Hidden Markov Models

So far, we assumed i.i.d data as input. However, we have to deal with sequential data in real world e.g. time-series, characters in a sequence, where there is some temporal correlation between data points.

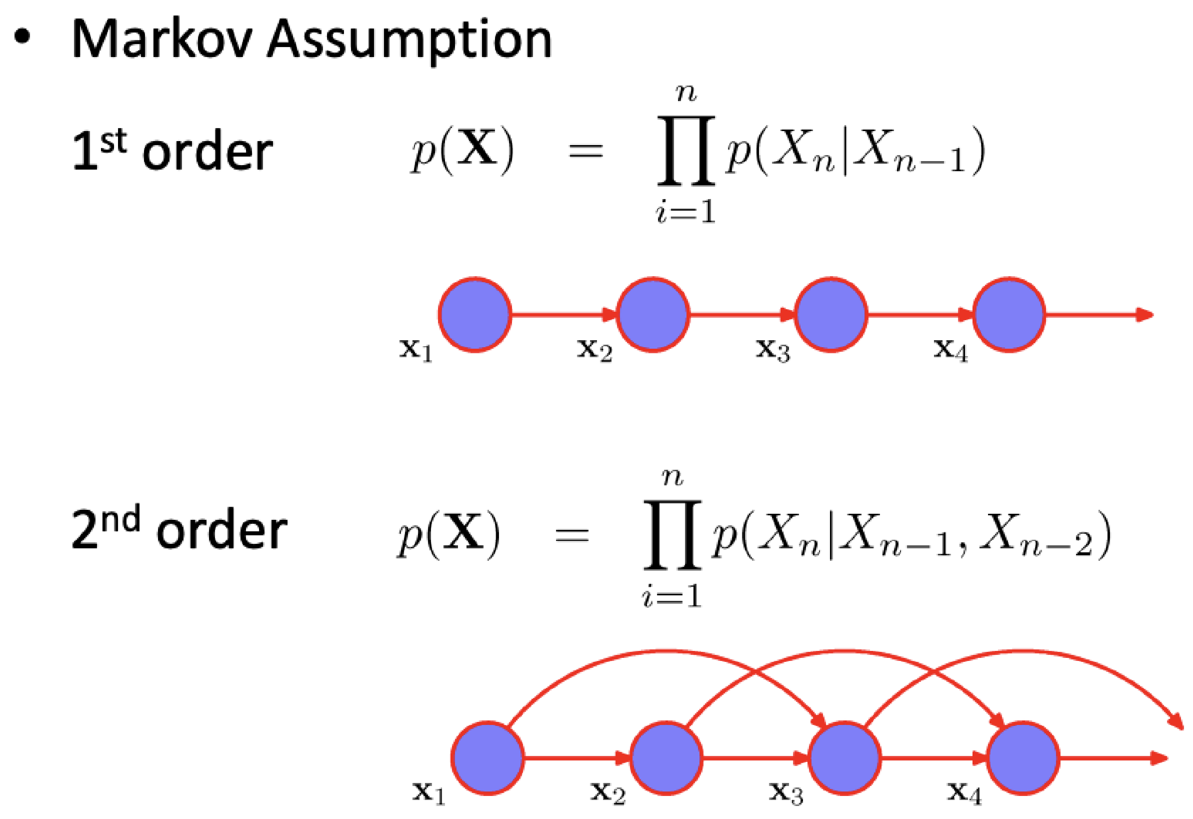
HMMs are probabilisitic generative models that can model sequential data. So, it's possible to synthesize new data by sampling HMMs.
Remember given an input sequence $X = (X_1, X_2, \ldots, X_n)$, joint distribution is given by:

$$p(X) = p(X_1, X_2, \ldots, X_n) = p(X_1)p(X_2|X_1)p(X_3|X_2, X_1)\ldots p(X_n|X_{n-1}, \ldots, X_1)$$
$$= \prod_{i=1}^{n} p(X_n|X_{n-1}, \ldots, X_1)$$

