Training Asynchronous Input/Output Hidden Markov Models

Yoshua Bengio
bengioy@iro.umontreal.ca
Dept. IRO
Universit de Montral
Montral, Qc, H3C 3J7
CANADA

Samy Bengio
bengio@inrs-telecom.uquebec.ca
INRS-Tlcommunications
16, Place du Commerce
Ile-des-Soeurs, Qc H3E 1H6
CANADA

http://www.iro.umontreal.ca/labs/neuro/

Abstract

In learning tasks in which input sequences are mapped to output sequences, it is often the case that the input and output sequences are not synchronous. For example, in speech recognition, acoustic sequences are longer than phoneme sequences. Input/Output Hidden Markov Models have already been proposed to represent the distribution of an output sequence given an input sequence of the same length. We extend here this model to the case of asynchronous sequences, and show an Expectation-Maximization algorithm for training such models.

1 Introduction

Supervised learning algorithms for sequential data minimize a training criterion that depends on pairs of input and output sequences. It is often assumed that input and output sequences are synchronized, i.e., that each input sequence has the same length as the corresponding output sequence. For instance, recurrent networks [Rumelhart et al., 1986] can be used to map input sequences to output sequences, for example minimizing at each time step the squared difference between the actual output and the desired output. Another example is a recently proposed recurrent mixture of experts connectionist architecture which has an interpretation as a probabilistic model, called *Input/Output Hidden Markov Model* (IOHMM) [Bengio and Frasconi, 1996, Bengio and Frasconi, 1995]. This model represents the distribution of an output sequence when given an input sequence of the same length, using a hidden state variable and a Markovian independence assumption, as in *Hidden Markov Models* (HMMs) [Levinson et al., 1983, Rabiner, 1989], in order to simplify the distribution. IOHMMs are a form of probabilistic transducers [Pereira et al., 1994, Singer, 1996], with input and output variables which can be discrete as well as continuous-valued.

However, in many sequential problems where one tries to map an input sequence to an output sequence, the length of the input and output sequences may not be equal. Input and output sequences could behave at different time scales. For example, in a speech recognition problem where one wants to map an acoustic signal to a phoneme sequence, each phoneme approximately corresponds to a subsequence of the acoustic signal, therefore the input acoustic sequence is generally longer than the output phoneme sequence, and the alignment between inputs and outputs is often not available.

In comparison with HMMs, emission and transition probabilities in IOHMMs vary with time as a function of an input sequence. Unlike HMMs, IOHMMs with discrete outputs are discriminant models. Furthermore, the transition probabilities and emission probabilities are generally better matched, which reduces a problem observed in speech recognition HMMs: because outputs are in a much higher dimensional space than transitions in HMMs, the dynamic range of transition probabilities is much less than that of emission probabilities. Therefore the choice between different paths (during recognition) is mostly influenced by emission rather than transition probabilities.

In this paper, we present an extension of IOHMMs to the asynchronous case. We first present the probabilistic model, then derive an exact Expectation-Maximization (EM) algorithm for training asynchronous IOHMMs. For complex distributions (e.g., using artificial neural networks to represent transition and emission distributions), a Generalized EM algorithm or gradient ascent in likelihood can be used. Finally, a Viterbi algorithm is presented for the recognition of new sequences.

2 The Model

Let us note x_1^T for an input sequence $x_1, x_2, \ldots, x_T$, and similarly y_1^S for an output sequence $y_1, y_2, \ldots, y_S$. In this paper we consider the case in which the output sequences are shorter than the input sequences. The more general case is an extension of this model (using "empty" transitions that do not "take" any time) and will be discussed elsewhere. As in HMMs and IOHMMs, we introduce a discrete *hidden state* variable, q_t, which will allow us to simplify the distribution $P(y_1^S|x_1^T)$ by using Markovian independence assumptions. The state sequence q_1^T is taken to be synchronous with the input sequence x_1^T.

In order to produce output sequences shorter than input sequences, *states may sometimes not emit an output symbol*, but instead would "emit" a null symbol ϵ. Therefore, there exist many sequences of states corresponding to the same output sequence, and a given sequence of states can correspond to output sequences of different lengths.

When conceived as a generative model of the output (given the input), an asynchronous IOHMM works as follows. At time $t = 0$, an initial state q_0 is chosen according to the distribution $P(q_0)$, and the length of the output sequence s is initialized to 0. At other time steps $t > 0$, a state q_t is first picked according to the *transition distribution* $P(q_t|q_{t-1}, x_t)$, using the state at the previous time step q_{t-1} and the current input x_t. A decision is then taken as to wether or not an output y_s will be produced at time t or not, according to the *emit-or-not distribution*. In the positive case, An output y_s is then produced according to the *emission distribution* $P(y_s|q_t, x_t)$. The length of the output sequence is increased from $s-1$ to s, and the s^{th} output y_s is emitted with probability $P(y_s|q_t, x_t)$. The parameters of the model are thus the initial state probabilities, $\pi(i) = P(q_0 = i)$, and the parameters of the emit-or-not, emission and transition conditional distribution models, $P(emit - or - not|q_t, x_t)$, $P(y_s|q_t, x_t)$ and $P(q_t|q_{t-1}, x_t)$. Since the input and output sequences are of different lengths, we will introduce another hidden variable, τ_t, specifically to represent the alignment between inputs and outputs, with $\tau_t = s$ meaning that s outputs have been emitted at time t.

Let us first formalize the independence assumptions and the form of the conditional distribution represented by the model. The conditional probability $P(y_1^S|x_1^T)$ can be written as a sum of terms $P(y_1^S, q_0^T, \tau_1^T|x_1^T)$ over all possible state sequences q_0^T such that the number of emitting states in each of these sequences is S (the length of the output sequence):

$$P(y_1^S|x_1^T) = \sum_{q_0^T, \tau_T = S} P(y_1^S, q_0^T, \tau_1^T|x_1^T) \qquad (1)$$

All S outputs must have been emitted by time T, so $\tau_T = S$. The hidden state q_t takes discrete values in a finite set. Each of the terms $P(y_1^S, q_0^T, \tau_1^T|x_1^T)$ corresponds to a particular sequence of states, and a corresponding alignment.

We summarize in table 1 the notation we have introduced and define additional notation used in this paper.

