Implementation of Baum-Welch algorithm for HMM in Mahout Samsara

Manogna Vemulapati

October 16, 2018

Introduction

A Hidden Markov Model (HMM) λ is specified as a triplet (A, B, π) where:

- The number of hidden states is N and they are specified as the set $S = \{S_0, S_1, ..., S_{N-1}\}$.
- The number of observation symbols is M and they are specified as the set $V = \{v_0, ..., v_{M-1}\}$.
- The state transition probability distribution matrix A is a matrix of dimension $N \times N$. The element a_{ij} of the matrix A is the probability of transitioning from state S_i to state S_j.
- The emission probability distribution matrix B is a matrix of dimension $N \times M$. The element $b_j(k)$ of the matrix B is the probability of emitting observation symbol v_k from state S_j.
- The probability distribution for the initial state is specified by the vector $\pi = \{\pi_i\}$ where pi_i is the probability of being in state S_i at time $t = 0$.

Given an observation sequence O of observation symbols from the set V, the learning problem is to adjust the model parameters λ such that the probability $P(O|\lambda)$ is maximized. The Baum-Welch algorithm provides a solution for the training problem.

Baum-Welch Algorithm

The Baum-Welch algorithm is an Expectation-Maximization (EM) algorithm which computes the maximum likelihood estimate of the parameters of HMM given a set of observation sequences. It is an iterative algorithm – in each iteration, it computes the forward variables and backward variables, then uses these variables to update the model parameters so that $P(O|\bar{\lambda}) > P(O|\lambda)$, where $\bar{\lambda}$ is the model with the updated parameters. The algorithm iterates until the model parameters converge.

Forward Variables

For an observation sequence of length T ($O_0...O_{T-1}$), the forward variables are defined as

$$\alpha_t(i) = P(O_0O_1...O_{t-1}, q_t = S_i|\lambda)$$

which is the probability of the partial observation sequence $O_0O_1...O_t$ up to time t and state S_i at time t, given the model λ. The forward variables are computed inductively as follows:

- Initialization:

$$\alpha_0(i) = \pi_i b_i(O_0), 0 \leq i \leq N-1$$

- Induction:

$$\alpha_{t+1}(j) = \left[\sum_{i=0}^{N-1} \alpha_t(i)a_{ij}\right] b_j(O_{t+1}), 0 \leq t \leq T-2, 0 \leq j \leq N-1$$

Backward Variables

The backward variable $beta_t(i)$ is the probability of the partial observation sequence from time $t+1$ to the end $T-1$, given the model λ and that the HMM is in state S_i at time t.

$$\beta_t(i) = P(O_{t+1}...O_{T-1}|q_t = S_i, \gamma)$$

The backward variables are computed inductively as follows.

- Initialization:

$$\beta_{T-1}(i) = 1, 0 \leq i \leq N-1$$

- Induction:

$$\beta_t(i) = \sum_{j=0}^{N-1} a_{ij}b_j(O_{t+1})\beta_{t+1}(j), 0 \leq t \leq T-2, 0 \leq i \leq N.$$

Gamma and Xi Variables

The gamma variable $\gamma_t(i)$ is the probability of being in state S_i at time t given the observation sequence O and the model λ.

$$\lambda_t(i) = P(q_t = S_i|O, \lambda) = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)}$$

The Xi variable $\xi_t(i,j)$ is the probability of being in state S_i at time t and in state S_j at time $t+1$ given the model λ and the observation sequence O.

$$\xi_t(i,j) = P(q_t = S_i, q_{t+1} = S_j|O, \lambda) = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1}\beta_{t+1}(j)}{P(O|\lambda)}$$

Probability of an observation sequence

The probability of an observation sequence O given a model λ is computed as follows.

$$P(O|\lambda) = \sum_{i=0}^{N-1} \alpha_{T-1}(i)$$

Update of Model Parameters

The sum of gamma variables for a particular state i. That is, the expression $\sum_{t=0}^{T-2} \gamma_t(i)$ can be interpreted as the expected number of times that the state S_i is visited given the model parameters and the observation sequence O. Additionally, the summation of Xi variables $\sum_{t=0}^{T-2} \xi_t(i,j)$ can be interpreted as the expected number of transitions from state S_i to state S_j. Hence the ratio of the latter over the former is the updated probability of transition from state S_i to state S_j. Thus, an iteration of Baum-Welch algorithm adjusts the parameters as below.

- Initial Probabilities Vector

$$\bar{\pi}_i = \gamma_0(i)$$

- State Transition Probability Distribution

$$\bar{a}_{ij} = \frac{\sum_{t=0}^{T-2} \xi_t(i,j)}{\sum_{t=0}^{T-2} \gamma_t(i)}$$

- Emission Probability Distribution

$$\bar{b}_j(k) = \frac{\sum_{t=0,O_t=v_k}^{T-1} \gamma_t(j)}{\sum_{t=0}^{T-1} \gamma_t(j)}$$

Numerical Stability and Scaling

The value of a forward variable $\alpha_t(i)$ quickly tends to zero as the value of t becomes large. The solution to this problem is to scale the forward variables at each induction step. One common scaling scheme is to define a scaling factor which depends only on time t but is independent of the state i as described below. The scaled forward variables $\ddot{\alpha}_t(i)$ and the scaling factors c_t are computed by induction as follows.