m^{th} order Markov Assumption simplifies this into the following:

$$p(X) = \prod_{i=1}^{n} p(X_n|X_{n-1}, \ldots X_{n-m})$$

> Current observation only depends on past m observations



- Markov Assumption

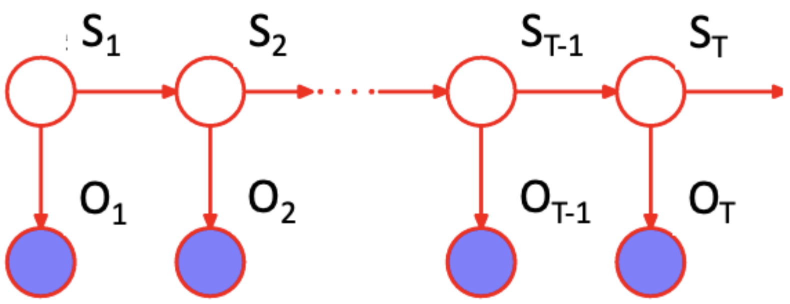
1st order $$p(\mathbf{X}) = \prod_{i=1}^{n} p(X_n|X_{n-1})$$

2nd order $$p(\mathbf{X}) = \prod_{i=1}^{n} p(X_n|X_{n-1}, X_{n-2})$$

The number of parameters to be estimated grows exponentially with n. $n-1$ order Markov model has $O(d^n)$ complexity.

Hidden Markov Models

How can we avoid such high order dependencies between observations?
HMM is a family of distributions that characterize sequential data with few parameters but are not limited by strong Markov assumptions:



Observation space	$O_t \in \{y_1, y_2, \ldots, y_K\}$
Hidden states	$S_t \in \{1, \ldots, S\}$

The joint distribution of HMM is given by:

$$p(S_1, \ldots, S_T, O_1, \ldots, O_T) = \prod_{t=1}^{T} p(O_t|S_t) \prod_{t=1}^{T} p(S_t|S_{t-1})$$

Assumptions:

1. 1^{st} order Markov assumption on hidden states S_t, $t = 1, \ldots, T$
2. Each observation depends on a single hidden state:

$$p(O_1, \ldots, O_T|S_1, \ldots, S_T) = \prod_{i=1}^{T} p(O_t|S_t)$$

Overall, the factorization of the joint distribution is the following:

$$p(S, O) = p(S_1) \prod_{t=2}^{T} p(S_t|S_{t-1}) \prod_{t=1}^{T} p(O_t|S_t)$$

HMM Model parameters are given by the following tuple:

$$\lambda = [A, B, \pi]$$

where

- $\pi = p(S_1 = i)$ are the initial probabilities,
- $p(S_t = j|S_{t-1} = i) = A_{ij}$ are the transition probabilities
- $p(O_t = y|S_t = i) = B_i^y$ are the emission probabilities

Three Problems in HMMs

Evaluation (Likelihood)

Given HMM parameters and observation sequence $O_1, O_2, \ldots O_t$, find the probability of observed sequence:
$p(O_1, O_2, \ldots, O_t|\lambda)$
For evaluation of this probability, one can use **FORWARD ALGORITHM** or **BACKWARD ALGORITHM** , which is different from FORWARD-BACKWARD ALGORITHM in the decoding step below.
$\alpha_t(s_i) = P(O_1, O_2, \ldots O_t, S_t = i|\lambda)$ where $O_1, \ldots, O_t$ is the sequence of observations up to time t.
1. Initialization:
$\alpha_1(s_i) = \pi_i b_{s_i}(O_i)$ for $1 \leq i \leq N$, where π_{s_i} is the initial probability of state i and $b_{s_i}(O_i)$ is the probability of observing O_1 given state s_i.
2. Induction:

$$\alpha_{t+1}(s_j) = (\sum_{i=1}^{N} \alpha_t(s_i) a_{ij}) b_j(O_{t+1})$$

for $1 \leq j \leq N$ and $1 \leq t \leq T$ where a_{ij} is the transition probability from state i to j and $b_j(O_{t+1})$ is the probability of observing O_{t+1} given state j.
3.Termination:
The probability of the entire observation sequence O given the model is computed by summing the final alpha values over all states:
$P(O|lambda) = \sum_{i=1}^{N} \alpha_T(s_i)$
Just sum all alphas for each state at the last timestep.