The Markovian conditional independence assumptions in this model mean that the state variable q_t summarizes sufficiently the past of the sequence, so

$$P(q_t|q_1^{t-1}, x_1^t) = P(q_t|q_{t-1}, x_t) \qquad (2)$$

and

$$P(y_s|q_1^t, x_1^t) = P(y_s|q_t, x_t). \qquad (3)$$

These assumptions are analogous to the Markovian independence assumptions used in HMMs and are essentially the same as in synchronous IOHMMs. Based on these two assumptions, the conditional probability can be efficiently represented and computed recursively, using an intermediate variable

$$\alpha(i, s, t) \stackrel{\text{def}}{=} P(q_t{=}i, \tau_t{=}s, y_1^s|x_1^t). \qquad (4)$$

The conditional probability of an output sequence, can be expressed in terms of this variable:

$$L \stackrel{\text{def}}{=} P(y_1^S|x_1^T) = \sum_{i \in F} \alpha(i, S, T) \qquad (5)$$

Table 1: **Notation used in the paper**

- S = size of the output sequence.
- T = size of the input sequence.
- N = number of states in the IOHMM.
- $a(i,j,t)$ = output of the module that computes $P(q_t{=}i|q_{t-1}{=}j, x_t)$
- $b(i,l,t)$ = output of the module that computes $P(y_s = l|q_t{=}i, x_t, \tau_t = s, \tau_{t-1} = s-1)$, the probability to emit the symbol l at time t in state i givent that state i emits at time t.
- $\epsilon(i,t)$ = output of the module that computes $P(\tau_t = s|\tau_{t-1} = s, q_t = i)$, the probability not to emit at time t in state i.
- $\pi(i) = P(q_0{=}i)$, initial probability of state i.
- $z_{i,t} = 1$ if $q_t = i$; $z_{i,t} = 0$ otherwise. These indicator variables give the state sequence.
- $m_{s,t} = 1$ if the system emits the s^{th} output at time t, $m_{s,t} = 0$ otherwise. These indicator variables give the input/output alignment.
- $\epsilon_{i,t} = 1$ means that state i did not emit at time t.
- $\tau_t = s$ means that the first s first outputs have been emitted at time t.
- $\sigma_{t,k} = 1$ if the $t^{\rm th}$ input symbol is k, $\sigma_{t,k} = 0$ otherwise.
- $\gamma_{s,k} = 1$ if the $s^{\rm th}$ output symbol is k, $\gamma_{s,k} = 0$ otherwise.
- $pred(i)$ is the set of all the predecessors states of state i.
- $succ(i)$ is the set of all the successors states of state i.

where F is a set of final states. These α's can be computed recursively in a way that is analogous to the forward pass of the Baum-Welsh algorithm for HMMs:

$$\begin{aligned}\alpha(i,s,t) \;=\; & \\ & b(i,y_s,t)(1-(\epsilon(i,t)) \sum_{j\in pred(i)} a(i,j,t)\,\alpha(j,s-1,t-1) \\ & + \epsilon(i,t) \sum_{j\in pred(i)} a(i,j,t)\,\alpha(j,s,t-1) \end{aligned} \tag{6}$$

where $a(i,j,t) = P(q_t{=}i|q_{t-1}{=}j,x_t)$, $b(i,y_s,t) = P(y_s|q_t{=}i,x_t,\tau_t{=}s,\tau_{t-1}{=}s-1)$, $\epsilon(i,t) = P(\tau_t{=}s|\tau_{t-1}{=}s,q_t{=}i,x_t)$ and $pred(i)$ is the set of states with an outgoing transition to state i. The derivation of this recursion using the independence assumptions can be found in the appendix.

3 An EM Algorithm for Asynchronous IOHMMs

The learning algorithm we propose is based on the maximum likelihood principle, i.e., here, maximizing the conditional likelihood of the training data. The algorithm could be generalized to one for maximizing the likelihood of the parameters given the data, by taking into account priors on those parameters. Let the training data, D, be a set of P input/output sequences independently sampled from the same distribution. Let T_p and S_p be the lengths of the $p^{\rm th}$ input and output sequence respectively:

$$D \stackrel{\rm def}{=} \{(x_1^{T_p}(p), y_1^{S_p}(p)); p{=}1\ldots P\} \tag{7}$$

Let Θ be the set of all the parameters of the model. Because each sequence is sampled independently, we can write the likelihood function as follows, omitting sequence indexes to simplify:

$$L(\Theta; D) = \prod_{p=1}^{P} P(y_1^{S_p}|x_1^{T_p};\Theta) \tag{8}$$

According to the maximum likelihood principle, the optimal parameters Θ are obtained when $L(\Theta; D)$ is maximized. We will show here how an iterative optimization to a local maximum can be achieved using the Expectation-Maximization (EM) algorithm.

EM is an iterative procedure for maximum likelihood estimation, originally formalized in [Dempster et al., 1977]. Each iteration is composed of two steps: an estimation step and a maximization step. The basic idea is to introduce an additional variable, q, which, if it were known, would greatly simplify the optimization problem. This additional variable is known as the missing or hidden data. A joint model of q with the observed variables must be set up. The set D_c which includes the data set D and values of the variable q for each of the examples, is known as the *complete data set.* Correspondingly, $L_c(\Theta; D_c)$ is referred as the *complete data likelihood.* Since q is not observed, L_c is a random variable and cannot be maximized directly. The EM algorithm is based on averaging $logL_c(\Theta; D_c)$ over the distribution of q, given the known data D and the previous value of the parameters $\hat{\Theta}$. This expected log likelihood is called the auxiliary function:

$$Q(\Theta;\hat{\Theta}) = E_q\left[logL_c(\Theta;D_c)|D,\hat{\Theta}\right] \tag{9}$$

In short, for each EM iteration, one first computes Q (E-step), then updates Θ such that it maximizes Q (M-step).

To apply EM to asynchronous IOHMMs, we need to choose hidden variables such that the knowledge of these variables would simplify the learning problem. Let q be a hidden variable representing the hidden state of the Markov model (such that $q_t = i$ means the system is in state i at time t). Knowledge of the state sequence q_1^T would make the estimation of parameters trivial (simple counting would suffice). Because states sometimes emit and sometime do not, we introduce an additional hidden variable, τ, representing the alignment between inputs (and states) and outputs, such that $\tau_t = s$ implies that s outputs have been emitted at time t.

Here, the *complete data* can be written as follows:

$$D_c \stackrel{\text{def}}{=} \{(x_1^{T_p}(p), y_1^{S_p}(p), q_1^{T_p}(p), \tau_1^{T_p}(p)); p{=}1\ldots P\} \tag{10}$$

The corresponding complete data likelihood is (again dropping (p) indices):

$$L_c(\Theta;D_c) = \prod_{p=1}^{P} P(y_1^{S_p}, q_1^{T_p}, \tau_1^{T_p}|x_1^{T_p};\Theta) \tag{11}$$

