02457 Non-Linear Signal Processing, Mini project in speech processing (Part II of III)

This exercise is based on L.R. Rabiner *A tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*, Proceedings of the IEEE **77**, 257-286, (1989).

Print and comment on the figures produced by the software as outlined below at the **Checkpoints**, this exercise is split in three parts, hence, performed over three Thursday sessions.

Training the sequence model

First, let us define the elements of a discrete observation HMM, $\mathcal{M} = \{S, \pi(1), \mathbf{A}, \mathbf{B}, \{\mathbf{y}_k, 1 \leq k \leq K\}\}$, characterized by the state equation and observation equation, i.e.,

$$\begin{aligned} \pi(t) &= \mathbf{A}\pi(t-1), \qquad & \pi(t) = [P(x_t = 1), \cdots, P(x_t = S)]^T \\ \mathbf{p}(t) &= \mathbf{B}\pi(t), \qquad & \mathbf{p}(t) = [P(y_t = 1), \cdots, P(y_t = K)]^T \end{aligned}$$

Here, S is the number of states in the model, x_t is the state random process, $\pi(t)$ is the state probability vector at time t, and $\mathbf{A}$ is the S-by-S state transition matrix with elements $a_{ij} = a(i|j) = P(x_t = i|x_{t-1} = j)$ (note: transpose of Rabiners definition!). In the observation equation, K is the number of distinct observation symbols per state (the number of vectors $\mathbf{y}_k$ in the VQ codebook), y_t is the observation random process, $\mathbf{p}(t)$ is the observation probability vector at time t, and $\mathbf{B}$ is the K-by-S observation probability matrix with elements $b_{ki} = b(k|i) = P(y_t = k|x_t = i)$.

Hence an HMM requires specification of the two model parameters, S and K, and specification of the three probability measures $\mathbf{A}$, $\mathbf{B}$, and $\pi(1)$. Thus, the training problem is to adjust the model parameters to maximize the likelihood $P(y|\mathcal{M})$, i.e., the probability of the observation sequence $y = \{y_1, y_2, \ldots, y_T\}$, given the model.

Let $\mathcal{I} = \{i_1, i_2, \ldots, i_T\}$ denote a specific state sequence, then the likelihood $P(y|\mathcal{M})$ can be found as

$$\begin{aligned} P(y|\mathcal{M}) &= \sum_{\mathcal{I}} P(y, \mathcal{I}|\mathcal{M}) = \sum_{\mathcal{I}} P(y|\mathcal{I}, \mathcal{M}) P(\mathcal{I}|\mathcal{M}) \\ &= \sum_{\mathcal{I}} b(y_1|i_1) b(y_2|i_2) \cdots b(y_T|i_T) \times P(x_1 = i_1) a(i_2|i_1) a(i_3|i_2) \cdots a(i_T|i_{T-1}) \end{aligned}$$

In practice, computation of $P(y|\mathcal{M})$ is based on the following forward/backward recur-

sions,

$$\begin{aligned}
\alpha(y_1^t, i) &= \sum_{j=1}^{S} \alpha(y_1^{t-1}, j) a(i|j) b(y_t|i), \qquad \alpha(y_1^1, i) = P(x_1 = i) b(y_1|i) \\
\beta(y_{t+1}^T|i) &= \sum_{j=1}^{S} \beta(y_{t+2}^T|j) a(j|i) b(y_{t+1}|j), \qquad \beta(y_{T+1}^T|i) = 1
\end{aligned}$$

where $\alpha(y_1^t, i)$ is the joint probability of having generated the partial forward sequence y_1^t and having arrived at state i at the t'th step. Similarly, $\beta(y_{t+1}^T|i)$ denote the probability of generating the backward partial sequence y_{t+1}^T, given that the state sequence emerges from state i at time t. To avoid numerical problems (underflow) in the computations of the forward/backward recursions, we must scale $\alpha(\cdot,\cdot)$ and $\beta(\cdot|\cdot)$ in each step with c_t as follows

$$\begin{aligned}
\hat{\alpha}(y_1^t, i) &= c_t \tilde{\alpha}(y_1^t, i), \qquad \tilde{\alpha}(y_1^t, i) = \sum_{j=1}^{S} \hat{\alpha}(y_1^{t-1}, j) a(i|j) b(y_t|i), \qquad c_t = \left(\sum_{i=1}^{S} \tilde{\alpha}(y_1^t, i) \right)^{-1} \\
\hat{\beta}(y_{t+1}^T|i) &= c_t \tilde{\beta}(y_{t+1}^T|i), \qquad \tilde{\beta}(y_{t+1}^T|i) = \sum_{j=1}^{S} \hat{\beta}(y_{t+2}^T|j) a(j|i) b(y_{t+1}|j)
\end{aligned}$$

