1 HMM training

The Baum-Welch approach, which effectively estimates model parameters through an iterative Expectation-Maximization process, is the mainstay of Hidden Markov Models (HMM) training. In order to prevent numerical underflow, we used a scaling factor. The algorithm uses both the Forward and Backward procedures to compute intermediate variables. These calculations make it possible to calculate two essential elements that are essential for parameter re-estimation: occupation likelihoods (state probabilities) and transition likelihoods (state transition probabilities). Using the determined likelihoods to optimize the likelihood of witnessing the training sequences, the procedure iteratively fine-tunes the HMM parameters (transition and emission probabilities) until convergence as shown in **Figure 1**.

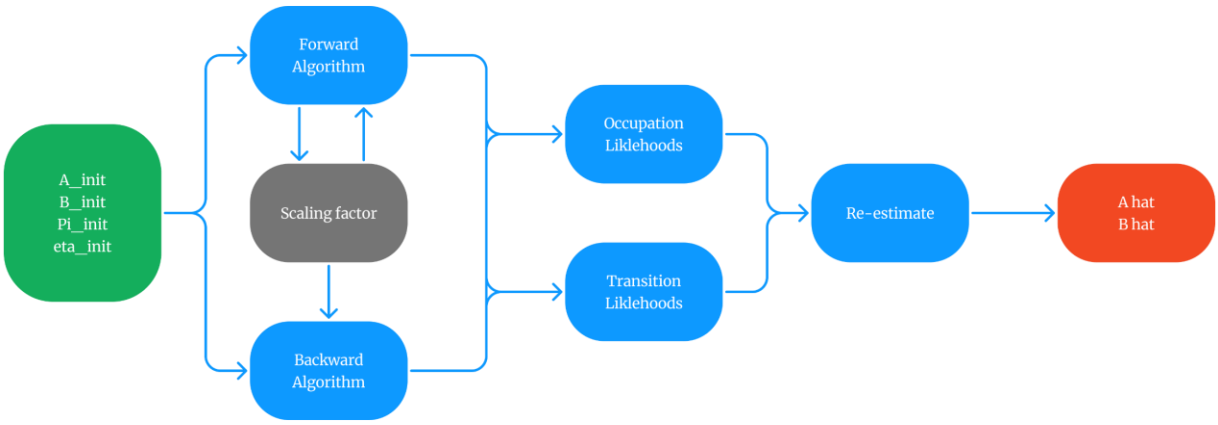

Figure 1 *HMM training algorithm*

1.1 Forward algorithm

By employing dynamic programming, the Forward algorithm effectively determines the probability of an observation sequence in a Hidden Markov Model without resorting to the exponential difficulty of adding up every potential state sequence. It functions in three essential steps:

Initialization Step: The procedure calculates the initial probability for each state i for the first observation by multiplying two terms: the likelihood that state I would emit the first observation $b_i(\mathbf{o}_1)$ and the probability of starting in state i (π_i) [1].

$$\alpha_1^r(i) = \pi_i b_i(\mathbf{o}_1^r) \tag{1}$$

Recursion Step: The algorithm considers every path that could lead to each current state for each succeeding time step. It calculates the probability for each state j at time t by taking into account Previous state probabilities $\alpha_{t-1}(i)$. Transition probabilities $a_{ij}$and Emission probability of current observation $b_j(\mathbf{o}_t^r)$.

$$\text{For } t = 2,3,\dots,T,$$
$$\alpha_t^r(j) = \left[\sum_{i=1}^{N} \alpha_{t-1}^r(i) a_{ij}\right] b_j(\mathbf{o}_t^r) \tag{2}$$

Termination Step: The terminal forward variables for each of the possible end states are added up to determine the final probability of the complete observation sequence.

$$P^r = P(\mathcal{O}^r \mid \lambda) = \sum_{i=1}^{N} \alpha_T^r(i)\eta_i. \tag{3}$$

Without explicitly listing every conceivable state sequence, this method effectively builds on previously determined values to compute the overall likelihood by accumulating probabilities throughout the process.

1.2 Backward algorithm

Similar to its Forward counterpart, the Backward method uses dynamic programming to calculate the likelihood of a sequence in a Hidden Markov Model, but it operates in the opposite way. It functions as follows:

Initialization Step: The procedure initializes the backward variables for every state with a value of η at the last observation time T. Our starting point for moving backward through the sequence is established by this initialization.

$$\beta_T^r(i) = \eta_i \tag{4}$$

Recursion Step: The Backward algorithm's recursive computation from the final time step backwards is the fundamental component of its dynamic programming methodology. By taking into account all potential future routes, the algorithm determines the probability of producing the remaining observations for each state at each time step prior to T. We take into consideration the probability of transitioning to any possible next state, the probability that the next state would emit the next observation we actually see, and the probability of generating all remaining observations from that next state in order to account for all possible paths leading from each current state. By going backwards through the observation series, this recursive approach effectively combines probability without explicitly listing every possibility, accumulating information about likely state sequences.

$$\text{For } t = T-1, T-2, \dots, 1,$$
$$\beta_t^r(i) = \sum_{j=1}^{N} a_{ij} b_j(\mathbf{o}_{t+1}^r) \beta_{t+1}^r(j) \tag{5}$$

