To appear in The Handbook of Brain Theory and Neural Networks, Second edition, (M.A. Arbib, Ed.), Cambridge, MA: The MIT Press, 2002.

http://mitpress.mit.edu

The MIT Press

Hidden Markov Models and other Finite State Automata for Sequence Processing

Hervé Bourlard[†,‡] and Samy Bengio[†]

[†] Dalle Molle Institute for Perceptual Artificial Intelligence (IDIAP)

P.O. Box 592, 1920 Martigny, Switzerland

[‡] Swiss Federal Institute of Technology, Lausanne (EPFL)

{bourlard,bengio}@idiap.ch

RUNNING HEAD: Hidden Markov Models and other Finite State Automata

Correspondence:

Hervé Bourlard

IDIAP

P.O. Box 592

1920 Martigny, Switzerland

bourlard@idiap.ch; Phone +41 27 721 77 20; FAX +41 27 721 77 12

INTRODUCTION

During these last 20 years, Finite State Automata (FSA), and more particularly Stochastic Finite State Automata (SFSA) and different variants of Hidden Markov Models (HMMs), have been used quite successfully to address several complex sequential pattern recognition problems, such as continuous speech recognition, cursive (handwritten) text recognition, time series prediction, biological sequence analysis, and many others.

FSA allow complex learning problems to be solved by assuming that the sequential pattern can be decomposed into piecewise stationary segments, encoded through the topology of the FSA. Each stationary segment can be parametrized in terms of a deterministic or stochastic function. In the latter case, it may also be possible that the SFSA state sequence is not observed directly but is a probabilistic function of the underlying finite state Markov chain. This thus yields to the definition of the powerful Hidden Markov Models, involving two concurrent stochastic processes: the sequence of HMM states modeling the sequential structure of the data, and a set of state output processes modeling the (local) stationary character of the data. The HMM is called “hidden” because there is an underlying stochastic process (i.e., the sequence of states) that is not observable, but affects the observed sequence of events.

Furthermore, depending on the way the SFSA is parametrized, and the way it is trained, SFSA (and HMM in particular) can be used as a *production model* (where the observation sequence is considered as an output signal being produced by the model) or as a *recognition model* (acceptor) (where the observation sequence is considered as being accepted by the model). Finally, it may also be the case that the HMM is used to explicitly model the stochastic relationship between two (input and output) event sequences, then yielding to a model usually referred to as input/output HMM.

The parameters of these models can be trained by different variants of the powerful Expectation-Maximization (EM) algorithm (Baum and Petrie, 1966; Liporace, 1982), which, depending on the criterion being used, is referred to as Maximum Likelihood (ML) or Maximum A Posteriori (MAP) training. However, although being part of the same family of models, all these models exhibit different properties. The present paper thus aims at presenting and comparing some of the variants of these powerful SFSA and HMM models currently used for sequence processing.

FINITE STATE AUTOMATA

In its more general form (Hopcroft, Motwani and Ullman, 2000), and as summarized in Table 1, a Finite State Automata (FSA), which will be denoted M in this paper, is defined as an abstract machine consisting of:

- A set of states $\mathcal{Q} = \{I, 1, \ldots, k, \ldots, K, F\}$, including the initial state I and final state F, also referred to as accepting state (in the case of recognizers). Variants of this include machines having multiple initial states and multiple accepting states. In the present paper, a specific state visited at time t will be denoted q_t.

- A set $\mathcal{Y}$ of (discrete or continuous) input symbols or vectors. A particular sequence of size T of input symbols/vectors will be denoted $Y = \{y_1, y_2, \ldots, y_t, \ldots, y_T\} = y_1^T$, where y_t represents the input symbol/vector at time t.

- A set $\mathcal{Z}$ of (continuous or discrete) output symbols or vectors. A particular sequence of size T of output symbols/vectors will be denoted $Z = z_1^T$, where z_t represents the output symbol/vector at time t.

- A *state transition function* $q_t = f(y_t, q_{t-1})$, which takes the current input event and the previous state q_{t-1} and returns the the next state q_t.

- An *emission function* $z_t = g(q_t, q_{t-1})$, which takes the current state q_t and the previous state q_{t-1} and returns an output event z_t. This automaton is usually know as a *Mealy FSA*, i.e., producing an output for each *transition*. As a variant of this, the emission function of a *Moore FSA* depends only on the current state, i.e. $z_t = g(q_t)$, thus producing an output for each visited state. It is however proved that there is a homomorphic equivalence between Mealy and Moore automata, provided an increase and renaming of the states.

