb{R}^{m\times n}$, $\Sigma \in \mathbb{R}^{n\times n}$ is a diagonal matrix consisting of singular values. We will set $C = U^\top O$. First, we show that $U^\top O$ is full rank and thus invertible.

Lemma 4. *Let U be the left singular vectors of $P_{2,1}$. We have $U^\top O$ is full rank.*

Proof. Using the property of the HMM, we can first write $P_{2,1} = OT\mathrm{diag}(\mu)O^\top$. This implies that $\mathrm{span}(P_{2,1}) \subseteq \mathrm{span}(O)$, which implies that $\mathrm{span}(U) \subseteq \mathrm{span}(O)$. To verify that $P_{2,1} = OT\mathrm{diag}(\mu)O^\top$, we can see that for any pair of i, j, we have $P_{2,1}[i,j] = \sum_{s_2\in[n]} O[i|s_2] \sum_{s_1\in[n]} T[s_2|s_1]\mu(s_1)O[j|s_1]$.

On the other hand, the assumption that $O \in \mathbb{R}^{n\times m}$ is full column rank implies that O^T is also full column rank, with the fact that $n \geq m$, we have $(O^\top)(O^\top)^+ = I$, which means that $P_{2,1}(O^\top)^+\mathrm{diag}(\mu)^{-1}T^{-1} = O$. This means that $\mathrm{span}(O) \subseteq \mathrm{span}(P_{2,1}) = \mathrm{span}(U)$.

Together, we have shown that $\mathrm{span}(U) = \mathrm{span}(O)$. Now to show that $U^\top O$ is full rank, take any vector $x \in \mathbb{R}^n$ where $x \neq 0$. We have $Ox \in \mathrm{span}(O) = \mathrm{span}(U)$, which means that there exists $\alpha \in \mathbb{R}^n$ with $\alpha \neq 0$, such that $Ox = U\alpha$. Note that then $U^\top Ox = U^\top U\alpha = \alpha \neq 0$. This means that $U^\top O$ does not have null space, which implies that $U^\top O$ is full rank. □

Now we show that b_1, b_∞, B_o can be represented by observable quantities $P_1, P_{2,1}$ and $P_{3,o,1}, \forall o \in [n]$.

Lemma 5. *We have $b_1 = U^\top P_1$.*

Proof. Recall that $b_1 = C\mu$. Plug in $C = U^\top O$, we have:

$$b_1 = U^\top(O\mu) = U^\top P_1,$$

where we use the fact that $P_1[i] = P(o_1 = i) = \sum_{s_1} O[o_1 = i|s_1]\mu(s_1) = O[i,:]\mu$, which means that $P_1 = O\mu$. □

Lemma 6. *We have $B_o = U^\top P_{3,o,1}(U^\top P_{2,1})^+$, $\forall o \in [n]$.*

Proof. Recall that $B_o = CA_oC^{-1}$. Using the definition of $C = U^\top O$, we have:

$$B_o = U^\top OA_o(U^\top O)^{-1}$$

Note that for $P_{3,o,1}$ we first have:

$$P_{3,o,1}[i,j] = \sum_{s_3,s_2,s_1} O(i|s_3)T(s_3|s_2)O(o|s_2)T(s_2|s_1)\mu(s_1)O(j|s_1)$$

which means that $P_{3,o,1} = OA_oT\mathrm{diag}(\mu)O$. Now we have:

$$\begin{aligned} P_{3,o,1} &= OA_oT\mathrm{diag}(\mu)O^\top = OA_o(U^\top O)^{-1}(U^\top O)T\mathrm{diag}(\mu)O^\top = OA_o(U^\top O)^{-1}U^\top(OT\mathrm{diag}(\mu)O^\top) \\ &= OA_o(U^\top O)^{-1}U^\top P_{2,1}. \end{aligned}$$

Since $U^\top P_{2,1} = \Sigma V^\top$ where recall $P_{2,1} = U^\top \Sigma V^\top$, we have $U^\top P_{2,1}$ being full row rank. Thus $(U^\top P_{2,1})(U^\top P_{2,1})^+$ is equal to identity. Thus, we have:

$$P_{3,o,1}(U^\top P_{2,1})^+ = OA_o(U^\top O)^{-1}.$$

This means that $B_o = U^\top OA_o(U^\top O)^{-1} = U^\top P_{3,o,1}(U^\top P_{2,1})^+$. □

Finally, we show that b_∞ can also be computed by observable quantities.