Decoding

Given HMM parameters and observation sequence $O_1, O_2, \ldots O_t$, find $argmaxp(S|O)$, the most probable sequence of hidden states based on current observations.

Note that finding the most probable sequence of latent spaces is not the same as that of finding set of states that are individually probable.
Forward-Backward Algorithm
It can be used to find the most likely state for any point in time.
Viterbi Decoding Algorithm
On the other hand, Viterbi decoding can be used to find the most likely sequence of states
1. Initialization:
Path probability value δ and backpointer value ψ:

$$\delta_1(s_i) = \pi_i b_{s_i}(O_1)$$

$$\psi_1(i) = 0$$

2.Recursion:
For $t = 2$ to T (length of observation):

$$\delta_t(s_j) = max_{1\leq i\leq N}(\delta_{t-1}(i)a_{ij})b_j(O_t)$$

$$\psi_t(j) = argmax_{1\leq i\leq N}(\delta_{t-1}(i)a_{ij})$$

3.Termination:
Find the final state S_T that maximizes the probability at the last time step:

$$P^* = max_{1\leq i\leq N}\delta_T(i)$$

$$S_T^* = argmax_{1\leq i\leq N}\delta_T(i)$$

4. Backtracking:
Input is the next state. Output is the current state:

$$S_t^* = \psi_{t+1}(S_{t+1}^*)$$

for $t = T-1, \dots, 1$

Learning

Given HMM with **unknown** parameters and observation sequence $O_1, O_2, \dots O_t$, find a new HMM explains the observations at least as well, or possibly better, i.e., such that $p(S|\lambda_{new}) \geq p(S|\lambda_{old})$

For this problem, one can use the following 3 algorithms:

1. MLE
2. Viterbi training (not the same as Viterbi decoding)
3. **Baum Welch** = EM on forward-backward algorithm
   We can estimate only locally maxima of such probability (remember EM is local optimization method).
   Unlike MLE in i.i.d setting, the likelihood doesn't factorize easily.

1.Initialization
Start with initial guesses for HMM parameters λ
2.Expectation:
a) Forward algorithm: compute forward probabilities α
b) Backward algorithm: compute backward probabilities β
c) Compute $\epsilon_t(i,j)$: the probability of transitioning from state i at time t to state j at time $t+1$ given the entire observation sequence x

$$\xi_t(i,j) = \frac{\alpha_t(S_i)a_{S_iS_j}b_{S_j}(x_{t+1})\beta_{t+1}(S_j)}{\sum_{z_t=S_1}^{S_N}\sum_{z_{t+1}=S_1}^{S_N}\alpha_t(z_t)a_{z_tz_{t+1}}b_{S_j}(x_{t+1})\beta_{t+1}(z_{t+1})}$$

d) Compute $\gamma_t(i)$: the probability of being in state i at time t given the observation sequence. Summing over j:

$$\gamma_t(i) = \sum_{j=1}^{N} \epsilon_t(i,j)$$

So, in the end, we have:
Expected number of transitions $S_i -> S_j$ at all times:
$\sum_{t=1}^{T} \epsilon_t(i,j)$
Expected number of times S_i is visited: $\sum_{t=1}^{T} \gamma_t(i)$
3.Maximization:
Reestimate everything;
a) $\pi_i = \gamma_1(i)$, $t = 1$ because we are estimating the **initial** state probabilities
b)

$$a_{ij} = \frac{\sum_{t=1}^{T} \epsilon_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

expected number of transitions s_i to s_j over expected number of times being in state s_i
c) $b_{s_i}(o_v)$ expected number of times being in s_i **AND** observing o_v over expected number of times being in s_i.