- Initialization

$$\ddot{\alpha}_0(i) = \alpha_0(i) \quad 0 \leq i \leq N-1$$
$$c_0 = \frac{1}{\sum_{i=0}^{N-1} \ddot{\alpha}_0(i)}$$
$$\hat{\alpha}_0(i) = c_0 \ddot{\alpha}_0(i) \quad 0 \leq i \leq N-1$$

- Induction

$$\ddot{\alpha}_t(i) = \sum_{j=0}^{N-1} \hat{\alpha}_{t-1}(j) a_{ji} b_i(O_t)$$
$$c_t = \frac{1}{\sum_{i=0}^{N-1} \ddot{\alpha}_t(i)}$$
$$\hat{\alpha}_t(i) = c_t \ddot{\alpha}_0(i) \quad 0 \leq i \leq N-1$$

To compute the scaled backward variables $\ddot{\beta}_t(i)$, the same scaling factors which were computed for the scaled forward variables are used.

- Initialization

$$\ddot{\beta}_{T-1}(i) = 1$$
$$\hat{\beta}_{T-1}(i) = c_{T-1} \ddot{\beta}_{T-1}(i)$$

- Induction

$$\ddot{\beta}_t(i) = \sum_{j=0}^{N-1} \hat{\beta}_{t+1}(j) a_{ji} b_i(O_{t+1})$$
$$\hat{\beta}_t(i) = c_t \ddot{\beta}_t(i)$$

Probability of an observation sequence with scaled variables

The probability of an observation sequence O given a model λ is computed as follows:

$$C_t = \prod_{\tau=0}^{t} c_\tau$$
$$P(O|\lambda) = 1/C_{T-1}$$

Using the scaled forward and backward parameters the model parameters are adjusted as follows:

- Initial Probabilities Vector

$$\bar{\pi}_i = \hat{\alpha}_0(i) \hat{\beta}_0(i) / c_0$$

- State Transition Probability Distribution

$$\bar{a}_{ij} = \frac{\sum_{t=0}^{T-2} \hat{\alpha}_t(i).a_{ij} b_j(O_{t+1}).\hat{\beta}_{t+1}(j)}{\sum_{t=0}^{T-2} \hat{\alpha}_t(i).\hat{\beta}_t(i)/c_t}$$

- Emission Probability Distribution

$$\bar{b}_j(k) = \frac{\sum_{t=0, O_t=v_k}^{T-1} \hat{\alpha}_t(j).\hat{\beta}_t(j)/c_t}{\sum_{t=0}^{T-1} \hat{\alpha}_t(j).\hat{\beta}_t(j)/c_t}$$

Training with Multiple Observation Sequences

Suppose we have L independent observation sequences where the observation sequence indexed by l is denoted by O_l and $0 \leq l \leq L-1$. In order to update the parameters of the model, we need to do the following.

- Starting probabilities: From each observation sequence, compute the expected number of times in each state at time $t = 0$. For each state i, we can compute the sum of expected number of times in that state at time $t = 0$ from all the sequences. From this we can update the initial probabilities vector.
- Expected number of transitions: From each observation sequence, compute the expected number of times of transition from state i to state j. For each ordered pair of states (i, j), we can compute the sum of expected number of times of transition from state i to state j due to all sequences. Once we compute the sums of transitions for row i of the transition matrix, we can update that row of the transition matrix by computing the total number of times visiting state
- Expected number of emissions: From each observation sequence, compute the expected number of times being in state i and emitting symbol j. For each state i and symbol j, we can compute the total number of times being in state i and emitting symbol j from all sequences. Each row of the emission matrix can be updated by computing the total number times visiting that row.

The parameters are updated as follows.

- Initial Probabilities Vector

$$\bar{\pi}_i = \frac{\sum_{l=0}^{L-1} \alpha_0^l(i)\beta_0^l(i)/P(O^l|\lambda)}{L}$$

- State Transition Probability Distribution

$$\bar{a}_{ij} = \frac{\sum_{l=0}^{L-1}\sum_{t=0}^{T_l-2} \alpha_t^l(i)a_{ij}b_j(O_{t+1}^l)\beta_{t+1}^l(j)/P(O^l|\lambda)}{\sum_{l=0}^{L-1}\sum_{t=0}^{T_l-2} \alpha_t^l(i)\beta_{t+1}^l(j)/P(O^l|\lambda)}$$

- Emission Probability Distribution

$$\bar{b}_j(k) = \frac{\sum_{l=0}^{L-1}\sum_{t=0,O_t=v_k}^{T_l-1} \hat{\alpha}_t^l(j).\hat{\beta}_t^l(j)/P(O^l|\lambda)}{\sum_{l=0}^{L-1}\sum_{t=0}^{T_l-1} \alpha_t^l(j)\beta_t^l(j)/P(O^l|\lambda)}$$

If we are using the scaled forward and backward variables, then the update equations are as follows.