Let $z_{i,t}$ be an indicator variable such that $z_{i,t} = 1$ if $q_t = i$, and $z_{i,t} = 0$ otherwise. Let $m_{s,t}$ be an indicator variable such that $m_{s,t} = 1$ if the s^{th} output is emitted at time t, and $m_{s,t} = 0$ otherwise. Let $\epsilon_{i,t} = 1$ if state i do not emit at time t, and $\epsilon_{i,t} = 0$ otherwise. Using these indicator variables and factorizing the likelihood equation, we obtain:

$$\begin{aligned} L_c(\Theta;D_c) \;=\; & \prod_{p=1}^{P}\prod_{t=1}^{T_p}\prod_{i=1}^{N} \\ & \left(\prod_{s=1}^{S_p} P(y_s|q_t{=}i, x_t, \tau_t{=}s, \tau_{t-1}{=}s{-}1)^{z_{i,t}m_{s,t}(1-\epsilon_{i,t})}\right) \cdot \\ & \left(\prod_{s=1}^{S_p} P(\tau{=}s|q_t{=}i, x_t, \tau_{t-1}{=}s{-}1)^{z_{i,t}m_{s,t}(1-\epsilon_{i,t})}\right) \cdot \\ & \left(\prod_{s=1}^{S_p} P(\tau{=}s|q_t{=}i, x_t, \tau_{t-1}{=}s)^{z_{i,t}m_{s,t}\epsilon_{i,t}}\right) \cdot \\ & \left(\prod_{j=1}^{N} P(q_t{=}i|q_{t-1}{=}j, x_t)^{z_{i,t}z_{j,t-1}}\right) \end{aligned} \tag{12}$$

Taking the logarithm we obtain the following expression for the complete data log likelihood:

$$\begin{aligned} logL_c(\Theta;D_c) \;=\; & \sum_{p=1}^{P}\sum_{t=1}^{T_p}\sum_{i=1}^{N} \\ & \left(\sum_{s=1}^{S_p} z_{i,t}m_{s,t}(1-\epsilon_{i,t}) \log P(y_s|q_t{=}i, x_t, \tau_t{=}s, \tau_{t-1}{=}s{-}1)\right) + \\ & \left(\sum_{s=1}^{S_p} z_{i,t}m_{s,t}(1-\epsilon_{i,t}) \log P(\tau_t{=}s|q_t{=}i, x_t, \tau_{t-1}{=}s{-}1)\right) + \\ & \left(\sum_{s=1}^{S_p} z_{i,t}m_{s,t}\epsilon_{i,t} \log P(\tau_t{=}s|q_t{=}i, x_t, \tau_{t-1}{=}s)\right) + \\ & \left(\sum_{j=1}^{N} z_{i,t}z_{j,t-1} \log P(q_t{=}i|q_{t-1}{=}j, x_t)\right) \end{aligned} \tag{13}$$

3.1 The Estimation Step

Let us define the auxiliary function $Q(\Theta;\hat{\Theta})$ as the expected value of $logL_c(\Theta;D_c)$ with respect to the hidden variables q and τ, given the data D and the previous set of parameters $\hat{\Theta}$:

$$Q(\Theta;\hat{\Theta}) = E_{q,\tau}\left[\log L_c(\Theta;D_c)|D,\hat{\Theta}\right] \tag{14}$$

$$\begin{aligned} Q(\Theta;\hat{\Theta}) \;=\; & \sum_{p=1}^{P}\sum_{t=1}^{T_p}\sum_{i=1}^{N} \\ & \left(\sum_{s=1}^{S_p} \hat{g}_{i,s,t} \log P(y_s|q_t{=}i, x_t, \tau_t{=}s, \tau_{t-1}{=}s{-}1)\right) + \\ & \left(\sum_{s=1}^{S_p} \hat{g}_{i,s,t} \log P(\tau_t{=}s|q_t{=}i, x_t, \tau_{t-1}{=}s{-}1)\right) + \\ & \left(\sum_{s=1}^{S_p} \hat{f}_{i,s,t} \log P(\tau_t{=}s|q_t{=}i, x_t, \tau_{t-1}{=}s)\right) + \\ & \left(\sum_{j=1}^{N} \hat{h}_{i,j,t} \log P(q_t{=}i|q_{t-1}{=}j, x_t)\right) \end{aligned} \tag{15}$$

where, by definition,

$$\hat{g}_{i,s,t} \stackrel{\text{def}}{=} E_{q,\tau}[q_t{=}i, \tau_t{=}s|\tau_{t-1}{=}s{-}1, x_1^T, y_1^S;\Theta] \tag{16}$$

$$\hat{f}_{i,s,t} \stackrel{\text{def}}{=} E_{q,\tau}[q_t{=}i, \tau_t{=}s|\tau_{t-1}{=}s, x_1^T, y_1^S;\Theta] \tag{17}$$

and

$$\hat{h}_{i,j,t} \stackrel{\text{def}}{=} E_{q}[q_t{=}i, q_{t-1}{=}j|x_1^T, y_1^S;\Theta] \tag{18}$$

The $\hat{}$ on $\hat{f}$, $\hat{g}$ and $\hat{h}$ denote that these expectations are computed using the previous value of the parameters, $\hat{\Theta}$. In order to compute $\hat{f}_{i,s,t}$, $\hat{g}_{i,s,t}$ and $\hat{h}_{i,j,t}$, we will use the already defined $\alpha(i,s,t)$ (equations 4 and 6), and introduce a new variable, $\beta(i,s,t)$, borrowing the notation from the HMM and IOHMM literature:

$$\beta(i,s,t) \stackrel{\text{def}}{=} P(y_{s+1}^{S}|q_t{=}i, \tau_t{=}s, x_{t+1}^{T}) \tag{19}$$

Like α, β can be computed recursively, but going backwards in time:

$$\begin{aligned}\beta(i,s,t) \;=\; & \\ \sum_{j\in succ(i)} & \left(\begin{array}{l} a(j,i,t{+}1)b(j,y_{s+1},t{+}1)\cdot \\ (1-\epsilon(i,t{+}1))\beta(j,s{+}1,t{+}1) \end{array} \right) \\ + \sum_{j\in succ(i)} & (a(j,i,t{+}1)\epsilon(j,t{+}1)\beta(j,s,t{+}1)) \end{aligned} \tag{20}$$

where $pred(i)$ is the set of predecessor states of state i and $succ(i)$ is the set of successor states of state i, $a(j,i,t)$ is the conditional transition probability from state i to state j at time t, $b(i,y_s,t)$ is the conditional probability to emit the s^{th} output at time t in state i given that this state emits at time t, and $\epsilon(i,t)$ is the probability not to emit at time t in state i. The proof of correctness of this recursion (using the Markovian independence assumptions) is given in the appendix.