Remarks

- An HMM is called **ergodic** if every state can be reached from every other state. In this case the transition probability matrix is full rank and has non-zero values inside.
- In practice, left-right HMMs are much more commonly used. Left-right HMMs only allow forward edges and self-loops, i.e., $a_{S_iS_j} = 0$ if $i > j$

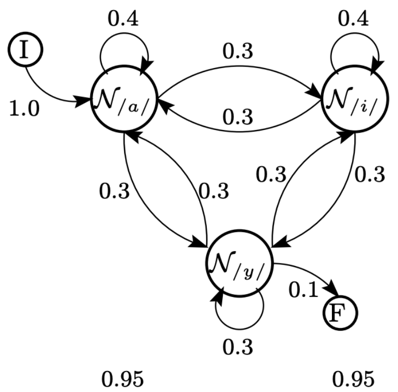
An example for drawing HMM from a transition matrix A:

Emission probabilities **Transition matrix** **Sketch of the model**

HMM1:

- state 1: initial state
- state 2: Gaussian $\mathcal{N}_{/a/}$
- state 3: Gaussian $\mathcal{N}_{/i/}$
- state 4: Gaussian $\mathcal{N}_{/y/}$
- state 5: final state

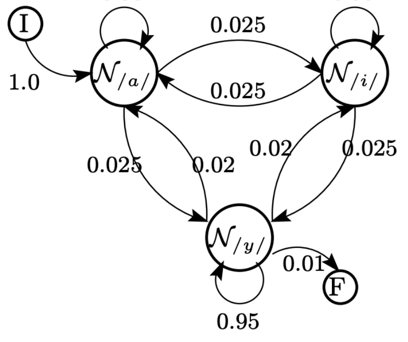
$$\begin{bmatrix} 0.0 & \mathbf{1.0} & 0.0 & 0.0 & 0.0 \\ 0.0 & \mathbf{0.4} & \mathbf{0.3} & \mathbf{0.3} & 0.0 \\ 0.0 & \mathbf{0.3} & \mathbf{0.4} & \mathbf{0.3} & 0.0 \\ 0.0 & \mathbf{0.3} & \mathbf{0.3} & \mathbf{0.3} & \mathbf{0.1} \\ 0.0 & 0.0 & 0.0 & 0.0 & \mathbf{1.0} \end{bmatrix}$$



