Hidden Markov Model

Haosheng Zhou

August 2022

Hidden Markov Model

Since HMM is a very important topic in the field of applied statistics, we provide a complete and rigorous introduction here to some most important problems and algorithms about HMM.

Basic Settings

Assume that we have a finite-time time-homogeneous Markov Chain $X_1, ..., X_T$ with initial distribution π and the transition matrix $A_{N\times N}$. Each random variable X_i takes value in the finite state space $S = \{s_1, ..., s_N\}$ and we adopt the convention that $A_{ij} = \mathbb{P}(X_{k+1} = s_j|X_k = s_i)$. Unfortunately, the realizations of these X_i are hidden and can never be observed.

Instead, we can observe the emissions of these hidden random variables, i.e. $Y_1, ..., Y_T$, taking values in the finite set $V = \{v_1, ..., v_M\}$. Here the connections between X_i and Y_j are characterized by the emission matrix $B_{N\times M}$ with $B_{ij} = \mathbb{P}(Y_k = v_j|X_k = s_i)$ and the assumption that $\mathbb{P}(Y_k = v_j|X_k = s_i)$ does not depend on k. In other words, the i-th row of B is a discrete probability distribution that can be seen as a fixed machine installed inside state s_i with this distribution, each time X_k arrives at state s_i at time k and hopes to generate an emission, it starts this machine to decide which element in V to draw as the value of Y_k. The crucial point here is that we assume the generation of Y_k to depend directly only on X_k and different $Y_1, ..., Y_T$ are generated independently. (**NOTE**: this does not mean that Y_k is independent of $X_1, ..., X_{k-1}, X_{k+1}, ..., X_T$ but means that conditional on X_k they are independent, i.e. such dependency holds by going through X_k.

To give the entire setting for an HMM, it's not hard to figure out that in order to fix the distribution of the Markov chain X_i, an initial distribution π is needed together with the transition matrix A and the finite state space S. For the emission part, emission matrix B and the set of all emission values V are required. As a result, the HMM can be determined by a tuple (S, V, π, A, B), where $\lambda = (\pi, A, B)$ is called the **parameter** of such HMM.

For simplicity, x_i appears in the expression of probability as an abbreviation of the event $\{X_i = x_i\}$, y_i as an abbreviation of the event $\{Y_i = y_i\}$, $\mathbb{P}(s_j|s_i)$ as an abbreviation of $\mathbb{P}(X_{k+1} = s_j|X_k = s_i) = A_{ij}$, $\mathbb{P}(v_j|s_i)$ as an abbreviation of $\mathbb{P}(Y_k = v_j|X_k = s_i) = B_{ij}$, and $B_j(y_k)$ as an abbreviation of $\mathbb{P}(Y_k = y_k|X_k = s_j)$ if not particularly specified.

The main assumptions of HMM are restated here as: (i): Markov property of process X_i, $\mathbb{P}(X_i|X_1, ..., X_{i-1}) = \mathbb{P}(X_i|X_{i-1})$, i.e. given the present state, past and future are independent. (ii): Emission independence of Y_i, i.e. $\mathbb{P}(Y_i|X_1, ..., X_T, Y_1, ..., Y_{i-1}, Y_{i+1}, ..., Y_T) = \mathbb{P}(Y_i|X_i)$, i.e. given X_i, Y_i is independent of anything else. Assume that the emissions are observed until time $T < \infty$, different kinds of questions can be asked about the HMM.

Likelihood

Likelihood computation happens when we know all about an HMM, i.e. the parameter λ is known, and also observe an emission sequence $y_1, ..., y_T \in V$. The goal is to compute how likely it is for us to observe this certain emission sequence.

Assume that the underlying hidden state sequence is $x_1, ..., x_T \in S$ and apply the law of total probability:

$$\begin{aligned}
\mathbb{P}(y_1, ..., y_T) &= \sum_{(x_1,...,x_T)} \mathbb{P}(x_1, ..., x_T)\, \mathbb{P}(y_1, ..., y_T | x_1, ..., x_T) && (1)\\
&= \sum_{(x_1,...,x_T)} \mathbb{P}(x_1, ..., x_T)\, \mathbb{P}(y_T | x_1, ..., x_T, y_1, ..., y_{T-1})\, \mathbb{P}(y_1, ..., y_{T-1} | x_1, ..., x_T) && (2)\\
&= \sum_{(x_1,...,x_T)} \mathbb{P}(x_1, ..., x_T)\, \mathbb{P}(y_T | x_T)\, \mathbb{P}(y_1, ..., y_{T-1} | x_1, ..., x_T) && (3)\\
&= \cdots && (4)\\
&= \sum_{(x_1,...,x_T)} \mathbb{P}(x_1, ..., x_T)\, \mathbb{P}(y_T | x_T) ... \mathbb{P}(y_1 | x_1) && (5)\\
&= \sum_{(x_1,...,x_T)} \pi(x_1) \mathbb{P}(x_2 | x_1) ... \mathbb{P}(x_T | x_{T-1})\, \mathbb{P}(y_T | x_T) ... \mathbb{P}(y_1 | x_1) && (6)
\end{aligned}$$

However, these formulas have little practical values, since we would have to consider all possible values taken by $(x_1, ..., x_T)$, N^T possibilities. For each possible value, the product has to be computed, so likelihood computation takes overall time $O(TN^T)$.

By noticing that there are actually a lot of repeated calculations happening, we can store the calculated partial results for repeated future use. This provides us with a dynamic programming (DP) scheme for likelihood calculations.