- Initial Probabilities Vector

$$\bar{\pi}_i = \frac{\sum_{l=0}^{L-1} \hat{\alpha}_0^l(i)\hat{\beta}_0^l(i)/c_0^l}{L}$$

- State Transition Probability Distribution

$$\bar{a}_{ij} = \frac{\sum_{l=0}^{L-1}\sum_{t=0}^{T_l-2} \hat{\alpha}_t^l(i)a_{ij}b_j(O_{t+1}^l)\hat{\beta}_{t+1}^l(j)}{\sum_{l=0}^{L-1}\sum_{t=0}^{T_l-2} \hat{\alpha}_t^l(i)\hat{\beta}_{t+1}^l(j)/ct^l}$$

- Emission Probability Distribution

$$\bar{b}_j(k) = \frac{\sum_{l=0}^{L-1}\sum_{t=0,O_t=v_k}^{T_l-1} \hat{\alpha}_t^l(j)\hat{\beta}_t^l(j)/c_t^l}{\sum_{l=0}^{L-1}\sum_{t=0}^{T_l-1} \hat{\alpha}_t^l(j)\hat{\beta}_t^l(j)/c_t^l}$$

Distributed Training in Samsara

The current implementation of distributed training of HMM in Samsara is based on the HMM training in MapReduce described in [2]. During each iteration of the Baum-Welch algorithm, each node in a cluster works on a block of independent observation sequences. Each node in the cluster executes the following steps for each observation sequence in the block.

- Compute forward variables matrix of dimensions $T \times N$ where T is the length of the observation sequence. The forward variables can be either scaled or not.
- Compute backward variables matrix of dimensions $T \times N$ where T is the length of the observation sequence. If the forward variables were scaled in the previous step, then use the same scaling factors to scale backward variables too.
- For each state i, compute the expected number of times being in that state at time $t = 0$.
- For each state i, compute the expected number of transitions from the state to every state j where $0 \leq j \leq N-1$.
- For each state i, compute the expected number of emissions of symbol k where $0 \leq k \leq M-1$.

The `mapBlock` operator transforms a block of observation sequences (which is a matrix with R rows representing a subset of R observation sequences) into a matrix of shape $R \times (N + N^2 + N * M)$.Each row in the input block (which is an independent observation sequence) is mapped to a row in the output block with $(N+N^2+N*M)$ columns as described below. The first N columns of the output row contain the values $\gamma_0(i)$, for $0 \leq i \leq N-1$, which are the probabilities of starting in state i for each of the N states. The next N^2 columns store the row

major representation of the $N \times N$ matrix which contains the expected transition counts. The element e_{ij} of this matrix is the expected number of transitions from state i to state j given the observation sequence. The last $N * M$ columns of the output block matrix store the row major representation of the $N \times M$ matrix of expected emission counts. The element f_{ij} of this matrix contains the expected number of times the symbol j is emitted from state i given the observation sequence. When all the blocks of the input DRM of observation sequences are processed, the parameters of the model are updated as follows.

- Initial State Probability Distribution: To update the initial probabilities vector, compute the total count of expected number of times of being in state i at time $t = 0$ for all $0 \leq N - 1$. The element π_i is calculated as the ratio of the total count of expected number of times of being in state i at time $t = 0$ and the sum of counts for all states.

- State Transition Probability Distribution: To update row i of the transition matrix, for each element a_{ij} we need to compute the cumulative expected number of transitions from state i to state j from all the observation sequences. The sum of all these cumulative expected number of transitions gives us the total expected number of times the state i is visited. If we divide the cumulative expected number of transitions from state i to state j from all the observation sequences by the total expected number of times the state i is visited, we get the updated probability of transition from state i to state j.

- Emission Probability Distribution: To update row j of the emission matrix, for each element $b_j(k)$, we need to compute the cumulative expected number of times the symbol k is emitted while being in state j from all the observation sequences. The sum of all these cumulative expected number of emissions gives us the total expected number of times the state j is visited. If we divide the cumulative expected number of emissions from state j of symbol k by the total expected number of times the state j is visited, we get the updated probability of emission from state j of symbol k.

References

[1] Dawei Shen, *Some Mathematics for HMM.* `https://pdfs.semanticscholar.org/4ce1/9ab0e07da9aa10be1c336400c8e4d8fc36c5.pdf`

[2] Jimmy Lin and Chris Dyer, *Data-intensive text processing with MapReduce.* `https://http://www.iro.umontreal.ca/~nie/IFT6255/Books/MapReduce.pdf`

[3] Xiaolin Li, Marc Parizeau and Rejean Plamondon, *Training Hidden Markov Models with Multiple Observations ? A Combinatorial Method*

https://http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.335.1457&rep=rep1&type=pdf