Lemma 7. *We have* $b_\infty^\top = P_1^\top (U^\top P_{2,1})^+$.

Proof. Recall that $b_\infty^\top = \mathbf{1}^\top C^{-1}$. Using the definition of C, we have:

$$b_\infty^\top = \mathbf{1}^\top (U^\top O)^{-1} = \mathbf{1}^\top (U^\top O)^{-1}(U^\top P_{2,1})(U^\top P_{2,1})^+$$

Note that $P_{2,1} = OT\mathrm{diag}(\mu)O^\top$, we have:

$$\mathbf{1}^\top (U^\top O)^{-1} U^\top P_{2,1} = \mathbf{1}^\top (U^\top O)^{-1} U^\top OT\mathrm{diag}(\mu)O^\top = \mathbf{1}^\top T\mathrm{diag}(\mu)O^\top = P_1^\top.$$

Thus, we have:

$$b_\infty^\top = P_1^\top (U^\top P_{2,1})^+.$$

□

2.3 Put things together: training algorithm and inference procedure

Let us summarize the algorithm below. The **training algorithm** consists of 3 steps below.

- estimate $P_1, P_{2,1}$, and $P_{3,o,1}$ as follows:

$$\hat{P}_1[i] = \frac{\sum_{n=1}^N \mathbf{1}(o_1^n = i)}{N}, \hat{P}_{2,1}[i,j] = \frac{\sum_{n=1}^N \mathbf{1}(o_1^n = j, o_2^n = i)}{N}, \hat{P}_{3,o,1}[i,j] = \frac{\sum_{n=1}^N \mathbf{1}(o_3^n = i, o_2^n = o, o_1^n = j)}{\sum_{n=1}^N \mathbf{1}(o_2^n = o)}$$

- Perform SVD on $\hat{P}_{2,1}$, i.e., $\hat{P}_{2,1} = \hat{U}\hat{\Sigma}\hat{V}^\top$;
- Estimate b_1, b_∞, B_o as follows:

$$\hat{b}_1 = \hat{U}^\top \hat{P}_1;\ \hat{b}_\infty^\top = \hat{P}_1^\top (\hat{U}^\top \hat{P}_{2,1})^+;\ \hat{B}_o = \hat{U}^\top \hat{P}_{3,o,1}(\hat{U}^\top \hat{P}_{2,1})^+, \forall o \in [n].$$

This algorithm is called spectral learning mostly because in step 2 above we performed SVD.

For **inference**, given any arbitary trajectory of observations $o_1, \ldots, o_t$, we can estimate its probability as:

$$\hat{P}(o_1, \ldots, o_t) = \hat{b}_\infty^\top \hat{B}_{o_t} \ldots \hat{B}_{o_1} \hat{b}_1.$$

Note that the above whole procedure does not reason about latent state at all, and when $N \to \infty$, we have $\hat{P}_1, \hat{P}_{2,1}, \hat{P}_{3,o,1}$ converge to its corresponding population level matrices.

3 Further reading: spectral learning with kernels and PAC learning in POMDPs

The above algorithm only covers the discrete tabular setting. There was a significant amount of effort in extending the spectral learning algorithm to HMMs w/a large number of states and observations (or even continuous state and observation space). A representative work is from Song et al. (2010) which embeds transition and observation operators into Reproducing Kernel Hilbert Spaces (RKHS).

HMMs are passive sequential models, i.e., there are no controls/actions and there is no reward to optimize. The partially observable MDP (POMDP) model extends HMM to control setting by adding actions and reward functions. Designing PAC algorithms for POMDPs turns out to be much more complicated than that for MDPs. The works from Uehara et al. (2022) and Zhan et al. (2022) are two of the most recent works on designing algorithms for PAC learning for POMDPs.

References

Daniel Hsu, Sham M Kakade, and Tong Zhang. A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences*, 78(5):1460–1480, 2012.

Le Song, Byron Boots, Sajid Siddiqi, Geoffrey J Gordon, and Alex Smola. Hilbert space embeddings of hidden markov models. *ICML*, 2010.

Masatoshi Uehara, Ayush Sekhari, Jason D Lee, Nathan Kallus, and Wen Sun. Provably efficient reinforcement learning in partially observable dynamical systems. *NeurIPS*, 2022.

Wenhao Zhan, Masatoshi Uehara, Wen Sun, and Jason D Lee. Pac reinforcement learning for predictive state representations. *ICLR*, 2022.