$D[t, j] \overset{def}{=} \mathbb{P}(X_t = s_j, y_1, ..., y_t)$, i.e. the probability of being in the j-th state at time t and seeing all t emissions so far. Apply the law of total probability w.r.t. the state at the previous time:

$$\begin{aligned}
&\mathbb{P}(X_t = s_j, y_1, ..., y_t) && (7)\\
=&\sum_{i=1}^{N} \mathbb{P}(X_{t-1} = s_i)\, \mathbb{P}(X_t = s_j, y_1, ..., y_t | X_{t-1} = s_i) && (8)\\
=&\sum_{i=1}^{N} \frac{D[t-1, i]}{\mathbb{P}(y_1, ..., y_{t-1} | X_{t-1} = s_i)} \mathbb{P}(y_t | X_t = s_j, X_{t-1} = s_i, y_1, ..., y_{t-1})\, \mathbb{P}(X_t = s_j, y_1, ..., y_{t-1} | X_{t-1} = s_i) && (9)\\
=&\sum_{i=1}^{N} D[t-1, i] \mathbb{P}(y_t | X_t = s_j)\, \mathbb{P}(X_t = s_j | y_1, ..., y_{t-1}, X_{t-1} = s_i) && (10)\\
=&\sum_{i=1}^{N} D[t-1, i] \mathbb{P}(y_t | X_t = s_j)\, \mathbb{P}(X_t = s_j | X_{t-1} = s_i) && (11)\\
& && (12)
\end{aligned}$$

Note that here $\mathbb{P}(X_t = s_j | X_{t-1} = s_i, y_1, ..., y_{t-1}) = \mathbb{P}(X_t = s_j | X_{t-1} = s_i)$ since y_{t-1} is fixed conditional on $X_{t-1} = s_i$ and X is Markov. In the expression above, we see the self-repeated structure with the third term on the right to be the entry in the transition matrix A and the second term on the right to be the entry in the emission matrix B. The complete recurrence relationship goes like:

$$D[t,j] = \sum_{i=1}^{N} D[t-1,i] B_j(y_t) A_{ij} \tag{13}$$

$$D[1,j] = \pi_j B_j(y_1) \tag{14}$$

gives the rule for dynamic programming. This is known as **the forward algorithm** for computing likelihood for emission sequences.

After the whole matrix D is updated, our final likelihood should be

$$\begin{aligned} &\mathbb{P}(y_1, ..., y_T) && (15)\\ &= \sum_{i=1}^{N} \mathbb{P}(X_T = s_i, y_1, ..., y_T) && (16)\\ &= \sum_{i=1}^{N} D[T,i] && (17) \end{aligned}$$

This DP reduces the time complexity to $O(TN^2)$, efficient and feasible.

Decoding

The second task for HMM that appears frequently is the decoding task, i.e. given the parameter λ of the HMM and an emission sequence $y_1, ..., y_T$, we would like to derive a state sequence $x_1, ..., x_T$ such that it's the most likely sequence of hidden states to appear.

This problem is not so easy as it seems to be. One might want to say: why don't we just take x_i to be the hidden state such that y_i is the most likely to be generated since we know the full structure of the emission sequence. Such approach is incorrect because it ignores the transition matrix. Doing so may cause the problem that the state sequence $x_1, ..., x_T$ generated is very unlikely to appear (for example, state x_1 may even have no probability transiting to state x_2).

Another more reasonable and intuitive approach would be: first compute the best value for x_1, i.e. the state such that y_1 is the most likely to appear. Afterwards, compute the best value for x_2 by taking the estimate of x_1 and the value of y_2 into consideration at the same time. In detail, choose x_2 such that

$$\begin{aligned} x_2^* &= argmax_{x_2} \mathbb{P}(x_2, y_2 | x_1) && (18)\\ &= argmax_{x_2} \mathbb{P}(y_2 | x_1, x_2) \mathbb{P}(x_2 | x_1) && (19)\\ &= argmax_{x_2} \mathbb{P}(y_2 | x_2) \mathbb{P}(x_2 | x_1) && (20) \end{aligned}$$

Such algorithm adopts the greedy strategy but falls in the pit of being too short-sighted. The error in the estimation of the most possible past hidden states accumulates and will still give incorrect results.

The **Viterbi algorithm** is then developed to fully solve this problem. Greedy strategy is still adopted, but in a more subtle way such that short-sighted strategy can give globally optimal result. Our goal is to maximize the likelihood of the observed emissions together with the underlying hidden states. In other words, we hope to do the following optimization:

$$x_1^*, ..., x_T^* = argmax_{x_1,...,x_T} \mathbb{P}(x_1, ..., x_T, y_1, ..., y_T) \tag{21}$$

$$= argmax_{x_1,...,x_T} \pi(x_1)\mathbb{P}(x_2|x_1) ... \mathbb{P}(x_T|x_{T-1}) \mathbb{P}(y_T|x_T) ... \mathbb{P}(y_1|x_1) \tag{22}$$

REMARK: it's important to distinguish $\mathbb{P}(x_1, ..., x_T, y_1, ..., y_T)$, $\mathbb{P}(y_1, ..., y_T)$, $\mathbb{P}(y_1, ..., y_T|x_1, ..., x_T)$ and to understand why the first probability is selected as the objective function for optimization here. Since the underlying hidden states are unknown, $\mathbb{P}(y_1, ..., y_T|x_1, ..., x_T)$ should not be a reasonable objective function (optimizing this probability just gives the first incorrect algorithm we discussed at the beginning of this decoding section, completely ignoring the effects of the transition matrix). Since we are hoping to derive the best x_i sequence but not the best y_i sequence, $\mathbb{P}(y_1, ..., y_T)$ is also not a reasonable choice. On the other hand, $\mathbb{P}(x_1, ..., x_T, y_1, ..., y_T)$ is a good choice because it not only considers the connection within x_i transitions but also considers the emissions from x_i to y_i.