Let $\alpha^0(i,s,t)$ be the part of $\alpha(i,s,t)$ computed when state i emits at time t:

$$\begin{aligned}\alpha^0(i,s,t) \;=\; & b(i,y_s,t)(1-\epsilon(i,t))\cdot \\ & \sum_{j\in pred(i)} a(i,j,t)\,\alpha(j,s{-}1,t{-}1)\end{aligned} \tag{21}$$

Similarly, $\alpha^1(i,s,t)$ is the part of $\alpha(i,s,t)$ computed when state i does not emit at time t:

$$\begin{aligned}\alpha^1(i,s,t) \;=\; & \epsilon(i,t)\cdot \\ & \sum_{j\in pred(i)} a(i,j,t)\,\alpha(j,s,t{-}1)\end{aligned} \tag{22}$$

We can now express $g_{i,s,t}$, $f_{i,s,t}$ and $h_{i,j,t}$ in terms of $\alpha^0(i,s,t)$, $\alpha^1(i,s,t)$ and $\beta(i,s,t)$ (see derivations in the appendix):

$$\begin{aligned} h_{i,j,t} \;=\; & \frac{a(i,j,t)}{L}\cdot \\ & \left(\begin{array}{l} \displaystyle\sum_{s=1}^{S} \left(\begin{array}{l} \alpha(j,s{-}1,t{-}1)b(i,y_s,t)\cdot \\ (1-\epsilon(i,t))\beta(i,s,t) \end{array}\right) \\ \displaystyle +\sum_{s=0}^{S} \alpha(j,s,t{-}1)\epsilon(i,t)\beta(i,s,t) \end{array} \right) \end{aligned} \tag{23}$$

$$g_{i,s,t} = \frac{\alpha^0(i,s,t)\beta(i,s,t)}{L} \tag{24}$$

and

$$f_{i,s,t} = \frac{\alpha^1(i,s,t)\beta(i,s,t)}{L} \tag{25}$$

3.2 The Maximization Step

After each estimation step, one has to maximize $Q(\Theta;\hat{\Theta})$. If the conditional probability distributions (for transitions as well as emissions) have a simple enough form (e.g. multinomial, generalized linear models, or mixtures of these) then one can maximize analytically Q, i.e., solve

$$\frac{\partial Q(\Theta;\hat{\Theta})}{\partial\Theta} = 0 \tag{26}$$

for Θ. Otherwise, if for instance conditional probabilities are implemented using non-linear systems (such as a neural network), then maximization of Q cannot be done in one step. In this case, we can apply a Generalized EM (GEM) algorithm, that simply requires an increase in Q at each optimization step, for example using gradient ascent in Q.

3.3 Multinomial Distributions

We describe here the maximization procedure for multinomial conditional probabilities (for transitions and emissions), i.e., which can be implemented as lookup tables. This applies to problems with discrete inputs and outputs. Let M_i be the number of input symbols, M_o the number of output classes, $\sigma_{t,k} = 1$ when the t^{th} input symbol is k, and $\sigma_{t,k} = 0$ otherwise. Also, let $\gamma_{s,k} = 1$ when the s^{th} output symbol is k, and $\gamma_{s,k} = 0$ otherwise.

For transition probabilities, let $w_{i,j,k} = P(q_t{=}i|q_{t-1}{=}j, x_{k,t}{=}1)$. The solution of equation (26) for $w_{i,j,k}$, with the constraints that transition probabilities must sum to 1, yields the following reestimation formula:

$$w_{i,j,k} = \frac{\displaystyle\sum_{t=1}^{T} \sigma_{t,k}\hat{h}_{i,j,t}}{\displaystyle\sum_{l=1}^{N}\sum_{t=1}^{T} \sigma_{t,k}\hat{h}_{l,j,t}} \tag{27}$$

For emission probabilities, let $\omega_{i,l,k} = P(y{=}l|q_t{=}i, x_{k,t}{=}1)$, with the constraint $\omega_{i,\epsilon,k} +$

$\sum_{l \neq \epsilon} \omega_{i,l,k} = 1$, then the reestimation formulae are the following:

$$\omega_{i,l,k} = \frac{\sum_{t=1}^{T} \sum_{s=1}^{S} \sigma_{t,k} \gamma_{s,l} \hat{g}_{i,s,t}}{\sum_{m=1}^{M_o} \sum_{t=1}^{T} \sum_{s=1}^{S} \sigma_{t,k} \gamma_{s,m} \hat{g}_{i,s,t}} \quad (28)$$

For emit-or-not probabilities, let $\psi_{i,0,k} = P(\tau_t{=}s|\tau_{t-1}{=}s-1, q_t{=}i, x_{k,t}{=}1)$ and $\psi_{i,1,k} = P(\tau_t{=}s|\tau_{t-1}{=}s, q_t{=}i, x_{k,t}{=}1)$, with the constraint that $\psi_{i,0,k} + \psi_{i,1,k} = 1$.

$$\psi_{i,0,k} = \frac{\sum_{t=1}^{T} \sum_{s=1}^{S} \sigma_{t,k} (1 - \epsilon_{i,t}) \hat{g}_{i,s,t}}{\sum_{t=1}^{T} \sum_{s=1}^{S} \sigma_{t,k} (\hat{g}_{i,s,t} + \hat{f}_{i,s,t})} \quad (29)$$

$$\psi_{i,1,k} = \frac{\sum_{t=1}^{T} \sum_{s=1}^{S} \sigma_{t,k} \epsilon_{i,t} \hat{f}_{i,s,t}}{\sum_{t=1}^{T} \sum_{s=1}^{S} \sigma_{t,k} (\hat{g}_{i,s,t} + \hat{f}_{i,s,t})} \quad (30)$$

3.4 Neural Networks or Other Complex Distributions

In the more general case where one cannot maximize analytically Q, we can compute the gradient for each parameter and apply gradient ascent in Q, yielding a GEM algorithm (or alternatively, directly maximize L by gradient ascent). For transition probability models $a(j,i,t) = P(q_t{=}j|q_{t-1}{=}i, x_t; w_i)$ for state i, with parameters w_i, the gradient of Q with respect to w_i is

$$\frac{\partial Q(\Theta;\hat{\Theta})}{\partial w_i} = \sum_{t=1}^{T} \sum_{j \in succ(i)} \left(\frac{\hat{h}_{j,i,t}}{a(j,i,t)} \frac{\partial a(j,i,t)}{\partial w_i} \right) \quad (31)$$

where $\frac{\partial a(j,i,t)}{\partial w_i}$ can be computed by back-propagation.

Similarly, for emission probability models $b(i, y_s, t) = P(y_s|q_t{=}i, x_t, \tau_t{=}s, \tau_{t-1}{=}s-1; \omega_i)$ with parameters ω_i, the gradient is

$$\frac{\partial Q(\Theta;\hat{\Theta})}{\partial \omega_i} = \sum_{t=1}^{T} \sum_{s=1}^{S} \left(\frac{\hat{g}_{i,s,t}}{b(i,y_s,t)} \frac{\partial b(i,y_s,t)}{\partial \omega_i} \right) \quad (32)$$

where, again, $\frac{\partial b(i,y_s,t)}{\partial \omega_i}$ can be computed by back-propagation.

Table 2: Overview of the learning algorithm for asynchronous IOHMMs

1. Estimation Step: **for each** training sequence (x_1^T, y_1^S) **do**
   (a) **for each** state $j \leftarrow 1 \ldots n$ **do**
      - compute $a(i,j,t)$, $b(j,y_s,t)$ and $\epsilon(j,t)$ according to the chosen distribution models.