Termination Step: The Backward algorithm's Termination step calculates the overall probability of the observation sequence by considering the initial state probabilities. After computing all backward variables back to time t=1, we can obtain the total probability of our observation sequence by combining the initial state probabilities, the emission probabilities of the first observation, and the backward variables at time t=1 for all possible states. This combination gives us the same total probability as the Forward algorithm, just computed in the reverse direction.

$$P^r = \sum_{i=1}^{N} \pi_i b_i(\mathbf{o}_1^r) \beta_1^r(i) \tag{6}$$

1.3 Underflow problem

Both forward and backward algorithms in Hidden Markov Models (HMM) encounter underflow issues when they constantly multiply probabilities, producing incredibly small quantities that are impossible for computers to process correctly. We employ scaling factors at each time step t to solve this. To maintain the values within a calculable range, we multiply the forward variables $\alpha_t^r(i)$ and backward variables $\beta_t^r(i)$ by a scaling coefficient c_t . To ensure consistency, the same scaling factors are applied to both algorithms, and these scaling coefficients are utilized to determine the final probability [2].

Forward scaling:

$$c_t = \frac{1}{\sum_{i=1}^{N} \alpha_t(i)} \tag{7}$$

$$\alpha_t(i) = \ c_t * \ \alpha_t(i) \tag{8}$$

Backward scaling

$$\beta_t(i) = \ c_t * \ \beta_t(i) \tag{9}$$

1.4 Occupation likelihoods

Occupation likelihood $\gamma_t(i)$, which is the probability of being in state i at time t given the complete observation sequence, is estimated in HMM after forward and backward variables have been calculated. It integrates data from backward variables $\beta_t(i)$, which takes future observations into account, and forward variables $\alpha_t(i)$, which takes observations up to time t into account. Each state's forward and backward variables are multiplied, and the result is normalized by the observation sequence's overall probability. Considering every observation, this gives a comprehensive picture of state occupancy. Understanding the most likely state sequence that produced the observations and re-estimating parameters in the Baum-Welch algorithm depend on these occupation probabilities.

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(\mathcal{O} \mid \lambda)} \tag{10}$$

1.5 Transition likelihoods

Given the complete observation sequence and model parameters, transition likelihood $\varepsilon_t(i,j)$ estimation determines the probability of being in state i at time t and changing to state j at time $t+1$. Forward variables $\alpha_t(i)$, backward variables $\beta_{t+1}(i)$, transition probabilities a_{ij} and emission probabilities $b_j(\mathrm{o}_{t+1})$ are used to calculate this probability. utilizing forward variables, the computation takes into account the likelihood of staying in state i until time t, transitioning to state j, producing the observation at $t+1$, and then utilizing backward variables to account for all further observations from $t+1$ onward. During the parameter re-estimation stage of the Baum-Welch method, these transition likelihoods are essential for updating the transition probability matrix.

$$\xi_t(i,j) = \frac{\alpha_{t-1}(i)a_{ij}b_j(\mathbf{o}_t)\beta_t(j)}{P} \tag{11}$$

1.6 Re-estimation

Using the occupation and transition likelihoods that were previously determined, the re-estimation stage of the HMM (Baum-Welch algorithm) modifies the model parameters $\lambda = \{\mathrm{A}, \mathrm{B}, \pi\}$. The occupation likelihood at $\mathrm{t} = 1$ is used to update the initial state probabilities π. To re-estimate the transition probabilities A, $\xi_t(i,j)$ and $\gamma_t(i)$ are used to calculate the ratio of expected transitions between states to total transitions from the source state. By using $\gamma_t(i)$ to compare observation occurrences in each state to total state occupations, the emission probabilities B are updated. The new model λ' formed by these modified parameters raises the likelihood $P(\mathcal{O} \mid \lambda)$ till convergence.

2 HMM evaluation

2.1 Viterbi algorithm

By monitoring the optimal route to every state at every time step, the Viterbi algorithm determines the most likely state sequence in HMM. It employs a dynamic programming technique with two primary variables: $\psi_t(i)$ holds the prior state that resulted in this highest probability, and $\delta_t(i)$ indicates the highest likelihood of a path terminating in state i at time t. As it advances through time, the algorithm uses emission probabilities $b_j(\mathbf{o}_t)$ and transition probabilities a_{ij} to calculate these variables. It requires scaling in order to avoid underflow, just like forward-backward algorithms. It finds the ideal state sequence by going back through $\psi_t(i)$ after arriving at the last observation. We have used logarithmic transformation and sum of logs instead of scaling as in forward and backward algorithms [3].

3 References

[1] J. H. M. Daniel Jurafsky, "Hidden Markov Models," in *Speech and Language Processing*, 2024.

[2] C. M. Bishop, "13-Sequential Data, 13.1-Hidden Markov Models, 13.2.4-Scaling factors," in *Pattern Recognition and Machine Learning*, 2006.

[3] T. S. R. Z. H. Z. Eric P. Xing, "Case Studies: HMM and CRF," *CMU, Department of Computer Science, Probabilistic Graphical Models, Lecture 6 ,* 2020.