Directly solving out this optimization problem is not at all realistic. As a result, DP is constructed once more. The trick here is to tear the maximum into two maximums w.r.t. the value of the state in the previous time.

$$V[t, j] \overset{def}{=} \max_{x_1,...,x_{t-1}} \mathbb{P}(x_1, ..., x_{t-1}, y_1, ..., y_t, X_t = s_j) \tag{23}$$

$$= \max_{x_1,...,x_{t-2}} \max_{i=1,...,N} \mathbb{P}(x_1, ..., x_{t-2}, y_1, ..., y_t, X_{t-1} = s_i, X_t = s_j) \tag{24}$$

$$= \max_{x_1,...,x_{t-2}} \max_{i=1,...,N} \mathbb{P}(y_t|X_t = s_j) \mathbb{P}(x_1, ..., x_{t-2}, y_1, ..., y_{t-1}, X_{t-1} = s_i, X_t = s_j) \tag{25}$$

$$= \max_{x_1,...,x_{t-2}} \max_{i=1,...,N} \mathbb{P}(y_t|X_t = s_j) \mathbb{P}(X_t = s_j|X_{t-1} = s_i) \mathbb{P}(x_1, ..., x_{t-2}, y_1, ..., y_{t-1}, X_{t-1} = s_i) \tag{26}$$

$$= \max_{i=1,...,N} B_j(y_t)A_{ij} \max_{x_1,...,x_{t-2}} \mathbb{P}(x_1, ..., x_{t-2}, y_1, ..., y_{t-1}, X_{t-1} = s_i) \tag{27}$$

$$= \max_{i=1,...,N} B_j(y_t)A_{ij}V[t-1, i] \tag{28}$$

For the boundary conditions, it's pretty clear that:

$$V[1, j] = \pi_j B_j(y_1) \tag{29}$$

And the maximum possible likelihood of such a state sequence is:

$$\max_{x_1,...,x_T} \mathbb{P}(x_1, ..., x_T, y_1, ..., y_T) \tag{30}$$

$$= \max_{x_1,...,x_{T-1}} \max_{j=1,2,...,N} \mathbb{P}(x_1, ..., x_{T-1}, y_1, ..., y_T, X_T = s_j) \tag{31}$$

$$= \max_{j=1,2,...,N} V[T, j] \tag{32}$$

As a result, after updating all entries of V, look through the part in V corresponding to time T and search for the largest number, such number is just $\max_{x_1,...,x_T} \mathbb{P}(x_1, ..., x_T, y_1, ..., y_T)$, i.e. the largest possible likelihood achieved among all state sequences for the given emission sequence.

To get the Viterbi path, i.e. the best state sequence $x_1, ..., x_T$, save a backtracing pointer when doing this DP. In each step computing $V[t, j] = \max_{i=1,...,N} B_j(y_t)A_{ij}V[t-1, i]$, figure out which i attains the maximum and record it in a backtracing matrix. Such i indicates that the best possible step to take in order to arrive at $V[t, j]$ is from $V[t-1, i]$, i.e. from state s_i at previous time. By following the entries in the backtracing matrix, we should manage to construct the Viterbi path. The time complexity is $O(TN^2)$.

Learning Parameters

For parameter learning, the setting changes. Now we have many observations of emission sequences as the output of an HMM, but we do not know λ. So we want to find a way to estimate the best π, A, B that match this HMM's outputs. The learning step is the most difficult problem, but also the most important step to practically build an HMM for decoding purposes.

Before talking about the classical **Baum-Welch** algorithm that solves the learning problem, let us introduce some probabilities that we will make use of in the following context. Assume for simplicity that only one single emission sequence $y_1, ..., y_T$ is observed. The first quantity is called the **forward probability**:

$$\alpha_t(j) \overset{def}{=} \mathbb{P}(X_t = s_j, y_1, ..., y_t|\lambda) \tag{33}$$

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i)A_{ij}B_j(y_t) \tag{34}$$

$$\alpha_1(j) = \pi_j B_j(y_1) \tag{35}$$

This is exactly what have already been introduced in the section of likelihood. A DP is constructed with forward probabilities to compute the likelihood of an emission sequence. The forward probability has its name for the reason that it looks at the event that we get to the j-th state at time t from a forward perspective, i.e. also knowing all the history of emissions.

NOTE: In the likelihood section, the probability is not conditioned on λ since at that time we assume λ to be known.

A slightly different version is the backward probability.

$$\beta_t(j) \overset{def}{=} \mathbb{P}(y_{t+1}, ..., y_T|X_t = s_j, \lambda) \tag{36}$$

To get the recurrence relationship similar to what we've got for forward probability:

$$\beta_t(j) = \mathbb{P}\left(y_{t+1}, ..., y_T | X_t = s_j, \lambda\right) \tag{37}$$

$$= \sum_{i=1}^{N} \mathbb{P}\left(X_{t+1} = s_i | X_t = s_j, \lambda\right) \mathbb{P}\left(y_{t+1}, ..., y_T | X_t = s_j, X_{t+1} = s_i, \lambda\right) \tag{38}$$

$$= \sum_{i=1}^{N} \mathbb{P}\left(X_{t+1} = s_i | X_t = s_j, \lambda\right) \mathbb{P}\left(y_{t+1} | y_{t+2}, ..., y_T, X_t = s_j, X_{t+1} = s_i, \lambda\right) \tag{39}$$

$$\mathbb{P}\left(y_{t+2}, ..., y_T | X_t = s_j, X_{t+1} = s_i, \lambda\right) \tag{40}$$

$$= \sum_{i=1}^{N} A_{ji} \mathbb{P}\left(y_{t+1} | X_{t+1} = s_i, \lambda\right) \mathbb{P}\left(y_{t+2}, ..., y_T | X_{t+1} = s_i, \lambda\right) \tag{41}$$

$$= \sum_{i=1}^{N} A_{ji} B_i(y_{t+1}) \beta_{t+1}(i) \tag{42}$$

The boundary case here is a little bit special:

$$\beta_T(j) = 1 \tag{43}$$

When the value of X_T is known, the value of Y_T is then fixed with no randomness.