   (b) **for each** state $i \leftarrow 1 \ldots n$ **do**
      - compute $\alpha_{i,s,t}$, $\beta_{i,s,t}$, and L using the current value of the parameters $\hat{\Theta}$ (equations 6, 5 and 20).
      - compute the posterior probabilities $\hat{h}_{i,j,t}$, $\hat{g}_{i,s,t}$ and $\hat{f}_{i,s,t}$ (equations 24, 23 and 25).
2. Maximization Step: **for each** state $j \leftarrow 1 \ldots n$ **do**
   (a) Adjust the transition probability parameters of state j using reestimation formulae such as equation 27 (or gradient ascent for non-linear modules, equation 31).
   (b) Adjust the emission probability parameters of state j using reestimation formulae such as equation 28 (or gradient ascent for non-linear modules, equation 32).
   (c) Adjust the emit-or-not probability parameters of state j using reestimation formulae such as equations 29 and 30 (or gradient ascent for non-linear modules, equation 33).

Finally, for emit-or-not probability models $\epsilon(i,t) = P(\tau_t{=}s|q_t{=}i, x_t, \tau_{t-1}{=}s; \psi_i)$ with parameters ψ_i, the gradient is

$$\frac{\partial Q(\Theta;\hat{\Theta})}{\partial \psi_i} = \sum_{t=1}^{T} \sum_{s=1}^{S} \left(\frac{\hat{f}_{i,s,t}}{\epsilon(i,t)} \frac{\partial \epsilon(i,t)}{\partial \psi_i} \right) + \sum_{t=1}^{T} \sum_{s=1}^{S} \left(\frac{\hat{g}_{i,s,t}}{(1-\epsilon(i,t))} \frac{\partial (1-\epsilon(i,t))}{\partial \psi_i} \right) \quad (33)$$

4 A Recognition Algorithm for Asynchronous IOHMMs

Given a trained asynchronous IOHMM, we want to recognize new sequences, i.e., given an input sequence, choose an output sequence according to the model. Ideally, we would like to pick the output sequence that is most likely, given the input sequence.

However, this would require an exponential number of computations (with respect to sequence length). Instead, like in the Viterbi algorithm for HMMs, we will consider the complete data model, and look for the joint values of states and outputs that is most likely. Thanks to a dynamic programming recurrence, we can compute the most likely state and output sequence in time that is proportional to the sequence length times the number of transitions (even though the number of such sequences is exponential in the sequence length).

Let us define $V(i,t)$ as the probability of the best state and output subsequence ending up in state i at time t:

$$V(i,t) = \max_{s, y_1^s, q_1^{t-1}} P(y_1^s, q_1^{t-1}, q_t{=}i, \tau_t{=}s | x_1^t) \tag{34}$$

where the maximum is taken over all possible lengths s of output sequences y_1^s. This variable can be computed recursively by dynamic programming (the derivation is given in the appendix):

$$\begin{aligned} V(i,t) \quad = \quad & \max(\epsilon(i,t), (1-\epsilon(i,t)) \max_l b(i,l,t)) \\ & \max_j (a(i,j,t) V(j,t{-}1)) \end{aligned} \tag{35}$$

At the end of the sequence, the best final state i^* which maximizes $V(i,T)$ is picked within the set of final states F. If the argmax in the above recurrence is kept, than the best predecessor j and best output (y_s or the empty symbol ϵ) for each (i,t) can be used to trace back the optimal state and output sequence from i^*, like in the Viterbi algorithm.

5 Conclusion

We have presented a novel model and training algorithm for representing conditional distributions of output sequences given input sequences of a different length. The distribution is simplified by introducing hidden variables for the state and the alignment of inputs and outputs, similarly to HMMs. The output sequence distribution is decomposed into conditional emission distributions for individual outputs (given a state and an input at time t) and conditional transition distributions (given a previous state and an input at time t). This is an extension of the already proposed IOHMMs [Bengio and Frasconi, 1996, Bengio and Frasconi, 1995] that allows input and output sequences to be asynchronous.

The parameters of the model can be estimated with an EM or GEM algorithm (depending on the form of the emission and transition distributions). Both the E-step and the M-step can be performed in time at worst proportional to the product of the lengths of the input and output sequences, times the number of transitions. A recognition algorithm similar to the Viterbi algorithm for HMMs has also been presented, which takes in the worse case time proportional to the length of the input sequence times the number of transitions.

In practice (especially when the number of states is large), both training and recognition can be sped up by using search algorithms (such as beam search) in the space of state sequences.

References

[Bengio and Frasconi, 1995] Bengio, Y. and Frasconi, P. (1995). An input/output HMM architecture. In Tesauro, G., Touretzky, D., and Leen, T., editors, *Advances in Neural Information Processing Systems 7*, pages 427–434. MIT Press, Cambridge, MA.

[Bengio and Frasconi, 1996] Bengio, Y. and Frasconi, P. (1996). Input/Output HMMs for sequence processing. *to appear in IEEE Transactions on Neural Networks*.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum-likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society B*, 39:1–38.

[Levinson et al., 1983] Levinson, S., Rabiner, L., and Sondhi, M. (1983). An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. *Bell System Technical Journal*, 64(4):1035–1074.

[Pereira et al., 1994] Pereira, F., Riley, M., and Sproat, R. (1994). Weighted rational transductions and their application to human language processing. In *ARPA Natural Language Processing Workshop*.

[Rabiner, 1989] Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

[Rumelhart et al., 1986] Rumelhart, D., Hinton, G., and Williams, R. (1986). Learning internal representations by error propagation. In Rumelhart, D. and McClelland, J., editors, *Parallel Distributed Processing*, volume 1, chapter 8, pages 318–362. MIT Press, Cambridge.

[Singer, 1996] Singer, Y. (1996). Adaptive mixtures of probabilistic transducers. In Mozer, M., Touretzky, D., and Perrone, M., editors, *Advances in Neural Information Processing Systems 8*. MIT Press, Cambridge, MA.