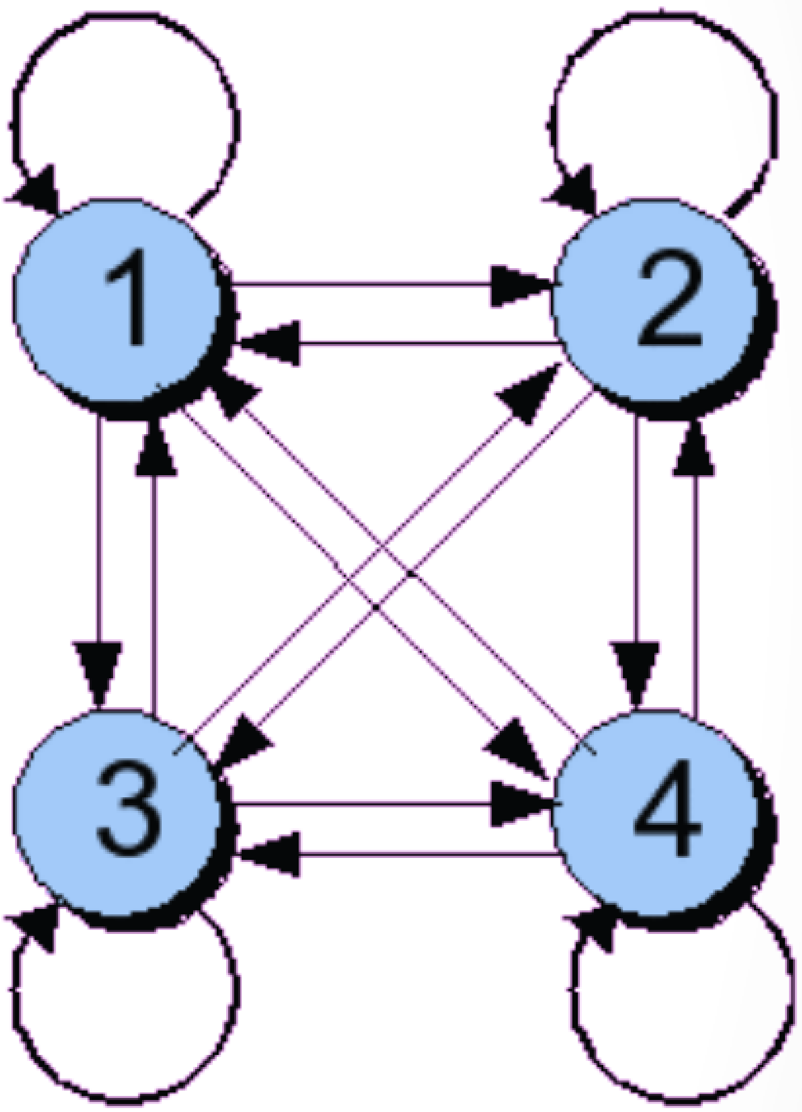
HMM2:

- state 1: initial state
- state 2: Gaussian $\mathcal{N}_{/a/}$
- state 3: Gaussian $\mathcal{N}_{/i/}$
- state 4: Gaussian $\mathcal{N}_{/y/}$
- state 5: final state

$$\begin{bmatrix} 0.0 & \mathbf{1.0} & 0.0 & 0.0 & 0.0 \\ 0.0 & \mathbf{0.95} & \mathbf{0.025} & \mathbf{0.025} & 0.0 \\ 0.0 & \mathbf{0.025} & \mathbf{0.95} & \mathbf{0.025} & 0.0 \\ 0.0 & \mathbf{0.02} & \mathbf{0.02} & \mathbf{0.95} & \mathbf{0.01} \\ 0.0 & 0.0 & 0.0 & 0.0 & \mathbf{1.0} \end{bmatrix}$$



Now, an example for ergodic HMM:



Ergodic =
fully-connected

Model Selection for HMM

Just have a separate validation dataset and try out different initialization to find the best model. If the dataset is limited, use cross-validation

To avoid bad local optima, better initialization would help (different number of states, different alphabet size)

Super-fast HMM
HMM with N states $S_1, S_2, \ldots, S_N$ but the starting probabilities for $S_3, S_4, \ldots, S_N$ are zero and non-zero for only S_1, S_2 states.

Since the transitions starting from state S_3 is limited only on one direction(no residual connections and no backward connections), overall time complexity of the HMM would reduce to $O(NT)$ instead of usual $O(N^2T)$

Disadvantages include:

1. Model expressiveness: real world processes often have more complex dependencies and state transitions, may suffer from underfitting and oversimplification
2. Limited flexibility to data. If the input is smaller and bigger, this model would not be able to adapt
   Small vs Large HMM
   Train one large HMM with different paths for different words to be able to share the representations of identical phonemes in the states. To find out which word was spoken, one can use Viterbi to reconstruct the path through the HMM.
   Benefits:
3. Reduced redundancy by sharing the states that represent identical phonemes, the overall number of parameters are significantly reduced
4. Shared learning
   Disadvantages:
5. Complexity increases as the vocabulary size increases
6. Decoding ambiguity: similar phoneme sequences might correspond to different words.
7. Scalability: adding new words to the vocabulary might require retraining the entire HMM or adjusting the structure, which can be expensive.