The forward and backward probabilities are two sides of a same coin and can both be used to construct DP for likelihood computation. The forward algorithm introduced in the previous section makes use of forward probability, but it's also easy to write out the **backward algorithm** that makes use of backward probability to compute the likelihood. The only difference is that forward probability considers all history appearing while backward probability considers all futures appearing.

Now we are all set with the tools necessary for the BW algorithm, let's then state the main thought of this algorithm. Think in a heuristic way: empirically, what would one do in order to build up $\lambda = (\pi, A, B)$ based on a single observed emission sequence? An easy answer would be to use empirical frequency in the sample as a substitute for the probability. For example, if we want to estimate π_j, we just divide the number of emission sequences starting with state s_j by the number of all emission sequences. If we want to estimate A_{ij}, we just divide the number of appearances of state transitions from s_i to s_j by the number of appearances of all state transitions from s_i to any other state. If we want to estimate B_{jk}, we just divide the number of appearances of state s_j emitting v_k by the overall number of appearances of state s_j.

You may not believe that we are already done! The Baum-Welch algorithm does exactly these things to find out the optimal parameter λ for an HMM. From the heuristic point of view, BW is an extremely intuitive algorithm!

Baum-Welch (BW)

Let's dive in to the details of BW. First consider the simplified version, where there is **only one** observed emission sequence $y_1, ..., y_T$.

Consider the estimation of A_{ij}, the transition probability from s_i to s_j. As we have just stated above, empirically, this is just the number of appearances of state transitions from s_i to s_j divided by the number of appearances of state transitions from s_i to any other state. However, we would face the following difficulties: (i): we only have one sample, so computing such ratio as an estimate always results in large bias (ii): more importantly, the states are hidden so we do not know the exact underlying state sequence, i.e. it's actually totally impossible to do what we've said above (**NOTE**: here we definitely can't use Viterbi to do the decoding since Viterbi requires knowing all parameters λ of the HMM). Despite the fact that we can't compute such ratio directly, we know the expectation of these hidden counts based on the emission sequence observed, a good substitute.

Let's try to figure out the expression for the expectation of C, the number of appearance s of state transitions from s_i to s_j given the emission sequence $y_1, ..., y_T$. Denote $\mathbb{I}$ for indicator function:

$$C = \mathbb{I}_{X_1=s_i, X_2=s_j|\lambda, y_1,...,y_T} + ... + \mathbb{I}_{X_{T-1}=s_i, X_T=s_j|\lambda, y_1,...,y_T} \tag{44}$$

$$\mathbb{E}C = \mathbb{P}(X_1 = s_i, X_2 = s_j|\lambda, y_1, ..., y_T) + ... + \mathbb{P}(X_{T-1} = s_i, X_T = s_j|\lambda, y_1, ..., y_T) \tag{45}$$

Here the reader might have such a question: why and how are we conditioning on λ? This question is quite natural since we actually don't know the true λ and want to estimate it by the learning process. However, note that the distribution of all random variables are known if and only if λ is given, so we have to add this λ to the condition. Doesn't this contradict our goal to find the true parameters of the HMM? Actually there's a subtle way to circumvent this problem by using the **iterative methods**. The iterative methods work if we first provide a prior estimate for λ (if there's no prior knowledge, such prior estimate can also be randomly generated) and then update λ by the information contained in the sample. By adopting such method, $\mathbb{E}C$ can absolutely be computed base on an old estimation of λ. The only concern now is to find a way to calculate the following probability:

$$\xi_t(i,j) \overset{def}{=} \mathbb{P}(X_t = s_i, X_{t+1} = s_j|y_1, ..., y_T, \lambda) \tag{46}$$

$$\mathbb{E}C = \sum_{t=1}^{T-1} \xi_t(i,j) \tag{47}$$

By applying the Bayes rule for the conditional probability measure $\mathbb{P}(\cdot|\lambda)$, ξ can be computed using A, B, α, β:

$$\begin{aligned}
\xi_t(i,j) &= \mathbb{P}(X_t = s_i, X_{t+1} = s_j | y_1, ..., y_T, \lambda) && (48)\\
&\propto \mathbb{P}(X_t = s_i, X_{t+1} = s_j|\lambda)\, \mathbb{P}(y_1, ..., y_T | X_t = s_i, X_{t+1} = s_j, \lambda) && (49)\\
&= \frac{\mathbb{P}(X_t = s_i, X_{t+1} = s_j|\lambda)}{\mathbb{P}(X_t = s_i | X_{t+1} = s_j, \lambda)} \mathbb{P}(y_1, ..., y_T, X_t = s_i | X_{t+1} = s_j, \lambda) && (50)\\
&= \frac{\mathbb{P}(X_t = s_i, X_{t+1} = s_j|\lambda)}{\mathbb{P}(X_t = s_i | X_{t+1} = s_j, \lambda)} \mathbb{P}(y_{t+1} | y_1, ..., y_t, y_{t+2}, ..., y_T, X_t = s_i, X_{t+1} = s_j, \lambda) && (51)\\
&\mathbb{P}(y_1, ..., y_t, y_{t+2}, ..., y_T, X_t = s_i | X_{t+1} = s_j, \lambda) && (52)\\
&= \frac{\mathbb{P}(X_t = s_i, X_{t+1} = s_j|\lambda)}{\mathbb{P}(X_t = s_i | X_{t+1} = s_j, \lambda)} \mathbb{P}(y_{t+1} | X_{t+1} = s_j, \lambda)\, \mathbb{P}(y_1, ..., y_t, X_t = s_i | X_{t+1} = s_j, \lambda) && (53)\\
&\mathbb{P}(y_{t+2}, ..., y_T | X_{t+1} = s_j, \lambda) && (54)\\
&= B_j(y_{t+1})\beta_{t+1}(j) \frac{\mathbb{P}(y_1, ..., y_t, X_t = s_i | X_{t+1} = s_j, \lambda)\, \mathbb{P}(X_t = s_i, X_{t+1} = s_j|\lambda)}{\mathbb{P}(X_t = s_i | X_{t+1} = s_j, \lambda)} && (55)\\
&= B_j(y_{t+1})\beta_{t+1}(j) \mathbb{P}(y_1, ..., y_t, X_t = s_i, X_{t+1} = s_j|\lambda) && (56)\\
&= B_j(y_{t+1})\beta_{t+1}(j) \mathbb{P}(X_{t+1} = s_j | y_1, ..., y_t, X_t = s_i, \lambda)\, \mathbb{P}(y_1, ..., y_t, X_t = s_i|\lambda) && (57)\\
&= B_j(y_{t+1})\beta_{t+1}(j) \mathbb{P}(X_{t+1} = s_j | X_t = s_i, \lambda)\, \mathbb{P}(y_1, ..., y_t, X_t = s_i|\lambda) && (58)\\
&= B_j(y_{t+1})\beta_{t+1}(j) A_{ij}\alpha_t(i) && (59)
\end{aligned}$$

Note that $\sum_{i,j=1}^N \xi_t(i,j) = 1$, the normalization constant is computed to get:

$$\xi_t(i,j) = \frac{B_j(y_{t+1})\beta_{t+1}(j)A_{ij}\alpha_t(i)}{\sum_{i,j=1}^N B_j(y_{t+1})\beta_{t+1}(j)A_{ij}\alpha_t(i)} \tag{60}$$

Now we are half way to success, the computations afterwards can all be based on such $\xi_t(i,j)$.

As stated above, the expectation of D, the number of appearances of state transitions from s_i to any other state given the emission sequence $y_1, ..., y_T$, is also of our concern. In a similar sense:

$$\begin{aligned}
D &= \mathbb{I}_{X_1 = s_i | y_1, ..., y_T, \lambda} + ... + \mathbb{I}_{X_{T-1} = s_i | y_1, ..., y_T, \lambda} && (61)\\
\mathbb{E}D &= \mathbb{P}(X_1 = s_i | y_1, ..., y_T, \lambda) + ... + \mathbb{P}(X_{T-1} = s_i | y_1, ..., y_T, \lambda) && (62)
\end{aligned}$$

As a result, only consider:

$$\begin{aligned}
\gamma_t(i) &\overset{def}{=} \mathbb{P}(X_t = s_i | y_1, ..., y_T, \lambda) && (63)\\
\mathbb{E}D &= \sum_{t=1}^{T-1} \gamma_t(i) && (64)
\end{aligned}$$

Luckily, we don't have to do these annoying computations once more since:

$$\gamma_t(i) = \mathbb{P}\left(X_t = s_i | y_1, ..., y_T, \lambda\right) \tag{65}$$

$$= \sum_{j=1}^{N} \mathbb{P}\left(X_t = s_i, X_{t+1} = s_j | y_1, ..., y_T, \lambda\right) \tag{66}$$

$$= \sum_{j=1}^{N} \xi_t(i,j) \tag{67}$$

By knowing all values of ξ, γ is also known. Actually, ξ is the only probability we have to calculate, since the events it considers are already the finest possible events that would ever be needed (**NOTE**: the explanation for this point is in the next section).

Return to the update of A_{ij}, we immediately conclude that by replacing the unknown empirical counts of state transitions with the expectation of those empirical counts of state transitions, the updated A_{ij} is:

$$A_{ij} = \frac{\mathbb{E}C}{\mathbb{E}D} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{68}$$

The π works as the initial distribution of hidden states, empirically, π_j should be the count of emission sequences where s_j is the initial state over the count of all emission sequences. Replace these two quantities by their respective expectations, we get the updated π_j:

$$\pi_j = \gamma_1(j) \tag{69}$$

The B works as the emission matrix, empirically, B_{jk} should be the count of states that emit v_k over the count of all states. Replace these two quantities by their respective expectations, we get the updated B_{jk}:

$$B_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j) \mathbb{I}_{y_t = v_k}}{\sum_{t=1}^{T} \gamma_t(j)} \tag{70}$$

Now that we have finished stating the BW algorithm for a single observed emission sequence. The following Alg. 1 provides the whole procedure.

The extension of BW algorithm to the case where we have multiple observed emission sequences is rather simple. Assume that there are R emission sequences denoted $(y_1^1, y_2^1, ..., y_T^1), ..., (y_1^R, y_2^R, ..., y_T^R)$, here subscripts stand for time and superscripts stand for the sample index, e.g. y_t^r stands for the value of emission at time t in the r-th sample emission sequence. It's natural that each sample provides us with information on the behavior of those parameters. The general Baum-Welch algorithm is provided in Alg. 2. Actually, the structure is barely the same as that in Alg. 1, with the only difference to be that for each sample, the probabilities are computed. When updating parameters, replace the numerator and denominator with the sample average of all R updates we would get as if we are running Alg. 1 for each single observed emission sequence. In brief, the difference only lies in: (i): compute probabilities for each sequence (ii): take the average of those expressions in Alg. 1 as the best update.