A Derivations for the EM algorithm

A.1 Recursion for $\alpha(i,s,t)$

$\alpha(i,s,t)$ represents the probability to be in state i at time t after having emitted the first s outputs:

$$\alpha(i,s,t) = P(q_t{=}i, \tau_t{=}s, y_1^s | x_1^t) \tag{36}$$

This probability can be decomposed into a sum of two probabilities: one for the case where i is an emitting state, and one for the other case. Then, each probability can be replaced by a sum over all the possible previous states that lead to this state.

$$\begin{aligned}
&\alpha(i,s,t)\\
&= P(q_t{=}i, \tau_t{=}s, \tau_{t-1}{=}s{-}1, y_1^s|x_1^t)\\
&\quad + P(q_t{=}i, \tau_t{=}s, \tau_{t-1}{=}s, y_1^s|x_1^t)\\
&= \sum_{j\in pred(i)} P(q_t{=}i, q_{t-1}{=}j, \tau_t{=}s, \tau_{t-1}{=}s{-}1, y_1^s|x_1^t)\\
&\quad + \sum_{j\in pred(i)} P(q_t{=}i, q_{t-1}{=}j, \tau_t{=}s, \tau_{t-1}{=}s, y_1^s|x_1^t)\\
&= \sum_{j\in pred(i)} P(q_t{=}i, q_{t-1}{=}j, \tau_t{=}s, \tau_{t-1}{=}s{-}1, y_1^s|x_1^t)\\
&\quad + \sum_{j\in pred(i)} P(q_t{=}i, q_{t-1}{=}j, \tau_t{=}s, \tau_{t-1}{=}s, y_1^s|x_1^t)\\
&= \sum_{j\in pred(i)} \begin{pmatrix} P(q_t{=}i|q_{t-1}{=}j, x_t)\cdot \\ P(y_s|q_t{=}i, x_t, \tau_t{=}s, \tau_{t-1}{=}s{-}1)\cdot \\ P(\tau_t{=}s|, \tau_{t-1} = s{-}1, q_t{=}i)\cdot \\ P(y_1^{s-1}, q_{t-1}{=}j, \tau_{t-1}{=}s{-}1|x_1^{t-1}) \end{pmatrix}\\
&\quad + \sum_{j\in pred(i)} \begin{pmatrix} P(q_t{=}i|q_{t-1}{=}j, x_t)\cdot \\ P(\tau_t{=}s|, \tau_{t-1} = s, q_t{=}i)\cdot \\ P(y_1^s, q_{t-1}{=}j, \tau_{t-1}{=}s|x_1^{t-1}) \end{pmatrix}\\
&= \sum_{j\in pred(i)} a(i,j,t) b(i,y_s,t)(1{-}\epsilon(i,t))\alpha(j,s{-}1,t{-}1)\\
&\quad + \sum_{j\in pred(i)} a(i,j,t)\epsilon(i,t)\alpha(j,s,t{-}1)
\end{aligned} \tag{37}$$

which leads to:

$$\begin{aligned}
\alpha(i,s,t) \;=\;& b(i,y_s,t)(1{-}\epsilon(i,t)) \sum_{j\in pred(i)} a(i,j,t)\alpha(j,s{-}1,t{-}1)\\
&+\epsilon(i,t) \sum_{j\in pred(i)} a(i,j,t)\alpha(j,s,t{-}1)
\end{aligned} \tag{38}$$

Initialization of these $\alpha(i,s,t)$ is done as follows:

$$\alpha(i,0,0) = \pi(i) \tag{39}$$

where $\pi(i) = P(q_0{=}i)$ is the prior initial probability to be in state i.

A.2 Recursion for $\beta(i,s,t)$

$\beta(i,s,t)$ represents the probability of emitting the last $S-s$ outputs, starting from state i at time t:

$$\beta(i,s,t) = P(y_{s+1}^S | q_t{=}i, \tau_t{=}s, x_{t+1}^T) \tag{40}$$

This probability can be rewritten as the sum over all the possible predecessors of state i, and then each term can be split into the emitting case and the non-emitting case:

$$\begin{aligned}
&\beta(i,s,t)\\
&= \sum_{j\in succ(i)} P(y_{s+1}^S, q_{t+1}{=}j | q_t{=}i, \tau_t{=}s, x_{t+1}^T)\\
&= \sum_{j\in succ(i)} P(y_{s+1}^S, q_{t+1}{=}j, \tau_{t+1}{=}s{+}1 | q_t{=}i, \tau_t{=}s, x_{t+1}^T)\\
&\quad + \sum_{j\in succ(i)} P(y_{s+1}^S, q_{t+1}{=}j, \tau_{t+1}{=}s | q_t{=}i, \tau_t{=}s, x_{t+1}^T)\\
&= \sum_{j\in succ(i)} \begin{pmatrix} P(q_{t+1}{=}j|q_t{=}i, x_{t+1})\cdot \\ P(y_{s+1}|\tau_{t+1} = s{+}1, \tau_t = s, q_{t+1}{=}j, x_{t+1})\cdot \\ P(\tau_{t+1}{=}s{+}1|\tau_t{=}s, q_{t+1}{=}j)\cdot \\ P(y_{s+2}^S|q_{t+1}{=}j, \tau_{t+1}{=}s{+}1, x_{t+1}^T) \end{pmatrix}\\
&\quad + \sum_{j\in succ(i)} \begin{pmatrix} P(q_{t+1}{=}j|q_t{=}i, x_{t+1})\cdot \\ P(\tau_{t+1}{=}s|\tau_t{=}s, q_{t+1}{=}j)\cdot \\ P(y_{s+1}^S|q_{t+1}{=}j, \tau_{t+1}{=}s, x_{t+1}^T) \end{pmatrix}\\
&= \sum_{j\in succ(i)} a(j,i,t{+}1) b(j,y_{s+1},t{+}1)(1{-}\epsilon(j,t{+}1))\beta(j,s{+}1,t{+}1)\\
&\quad + \sum_{j\in succ(i)} a(j,i,t{+}1)\epsilon(j,t{+}1)\beta(j,s,t{+}1)
\end{aligned} \tag{41}$$

Hence,

$$\begin{aligned}
\beta(i,s,t) \;=\;& \\
&\sum_{j\in succ(i)} a(j,i,t{+}1) b(j,y_{s+1},t{+}1)(1{-}\epsilon(j,t{+}1))\beta(j,s{+}1,t{+}1)\\
&+ \sum_{j\in succ(i)} a(j,i,t{+}1)\epsilon(j,t{+}1)\beta(j,s,t{+}1)
\end{aligned} \tag{42}$$

Finally, initialization of $\beta(i,s,t)$ is done as follows: if i is a final state, $\beta(i,s,t) = 1$, else $\beta(i,s,t) = 0$.