Finally, in the case of sequential pattern processing, the processed sequence is often represented as an *observed sequence* of symbols or vectors which, depending on the type of automata and optimization criterion, will sometimes be considered as input or output events. To accommodate this flexibility in the sequel of the present paper, we thus also define the *observed sequence* of size T as $X = x_1^T$, where x_t is the observed event/vector at time t. For example, in the case of speech recognition, x_t would be the acoustic vector resulting of the spectral analysis of the signal at time t, and is equivalent to z_t (since in that case the observations are the outputs of the FSA).

	Deterministic FSA	Stochastic Finite State Automata (SFA)			
		Markov Model	HMM	HMM/ANN	IOHMM
States	$k \in \mathcal{Q}$	$x_t = k \in \mathcal{Q}$	$k \in \mathcal{Q}$	$k \in \mathcal{Q}$	$k \in \mathcal{Q}$
Input symbols	$y_t \in \mathcal{Y}$	—	—	$x_t = y_t \in \mathcal{Y}$	$x_t = y_t \in \mathcal{Y}$
Output symbols	$z_t \in \mathcal{Z}$	—	$x_t = z_t \in \mathcal{Z}$	—	$z_t \in \mathcal{Z}$
Transition law	$q_t = f(y_t, q_{t-1})$	trans. probabs. $p(x_t \mid x_{t-1})$	trans. probabs. $p(q_t \mid q_{t-1})$	cond. trans. probabs. $p(q_t \mid x_t, q_{t-1})$	cond. trans. probabs. $p(q_t \mid x_t, q_{t-1})$
Emission law					
Mealy	$z_t = g(q_t, q_{t-1})$	—	emission on transition $x_t = g(q_t, q_{t-1})$ $p(x_t \mid q_{t-1}, q_t)$	—	$p(z_t \mid q_t, q_{t-1}, x_t, x_{t-1})$
Moore	$z_t = g(q_t)$	—	emission on state $x_t = g(q_t)$ $p(x_t \mid q_t)$	—	$p(z_t \mid q_t, x_t)$
Training Methodology	Many often based on heuristics	Relative Counts (incl. smoothing)	EM Viterbi	REMAP (GEM)	GEM EM, GD Viterbi
Training Criterion	Deterministic	Maximum Likelihood	Maximum Likelihood	Maximum A Posteriori	Maximum A Posteriori

A *deterministic FSA* is one where the transition and emission functions $f(.)$ and $g(.)$ are deterministic, implying that the output event and next state are uniquely determined by a single input event (i.e., there is exactly one transition for each given input event and state). In comparison, a non-deterministic FSA is one where the next state depends not only on the current input event, but also on a number of subsequent input events. However, it is often possible to transform a non-deterministic FSA into a deterministic FSA, at the cost of a significant increase of the possible number of input symbols.

It is not the goal of the present paper to further discuss deterministic FSA, which have been largely used in language theory (Hopcroft, Motwani and Ullman, 2000) where FSA are often used to accept or reject a language, i.e., certain sequences of input events. Many training approaches have been developed, mainly aiming at automatically inferring the FSA topology from a set of observation sequences. However, these often depend on the assumed properties (grammar) of the sequences to be processed, and finding the minimum FSA (minimizing the number of states and transitions) is often an open issue.

STOCHASTIC FINITE STATE AUTOMATA

A *Stochastic FSA* (SFSA) is a Finite State Automaton where the transition and/or emission functions are probabilistic functions. In the case of Markov Models, there is a one-to-one relationship between the observation and the state, and the transition function is probabilistic. In the case of Hidden Markov Models, the emission function is also probabilistic, and the states are no longer directly observable through the input events. Instead, each state produces one of the possible output events with a certain probability.