Algorithm 1 Baum-Welch for a single observed emission sequence

```
Input: $y_1, ..., y_T$ as the observed emission sequence
Output: Estimated parameters $\pi, A, B$ of HMM, $\pi$ the initial distribution of hidden states, $A$ the transition matrix,
    $B$ the emission matrix
 1: Set prior estimates for $\pi, A, B$
 2: while not converge do
 3:     {Compute Forward Probability}
 4:     $\alpha_1(j) = \pi_j B_j(y_1)$
 5:     $\alpha_t(j) = \sum_{i=1}^N \alpha_{t-1}(i) A_{ij} B_j(y_t)$
 6:
 7:     {Compute Backward Probability}
 8:     $\beta_T(j) = 1$
 9:     $\beta_t(j) = \sum_{i=1}^N \beta_{t+1}(i) A_{ji} B_i(y_{t+1})$
10:
11:     {Compute $\xi_t(i,j)$}
12:     $\xi_t(i,j) = \frac{B_j(y_{t+1})\beta_{t+1}(j)A_{ij}\alpha_t(i)}{\sum_{i,j=1}^N B_j(y_{t+1})\beta_{t+1}(j)A_{ij}\alpha_t(i)}$
13:
14:     {Compute $\gamma_t(i)$}
15:     $\gamma_t(i) = \sum_{j=1}^N \xi_t(i,j)$
16:
17:     {Update the estimates for $\pi, A, B$}
18:     $\pi_j = \gamma_1(j)$
19:     $A_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$
20:     $B_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j) \mathbb{I}_{y_t = v_k}}{\sum_{t=1}^{T} \gamma_t(j)}$
21: end while
```

Algorithm 2 Baum-Welch for multiple observed emission sequence

Input: $(y_1^1, y_2^1, ..., y_T^1), ..., (y_1^R, y_2^R, ..., y_T^R)$ as R observed emission sequences

Output: Estimated parameters π, A, B of HMM, π the initial distribution of hidden states, A the transition matrix, B the emission matrix

```
1:  Set prior estimates for $\pi, A, B$
2:  while not converge do
3:      for r=1,...,R do
4:          {Compute Forward Probability}
5:          $\alpha_1^r(j) = \pi_j B_j(y_1^r)$
6:          $\alpha_t^r(j) = \sum_{i=1}^N \alpha_{t-1}^r(i) A_{ij} B_j(y_t^r)$
7:
8:          {Compute Backward Probability}
9:          $\beta_T^r(j) = 1$
10:         $\beta_t^r(j) = \sum_{i=1}^N \beta_{t+1}^r(i) A_{ji} B_i(y_{t+1}^r)$
11:
12:         {Compute $\xi_t^r(i,j)$}
13:         $\xi_t^r(i,j) = \frac{B_j(y_{t+1}^r)\beta_{t+1}^r(j)A_{ij}\alpha_t^r(i)}{\sum_{i,j=1}^N B_j(y_{t+1}^r)\beta_{t+1}^r(j)A_{ij}\alpha_t^r(i)}$
14:
15:         {Compute $\gamma_t(i)$}
16:         $\gamma_t^r(i) = \sum_{j=1}^N \xi_t^r(i,j)$
17:     end for
18:
19:     {Update the estimates for $\pi, A, B$}
20:     $\pi_j = \frac{1}{R}\sum_{r=1}^R \gamma_1^r(j)$
21:     $A_{ij} = \frac{\sum_{r=1}^R \sum_{t=1}^{T-1} \xi_t^r(i,j)}{\sum_{r=1}^R \sum_{t=1}^{T-1} \gamma_t^r(i)}$
22:     $B_{jk} = \frac{\sum_{r=1}^R \sum_{t=1}^{T} \gamma_t^r(j)\mathbb{I}_{y_t^r = v_k}}{\sum_{r=1}^R \sum_{t=1}^{T} \gamma_t^r(j)}$
23: end while
```

For the time being, we should be familiar with all details of the general Baum-Welch algorithm. However, only heuristic motivations are described in this section and nothing rigorous in mathematics for this algorithm has appeared so far. The following questions may arise: why is such update strategy the best? In what sense is such strategy the best? The next section would give us the answers by viewing BW as a special case of the EM algorithm.

Baum-Welch as a case of EM

We want to argue that the BW algorithm is the application of the expectation maximization scheme. First recall that EM scheme assumes the knowledge on an old estimate of $\hat{\pi}, \hat{A}, \hat{B}$ and hopes to find a new estimate π, A, B that maximizes the expectation of log-likelihood. Assume we have $\hat{\lambda} = (\hat{\pi}, \hat{A}, \hat{B})$ as the knowledge on the parameters we already have, i.e. an old estimate for the parameters, and $\lambda^* = (\pi^*, A^*, B^*)$ to be the best update under EM scheme. First consider the single-sample case, where we are given a single emission sequence $y_1, ..., y_T$. By EM:

$$\lambda^* = argmax_\lambda \mathbb{E}[\log p(x_1, ..., x_T, \lambda | y_1, ..., y_T) | y_1, ..., y_T, \hat{\lambda}] \tag{71}$$

NOTE: we are conditioning on $x_1, ..., x_T$ having the distribution fixed by $y_1, ..., y_T$ and the old estimate $\hat{\lambda}$. Here p stands for density/probability mass function. Write x as an abbreviation of $x_1, ..., x_T$ and y as an abbreviation of $y_1, ..., y_T$. Some standard calculations for EM tell us:

$$\lambda^* = argmax_\lambda [\log p(\lambda) + \int \log p(x, y|\lambda) \cdot p(x|y, \hat{\lambda})\, dx] \tag{72}$$