A.3 Computation of $h_{i,j,t}$

$h_{i,j,t}$ represents the probability of going through the transition from state i to state j at time t, given the input and output sequences:

$$\begin{aligned} h_{i,j,t} &= P(q_t{=}i, q_{t-1}{=}j|x_1^T, y_1^S) \\ &= \frac{P(y_1^S, q_t{=}i, q_{t-1}{=}j|x_1^T)}{P(y_1^S|x_1^T)} \end{aligned} \tag{43}$$

and since $L = P(y_1^S|x_1^T)$

$$\begin{aligned} &P(y_1^S, q_t{=}i, q_{t-1}{=}j|x_1^T) = \\ &\sum_{s=1}^{S} P(y_1^S, q_t{=}i, q_{t-1}{=}j, \tau_t{=}s, \tau_{t-1} = s{-}1|x_1^T) \\ &\quad + \sum_{s=1}^{S} P(y_1^S, q_t{=}i, q_{t-1}{=}j, \tau_t{=}s, \tau_{t-1} = s|x_1^T) \\ &= \sum_{s=1}^{S} \left(\begin{array}{l} P(y_1^{s-1}, q_{t-1}{=}j, \tau_{t-1}{=}s{-}1|x_1^T)\cdot \\ P(q_t{=}i|q_{t-1}{=}j, x_1^T)\cdot \\ P(\tau_t = s|\tau_{t-1} = s{-}1, q_t{=}i, x_1^T)\cdot \\ P(y_s|q_t{=}i, \tau_t = s, \tau_{t-1} = s{-}1, x_t)\cdot \\ P(y_{s+1}^S|q_t{=}i, \tau_t{=}s, x_1^T) \end{array} \right) \\ &\quad + \sum_{s=1}^{S} \left(\begin{array}{l} P(y_1^s, q_{t-1}{=}j, \tau_{t-1}{=}s|x_1^T)\cdot \\ P(q_t{=}i|q_{t-1}{=}j, x_1^T)\cdot \\ P(y_{s+1}^S|q_t{=}i, \tau_t{=}s, x_1^T)\cdot \\ P(\tau_t = s|\tau_{t-1} = s, q_t{=}i, x_1^T) \end{array} \right) \\ &= \sum_{s=1}^{S} \alpha(j, s{-}1, t{-}1) a(i,j,t) b(i, y_s, t)(1{-}\epsilon(i,t))\beta(i,s,t) \\ &\quad + \sum_{s=1}^{S} \alpha(j, s, t{-}1) a(i,j,t)\epsilon(i,t)\beta(i,s,t) \end{aligned} \tag{44}$$

so,

$$\begin{aligned} h_{i,j,t} &= \\ &\frac{a(i,j,t)}{L} \sum_{s=1}^{S} \alpha(j, s{-}1, t{-}1) b(i, y_s, t)(1{-}\epsilon(i,t))\beta(i,s,t) \\ &+ \frac{a(i,j,t)}{L} \sum_{s=1}^{S} \alpha^1(j, s, t{-}1)\epsilon(i,t)\beta(i,s,t) \end{aligned} \tag{45}$$

A.4 Computation of $g_{i,s,t}$

$g_{i,s,t}$ represents the probability that the system is in state i at time t, that state i emits at time t, and that the system has already emitted the first s outputs:

$$\begin{aligned} g_{i,s,t} &= P(q_t{=}i, \tau_t{=}s|\tau_{t-1}{=}s{-}1, x_1^T, y_1^S) \\ &= \frac{P(y_1^S, q_t{=}i, \tau_t{=}s|\tau_{t-1}{=}s{-}1, x_1^T)}{P(y_1^S|x_1^T)} \\ &= \frac{1}{L}\cdot P(y_{s+1}^S|q_t{=}i, \tau_t{=}s, x_1^T)\cdot \\ &\quad P(y_1^s, q_t{=}i, \tau_t{=}s|\tau_{t-1}{=}s{-}1, x_1^T) \end{aligned}$$

so,

$$g_{i,s,t} = \frac{\alpha^0(i,s,t)\beta(i,s,t)}{L} \tag{46}$$

A.5 Computation of $f_{i,s,t}$

$f_{i,s,t}$ represents the probability that the system is in state i at time t, that state i do not emit at time t, and that the system has already emitted the first s outputs:

$$\begin{aligned} f_{i,s,t} &= P(q_t{=}i, \tau_t{=}s|\tau_{t-1}{=}s, x_1^T, y_1^S) \\ &= \frac{P(y_1^S, q_t{=}i, \tau_t{=}s|\tau_{t-1}{=}s, x_1^T)}{P(y_1^S|x_1^T)} \\ &= \frac{1}{L}\cdot P(y_{s+1}^S|q_t{=}i, \tau_t{=}s, x_1^T)\cdot \\ &\quad P(y_1^s, q_t{=}i, \tau_t{=}s|\tau_{t-1}{=}s, x_1^T) \end{aligned}$$

so,

$$f_{i,s,t} = \frac{\alpha^1(i,s,t)\beta(i,s,t)}{L} \tag{47}$$

B Reestimation Formulae for Multinomial Distributions

B.1 Transition Probabilities

Let us define each parameter of the lookup table as follows:

$$w_{i,j,k} = P(q_t{=}i|q_{t-1}{=}j, x_{k,t}{=}1) \tag{48}$$

These parameters should be positive and sum to 1 over i:

$$\sum_{i \in succ(j)} w_{i,j,k} = 1 \qquad \forall j = 1 \ldots N, \forall k = 1 \ldots M_i \tag{49}$$

To incorporate this constraint into the system, we can introduce a new term in the cost function:

$$J(\Theta;\hat{\Theta}) = Q(\Theta;\hat{\Theta}) + \sum_{j=1}^{N}\sum_{k=1}^{M_i}\left(1 - \sum_{i\in succ(j)} w_{i,j,k}\right)\lambda_{j,k} \tag{50}$$

where the $\lambda_{j,k}$ are Lagrange multipliers. Taking the derivatives of J with respect to the parameters, we find:

$$\frac{\partial J}{\partial w_{i,j,k}} = \sum_{t=1}^{T} \sigma_{t,k}\frac{\hat{h}_{i,j,t}}{w_{i,j,k}} - \lambda_{j,k} \tag{51}$$

This expression is equal to zero when

$$w_{i,j,k} = \frac{\sum_{t=1}^{T}\sigma_{t,k}\hat{h}_{i,j,t}}{\lambda_{j,k}} \tag{52}$$

With the probabilistic constraint, we obtain the reestimation formula:

$$w_{i,j,k} = \frac{\sum_{t=1}^{T}\sigma_{t,k}\hat{h}_{i,j,t}}{\sum_{l=1}^{N}\sum_{t=1}^{T}\sigma_{t,k}\hat{h}_{l,j,t}} \tag{53}$$

B.2 Emission Probabilities

For the emission probabilities, the parameters are

$$\omega_{i,l,k} = P(y{=}l|q_t{=}i, x_{k,t}{=}1, \tau_t{=}s, \tau_{t-1}{=}s{-}1). \tag{54}$$

These parameters should be positive and sum to 1 over l:

$$\sum_{l=1}^{M_o} \omega_{i,l,k} = 1 \qquad \forall i = 1 \ldots N, \forall k = 1 \ldots M_i. \tag{55}$$

Again using Lagrange multipliers, λ_k, to enforce this constraint:

$$J(\Theta;\hat{\Theta}) = Q(\Theta;\hat{\Theta}) + \sum_{i=1}^{N}\sum_{k=1}^{M_i}\left(1 - \sum_{l=1}^{M_o}\omega_{i,l,k}\right)\lambda_k \tag{56}$$