Based on the scaled forward/backward recursions, an iterative estimation procedure for computing a Hidden Markov Model, $\mathcal{M}$, corresponding to a local maximum of the likelihood $P(y|\mathcal{M})$, can be devised. This algorithm, known as F-B reestimation, takes a model $\mathcal{M} = \{S, \pi(1), \mathbf{A}, \mathbf{B}, \{\mathbf{y}_k, 1 \le k \le K\}\}$ and the training observation, $y = y_1^T$, to compute a new model $\bar{\mathcal{M}} = \{S, \bar{\pi}(1), \bar{\mathbf{A}}, \bar{\mathbf{B}}, \{\mathbf{y}_k, 1 \le k \le K\}\}$ as

$$\begin{aligned}
\bar{a}(j|i) &= \frac{\sum_{t=1}^{T-1} \hat{\alpha}(y_1^t, i) a(j|i) b(y_{t+1}|j) \hat{\beta}(y_{t+2}^T|j)}{\sum_{t=1}^{T-1} \hat{\alpha}(y_1^t, i) \hat{\beta}(y_{t+1}^T|i)/c_t} \\
\bar{b}(k|j) &= \frac{\sum_{y_t=k, t=1}^{T} \hat{\alpha}(y_1^t, j) \hat{\beta}(y_{t+1}^T|j)/c_t}{\sum_{t=1}^{T} \hat{\alpha}(y_1^t, j) \hat{\beta}(y_{t+1}^T|j)/c_t} \\
\bar{\pi}_i(1) &= P(x_1 = i) = \frac{\hat{\alpha}(y_1^1, i) \hat{\beta}(y_2^T|i)}{c_1}
\end{aligned}$$

If the likelihood has increased such that $P(y|\bar{\mathcal{M}}) - P(y|\mathcal{M}) \ge \epsilon$, where ϵ is a given tolerance, then reestimate the model with $\mathcal{M} = \bar{\mathcal{M}}$. The required likelihood can be obtained from any time slot in the lattice, i.e.,

$$P(y|\mathcal{M}) = \sum_{i=1}^{S} \alpha(y_1^t, i) \beta(y_{t+1}^T|i) = \sum_{i=1}^{S} \alpha(y_1^T, i) = \left(\prod_{\tau=1}^{T} c_\tau \right)^{-1}$$

Again, to avoid numerical problems ($P(y|\mathcal{M})$ is small), the log-likelihood measure is used instead as given by

$$\log P(y|\mathcal{M}) = -\sum_{\tau=1}^{T} \log c_\tau$$

The last thing to consider is the training of an HMM with L multiple observation sequences (different occurrences of the same word). Let $y^{(l)}$ denote the lth observation of length T_l, and let superscript l indicate results for this sequence, then the F-B reestimation algorithm must be modified as

$$\begin{aligned}
\bar{a}(j|i) &= \frac{\sum_{l=1}^{L}\sum_{t=1}^{T_l-1}\hat{\alpha}^{(l)}(y_1^t,i)a(j|i)b(y_{t+1}|j)\hat{\beta}^{(l)}(y_{t+2}^{T_l}|j)}{\sum_{l=1}^{L}\sum_{t=1}^{T_l-1}\hat{\alpha}^{(l)}(y_1^t,i)\hat{\beta}^{(l)}(y_{t+1}^{T_l}|i)/c_t^{(l)}} \\
\bar{b}(k|j) &= \frac{\sum_{l=1}^{L}\sum_{y_t=k,t=1}^{T_l}\hat{\alpha}^{(l)}(y_1^t,j)\hat{\beta}^{(l)}(y_{t+1}^{T_l}|j)/c_t^{(l)}}{\sum_{l=1}^{L}\frac{1}{P(y^{(l)}|\mathcal{M})}\sum_{t=1}^{T_l}\hat{\alpha}^{(l)}(y_1^t,j)\hat{\beta}^{(l)}(y_{t+1}^{T_l}|j)/c_t^{(l)}} \\
\bar{\pi}_i(1) &= P(x_1=i) = \frac{1}{L}\sum_{l=1}^{L}\frac{\hat{\alpha}^{(l)}(y_1^1,i)\hat{\beta}^{(l)}(y_2^T|i)}{c_1^{(l)}}
\end{aligned}$$

Checkpoint 10.3

Inspect the update rules for HMM training. Show that they are multiplicative in $\bar{\mathbf{A}}$, i.e., that the updated value is proportional to the 'old' value. This means that if an entry in transition matrix is initialized to value zero it will remain zero after the update.

We consider a simulation based on a given HMM 'teacher', a model with a specific set of parameters as set up in the matlab script main10c.m. Create a flow chart of the operation of the script. What type of sequences are expected from the teacher HMM?.

We want to generate bias-variance trade off curves, i.e. training and test log-likelihoods as a function of the nunmber of hidden states in the student HMM. Explain why the likelihood on a test set can be used to evaluate the student. This experiment will reveal if we can 'over-train' the HMM if too many states are used. How many parameters are estimated in the HMM for a given number of hidden states. Explain the bias-variance trade off for these models. Can the transition matrices and emission probabilities be directly compared? (Hint: permutation symmetry of the hidden state index).

How many states and emissions are present in the teacher HMM? Is the EM algorithm trained HMM reproducible, i.e., consistent in determining the number of states? Which model has the highest value of the likelihood among the trained models with different no. of states? The EM algorithm is run five times to get an ensemble average of the suggested no. of states. What is the overall typical number of states to recommend?

Try to stop the matlab script and investigate the estimated transition matrix, when the no. of states is 8 (insert a break point). Usually to many free parameters—here states—result in overtraining. This is not the case here. Why is the model not overtrained? See figure 1 - Hint: States and their emission probabilities can be cloned.

Checkpoint 10.4

We will now investigate the convergence of the EM algorithm. We will start the EM algorithm with the optimal parameters, i.e. the teacher's parameters, to see whether the solution stays close to the teacher or not. The matlab script file `main10d.m` will run with two different initial conditions: the teacher and random initialization. Compare the likelihood of the two different initializations. Try to run several times and change the number of training samples `nS`. Comment on what you observe, e.g., interpret the trained HMM. What type of sequences do they produce?

Peter S. K. Hansen and Lars Kai Hansen, November 2001, November 2005.
Rasmus Elsborg Madsen and Ole Winther, November 2003.