Here, assume that we have no knowledge on the prior of these parameters by taking $p(\lambda)$ as constant, i.e. the flat prior. **NOTE**: When one is solving problems in a specific field, it's natural to set non-trivial priors. Since only the *argmax* matters, the $\log p(\lambda)$ term disappears:

$$\lambda^* = argmax_\lambda \int \log p(x, y|\lambda) \cdot p(x|y, \hat{\lambda})\, dx \tag{73}$$

Everything until now is standard in EM scheme. Now plug in the probability mass functions. $p(x, y|\lambda)$ is the joint probability mass of states and emissions conditional on knowing the parameters. Luckily, it has been computed before:

$$p(x_1, ..., x_T, y_1, ..., y_T|\lambda) = \pi(x_1)\mathbb{P}\left(x_2|x_1, \lambda\right) ...\mathbb{P}\left(x_T|x_{T-1}, \lambda\right) \mathbb{P}\left(y_T|x_T, \lambda\right) ...\mathbb{P}\left(y_1|x_1, \lambda\right) \tag{74}$$

Now for the term $p(x|y, \hat{\lambda})$, it's the probability mass of the hidden states given the emission sequence and the old estimates of parameters. It seems hard to deal with, so we may keep this term for now to see whether where are other ways to deal with this later.

$$\lambda^* = argmax_\lambda \sum_{(x_1,...,x_T)} [\log \pi(x_1) + \log \mathbb{P}(x_2|x_1,\lambda) + ... + \log \mathbb{P}(x_T|x_{T-1},\lambda) + \log \mathbb{P}(y_T|x_T,\lambda) + ... + \log \mathbb{P}(y_1|x_1,\lambda)] \cdot p(x|y,\hat{\lambda}) \tag{75}$$

Let's apply a standard trick in EM, to change $(x_1, ..., x_T)$ into $(s_{l_1}, ..., s_{l_T})$ to indicate their values, with $l_1, ..., l_T$ taking values in the set $[N] = \{1, ..., N\}$.

$$\begin{aligned} \lambda^* = argmax_\lambda \sum_{l_1,...,l_T\in[N]} [\log \pi_{l_1} + \log A_{l_1 l_2} + ... + \log A_{l_{T-1} l_T} + \log B_{l_T}(y_T) + ... + \log B_{l_1}(y_1)] \quad (76)\\ \mathbb{P}\left(X_1 = s_{l_1}, ..., X_T = s_{l_T}|y_1, ..., y_T, \hat{\lambda}\right) \quad (77) \end{aligned}$$

From the expression above, we know that for the sum w.r.t. $\log\pi$ and $\log B$, only the value l_i matters, and for the sum w.r.t. $\log A$, only the value of the tuple (l_{i-1}, l_i) matters. The terms in the sum are now having a public factor $\mathbb{P}\left(X_1 = s_{l_1}, ..., X_T = s_{l_T}|y_1, ..., y_T, \hat{\lambda}\right)$. Why don't we decompose this sum w.r.t. the values of $\log\pi, \log A, \log B$ since they are the terms that contain the variables of our concern.

$$\begin{aligned} \lambda^* = argmax_\lambda \sum_{i=1}^N \sum_{(l_1,...,l_T),l_1=i} \log \pi_i \cdot \mathbb{P}\left(X_1 = s_i, X_2 = s_{l_2}, ..., X_T = s_{l_T}|y, \hat{\lambda}\right) \quad (78)\\ + \sum_{i,j=1}^N \sum_{(l_1,...,l_T),(l_1,l_2)=(i,j)} \log A_{ij} \cdot \mathbb{P}\left(X_1 = s_i, X_2 = s_j, X_3 = s_{l_3} ..., X_T = s_{l_T}|y, \hat{\lambda}\right) \quad (79)\\ + ... \quad (80)\\ + \sum_{i,j=1}^N \sum_{(l_1,...,l_T),(l_{T-1},l_T)=(i,j)} \log A_{ij} \cdot \mathbb{P}\left(X_1 = s_{l_1}, ..., X_{T-2} = s_{l_{T-2}}, X_{T-1} = s_i, X_T = s_j|y, \hat{\lambda}\right) \quad (81)\\ + \sum_{i=1}^N \sum_{(l_1,...,l_T),l_1=i} \log B_i(y_1) \cdot \mathbb{P}\left(X_1 = s_i, X_2 = s_{l_2}, ..., X_T = s_{l_T}|y, \hat{\lambda}\right) \quad (82)\\ + ... \quad (83)\\ + \sum_{i=1}^N \sum_{(l_1,...,l_T),l_T=i} \log B_i(y_T) \cdot \mathbb{P}\left(X_1 = s_{l_1}, ..., X_{T-1} = s_{l_{T-1}}, X_T = s_i|y, \hat{\lambda}\right) \quad (84) \end{aligned}$$

After combining the same values of the log terms in the original expression, we get:

$$\lambda^* = argmax_\lambda \sum_{i=1}^N \log \pi_i \sum_{l_2,...,l_T} \mathbb{P}\left(X_1 = s_i, X_2 = s_{l_2}, ..., X_T = s_{l_T} | y, \hat{\lambda}\right) \tag{85}$$

$$+ \sum_{i,j=1}^N \log A_{ij} \sum_{l_3,...,l_T} \mathbb{P}\left(X_1 = s_i, X_2 = s_j, X_3 = s_{l_3} ..., X_T = s_{l_T} | y, \hat{\lambda}\right) \tag{86}$$

$$+ ... \tag{87}$$

$$+ \sum_{i,j=1}^N \log A_{ij} \sum_{l_1,...,l_{T-2}} \mathbb{P}\left(X_1 = s_{l_1}, ..., X_{T-2} = s_{l_{T-2}}, X_{T-1} = s_i, X_T = s_j | y, \hat{\lambda}\right) \tag{88}$$

$$+ \sum_{i=1}^N \log B_i(y_1) \sum_{l_2,...,l_T} \mathbb{P}\left(X_1 = s_i, X_2 = s_{l_2}, ..., X_T = s_{l_T} | y, \hat{\lambda}\right) \tag{89}$$

$$+ ... \tag{90}$$

$$+ \sum_{i=1}^N \log B_i(y_T) \sum_{l_1,...,l_{T-1}} \mathbb{P}\left(X_1 = s_{l_1}, ..., X_{T-1} = s_{l_{T-1}}, X_T = s_i | y, \hat{\lambda}\right) \tag{91}$$