The derivatives of J with respect to the parameters are

$$\frac{\partial J}{\partial \omega_{i,l,k}} = \sum_{t=1}^{T}\sum_{s=1}^{S}\sigma_{t,k}\gamma_{s,l}\frac{\hat{g}_{i,s,t}}{\omega_{i,l,k}} - \lambda_k, \tag{57}$$

which is equal to zero when

$$\omega_{i,l,k} = \frac{\sum_{t=1}^{T}\sum_{s=1}^{S}\sigma_{t,k}\gamma_{s,l}\hat{g}_{i,s,t}}{\lambda_k} \tag{58}$$

With the probabilistic constraint, we obtain the reestimation formula:

$$\omega_{i,l,k} = \frac{\sum_{t=1}^{T}\sum_{s=1}^{S}\sigma_{t,k}\gamma_{s,l}\hat{g}_{i,s,t}}{\sum_{m=1}^{M_o}\sum_{t=1}^{T}\sum_{s=1}^{S}\sigma_{t,k}\gamma_{s,m}\hat{g}_{i,s,t}} \tag{59}$$

B.3 Emit or Not Probabilities

For the emit-or-not probabilities, the parameters are

$$\psi_{i,0,k} = P(\tau_t = s|q_t{=}i, x_{k,t}{=}1, \tau_{t-1}{=}s{-}1). \tag{60}$$

and

$$\psi_{i,1,k} = P(\tau_t = s|q_t{=}i, x_{k,t}{=}1, \tau_{t-1}{=}s). \tag{61}$$

These parameters should be positive and sum to 1 over l:

$$\psi_{i,0,k} + \psi_{i,1,k} = 1 \qquad \forall i = 1 \ldots N, \forall k = 1 \ldots M_i. \tag{62}$$

Again using Lagrange multipliers, λ_k, to enforce this constraint:

$$J(\Theta;\hat{\Theta}) = Q(\Theta;\hat{\Theta}) + \sum_{i=1}^{N}\sum_{k=1}^{M_i}\left(1 - (\psi_{i,0,k} + \psi_{i,1,k})\right)\lambda_k \tag{63}$$

The derivatives of J with respect to the parameters are

$$\frac{\partial J}{\partial \psi_{i,0,k}} = \sum_{t=1}^{T}\sum_{s=1}^{S}\sigma_{t,k}(1{-}\epsilon_{i,t})\frac{\hat{g}_{i,s,t}}{\psi_{i,0,k}} - \lambda_k, \tag{64}$$

and

$$\frac{\partial J}{\partial \psi_{i,1,k}} = \sum_{t=1}^{T}\sum_{s=1}^{S}\sigma_{t,k}\epsilon_{i,t}\frac{\hat{f}_{i,s,t}}{\psi_{i,1,k}} - \lambda_k, \tag{65}$$

which are equal to zero when

$$\psi_{i,0,k} = \frac{\sum_{t=1}^{T}\sum_{s=1}^{S}\sigma_{t,k}(1{-}\epsilon_{i,t})\hat{g}_{i,s,t}}{\lambda_k} \tag{66}$$

and

$$\psi_{i,1,k} = \frac{\sum_{t=1}^{T}\sum_{s=1}^{S}\sigma_{t,k}\epsilon_{i,t}\hat{f}_{i,s,t}}{\lambda_k} \tag{67}$$

With the probabilistic constraint, we obtain the reestimation formulae:

$$\psi_{i,0,k} = \frac{\sum_{t=1}^{T}\sum_{s=1}^{S}\sigma_{t,k}(1{-}\epsilon_{i,t})\hat{g}_{i,s,t}}{\sum_{t=1}^{T}\sum_{s=1}^{S}\sigma_{t,k}(\hat{f}_{i,s,t} + \hat{g}_{i,s,t})} \tag{68}$$

and

$$\psi_{i,1,k} = \frac{\sum_{t=1}^{T}\sum_{s=1}^{S}\sigma_{t,k}\epsilon_{i,t}\hat{f}_{i,s,t}}{\sum_{t=1}^{T}\sum_{s=1}^{S}\sigma_{t,k}(\hat{f}_{i,s,t} + \hat{g}_{i,s,t})} \tag{69}$$

C Recurrence for the Recognition Algorithm

Let us define $V(i,t)$ as follows:

$$V(i,t) = \max_{s,\tau,y_1^s,q_1^{t-1}} P(y_1^s, q_1^{t-1}, q_t{=}i, \tau_t{=}s|x_1^t) \tag{70}$$

Then

$$\begin{aligned}
&P(y_1^s, q_1^{t-1}, q_t{=}i, \tau_t{=}s|x_1^t)\\
&\quad= P(y_1^s, q_1^{t-1}, q_t{=}i, \tau_t{=}s, \tau_{t-1}{=}s{-}1|x_1^t)\\
&\qquad+P(y_1^s, q_1^{t-1}, q_t{=}i, \tau_t{=}s, \tau_{t-1}{=}s|x_1^t)\\
&\quad= \begin{pmatrix} P(y_s|q_t{=}i, x_t, \tau_t{=}s, \tau_{t-1}{=}s{-}1)\cdot \\ P(q_t{=}i|q_{t-1}, x_t)\cdot \\ P(\tau_t{=}s|\tau_{t-1}{=}s{-}1, q_t{=}i)\cdot \\ P(y_1^{s-1}, q_1^{t-1}, \tau_{t-1}{=}s{-}1|x_1^{t-1}) \end{pmatrix}\\
&\qquad+ \begin{pmatrix} P(q_t{=}i|q_{t-1}, x_t)\cdot \\ P(\tau_t{=}s|\tau_{t-1}{=}s, q_t{=}i)\cdot \\ P(y_1^s, q_1^{t-1}, \tau_{t-1}{=}s|x_1^{t-1}) \end{pmatrix}
\end{aligned}$$

so,

$$\begin{aligned}
V(i,t) \;&=\\
&\max_{s,\tau,y_1^s,q_1^{t-1}} \begin{pmatrix} \max(b(i,y_s,t)(1{-}\epsilon(i,t))a(i,j,t)\cdot \\ P(y_1^{s-1}, q_1^{t-1}, \tau_{t-1}{=}s{-}1|x_1^{t-1}), \\ \epsilon(i,t)a(i,j,t)P(y_1^s, q_1^{t-1}, \tau_{t-1}{=}s|x_1^{t-1})) \end{pmatrix}
\end{aligned}$$

We distribute the maximizations inside the right hand side factors and obtain the recursive formula:

$$\begin{aligned} V(i,t) &= \max(\epsilon(i,t), (1-\epsilon(i,t)) \max_l b(i,l,t)) \\ &\quad \max_{j=x_{t-1}} (a(i,j,t)V(j,t-1)) \end{aligned} \tag{71}$$

The initialization of $V(i,0)$ is performed as follows:

$$V(i,0) = \pi(i) \tag{72}$$