The expression seems to be a mess, but actually we are very close to success if we simplify the sum w.r.t l_i:

$$\lambda^* = argmax_\lambda \sum_{i=1}^N \log \pi_i \mathbb{P}\left(X_1 = s_i | y, \hat{\lambda}\right) + \sum_{i,j=1}^N \log A_{ij} \mathbb{P}\left(X_1 = s_i, X_2 = s_j | y, \hat{\lambda}\right) \tag{92}$$

$$+ ... \tag{93}$$

$$+ \sum_{i,j=1}^N \log A_{ij} \mathbb{P}\left(X_{T-1} = s_i, X_T = s_j | y, \hat{\lambda}\right) + \sum_{i=1}^N \log B_i(y_1) \mathbb{P}\left(X_1 = s_i | y, \hat{\lambda}\right) \tag{94}$$

$$+ ... \tag{95}$$

$$+ \sum_{i=1}^N \log B_i(y_T) \mathbb{P}\left(X_T = s_i | y, \hat{\lambda}\right) \tag{96}$$

I am definitely sure that you have seen these terms before because they are just the probability γ and ξ. That's exciting! **Note**: in Alg. 1 and 2, the probability γ and ξ are both computed using old estimates $\hat{\lambda}$, so the notations are consistent, in the following context, by default, ξ, γ are all computed using old estimate $\hat{\lambda}$.

$$\lambda^* = argmax_\lambda \sum_{i=1}^N \log \pi_i \gamma_1(i) + \sum_{i,j=1}^N \log A_{ij}\xi_1(i,j) + ... + \sum_{i,j=1}^N \log A_{ij}\xi_{T-1}(i,j) \tag{97}$$

$$+ \sum_{i=1}^N \log B_i(y_1)\gamma_1(i) + ... + \sum_{i=1}^N \log B_i(y_T)\gamma_T(i) \tag{98}$$

After getting such an expression, it's then a trivial optimization problem w.r.t. all π_j, A_{ij}, B_{jk}. Note the natural constraints on these parameters that:

$$\sum_{j=1}^N \pi_j = 1 \tag{99}$$

$$\forall i \in [N], \sum_{j=1}^N A_{ij} = 1 \tag{100}$$

$$\forall j \in [N], \sum_{k=1}^M B_{jk} = 1 \tag{101}$$

Apply the Lagrange multiplier method for constrained optimization problems, the update formulas derived are the same as those in Alg. 1, proving that BW is essentially a special case of the EM scheme. Let us compute the update formulas below ony for π_i and the readers are welcome to compute those for A, B as exercise. Q is the Lagrange functional for the whole problem and μ_π, μ_A, μ_B are Lagrange multipliers. There's an abuse of notations below for simplicity, $\vec{1}$ stands for a column vector with all entries to be 1 of an appropriate size:

$$Q(\pi, A, B, \mu_\pi, \mu_A, \mu_B) = \sum_{i=1}^N \log \pi_i \gamma_1(i) + \sum_{i,j=1}^N \log A_{ij}\xi_1(i,j) + ... + \sum_{i,j=1}^N \log A_{ij}\xi_{T-1}(i,j) \tag{102}$$

$$+ \sum_{i=1}^N \log B_i(y_1)\gamma_1(i) + ... + \sum_{i=1}^N \log B_i(y_T)\gamma_T(i) \tag{103}$$

$$+ \mu_\pi \left(\sum_{i=1}^N \pi_i - 1 \right) + \mu_A^T(A\vec{1} - \vec{1}) + \mu_B^T(B\vec{1} - \vec{1}) \tag{104}$$

Extract the part relevant to π:

$$q(\pi, \mu_\pi) = \sum_{i=1}^{N} \log \pi_i \gamma_1(i) + \mu_\pi \left(\sum_{i=1}^{N} \pi_i - 1 \right) \tag{105}$$

$$\frac{\partial q}{\partial \pi_i} = \frac{\gamma_1(i)}{\pi_i} + \mu_\pi \tag{106}$$

$$\frac{\partial q}{\partial \mu_\pi} = \sum_{i=1}^{N} \pi_i - 1 \tag{107}$$

Some calculations tell us that:

$$\pi_j = \frac{\gamma_1(j)}{\sum_{i=1}^{N} \gamma_1(i)} \tag{108}$$

$$\sum_{i=1}^{N} \gamma_1(i) = \sum_{i=1}^{N} \mathbb{P}\left(X_1 = s_i | y_1, ..., y_T, \hat{\lambda} \right) = 1 \tag{109}$$

giving us exactly the same update formula as that in Alg. 1.

Of course, Alg. 2 can also be derived by EM in exactly the same way we have proved for the single emission sequence case above. The proof won't be presented here and the readers are welcome to take it as an exercise. (Just follow all the steps above and replace $y_1, ..., y_T$ with $(y_1^1, y_2^1, ..., y_T^1), ..., (y_1^R, y_2^R, ..., y_T^R)$, you will see that the numerator and denominator in Alg. 2 appear with a structure of average as a natural correspondence to the structure of expectation.)