Depending on their structure (as discussed below), transition and emission (probability density) functions are thus represented in terms of a set of parameters Θ, which will have to be estimated on representative training data. If X represents the whole sequence of training data, and M its associated SFSA, the estimation of optimal parameter set Θ^* is usually achieved by optimizing a *maximum likelihood criterion*:

$$\Theta^* = \underset{\Theta}{\text{argmax}}\; p(X|M,\Theta) \tag{1}$$

or a *maximum a posteriori* criterion, which could be either

$$\Theta^* = \underset{\Theta}{\text{argmax}}\; p(M|X,\Theta) = \underset{\Theta}{\text{argmax}}\; p(X|M,\Theta)p(M|\Theta) \tag{2}$$

or

$$\Theta^* = \underset{\Theta}{\text{argmax}}\; p(M,\Theta|X) = \underset{\Theta}{\text{argmax}}\; p(X|M,\Theta)p(M,\Theta)\;. \tag{3}$$

In the first case, we take into account the prior distribution of the model M while in the second case, we take into account the prior distribution of the model M as well as the parameters Θ.

Markov Models

The simplest form of SFSA, as presented in the second column of Table 1, is a Markov model where states are directly associated with the observations. We are interested in modelling

$$p(X) = p(F|x_1^T)p(x_1|I)\prod_{t=2}^{T} p(x_t|x_1^{t-1}, I)$$

which can be simplified, using the k^{th} order Markov assumption by

$$p(X) = p(F|x_{T-k+1}^T)p(x_1|I)\prod_{t=2}^{T} p(x_t|x_{t-k}^{t-1})$$

which leads to the simplest case of the first-order Markov model,

$$p(X) = p(F|x_T)p(x_1|I)\prod_{t=2}^{T} p(x_t|x_{t-1})$$

where $p(x_1|I)$ is the initial state probability and the other terms can be seen as a transition probabilities. Note that any k^{th} order Markov model can be expressed as a first-order Markov model, at the cost of possibly exponentially more states. Note also that the transition probabilities are time invariant, i.e. $p(x_t{=}\ell|x_{t-1}{=}k)$ is fixed for all t.

The set of parameters, represented by the $(K \times K)$-transition probability matrix, i.e.

$$\Theta = p(x_t{=}\ell|x_{t-1}{=}k),\ \forall \ell, k \in \mathcal{Q}$$

is then directly estimated on a large amount of possible observation (and, thus, state) sequences such that:

$$\Theta^* \;=\; \underset{\Theta}{\text{argmax}}\; p(X|M, \Theta)$$

and simply amounts to estimating the relative counts of observed transitions, possibly smoothed in the case of undersampled training data, i.e.:

$$p(x_t{=}\ell|x_{t-1}{=}k) = \frac{n_{k\ell}}{n_k}$$

where $n_{k\ell}$ stands for the number of times a transition from state k to state ℓ was observed, while n_k represents the number of times state k was visited.

It is sometimes desirable to compute the probability to go from the initial state I to the final state F in exactly T steps, which could naively be estimated by summing path likelihoods over all possible paths of length T in model M, i.e.

$$p(F|I) = p(x_1|I) \sum_{paths} p(F|x_T) \prod_{t=2}^{T} p(x_t|x_{t-1})$$

although there is a possibly exponential number of paths to explore. Fortunately, a more tractable solution exists, using the intermediate variable

$$\alpha_t(\ell) = p(x_t{=}\ell, x_1^{t-1}, I)$$

which can be computed using the *forward recurrence*:

$$\alpha_t(\ell) = \sum_k \alpha_{t-1}(k) p(x_t{=}\ell|x_{t-1}{=}k) \tag{4}$$

and can be used as followed

$$p(F|I) = \alpha_{T+1}(F) \ .$$

Replacing the sum operator in (4) by the max operator is equivalent to finding the most probable path of length T between I and F.

Although quite simple, Markov models have many usage. For example, they are used in all state-of-the-art continuous speech recognition system to represent statistical grammars (Jelinek, 1998), usually referred to as N-grams, and estimating the probability of a sequence of K words

$$p(w_1^K) \approx \prod_{k=N+1}^{K} p(w_k|w_{k-N}^{k-1})$$

which is equivalent to assuming that possible word sequences can be modeled by a Markov model of order N.

Hidden Markov Models (HMM)

In many sequential pattern processing/classification problems (such as speech recognition and cursive handwriting recognition), one of the greatest difficulties is to simultaneously model the inherent statistical variations in sequential rates and feature characteristics. In this respect, Hidden Markov Models (HMMs) have been one of the most successful approaches used so far. As presented in Table 1, an HMM is a particular form of SFSA where Markov models (modeling the sequential properties of the data) are complemented by a second stochastic process modeling the local properties of the data. The HMM is called “hidden” because

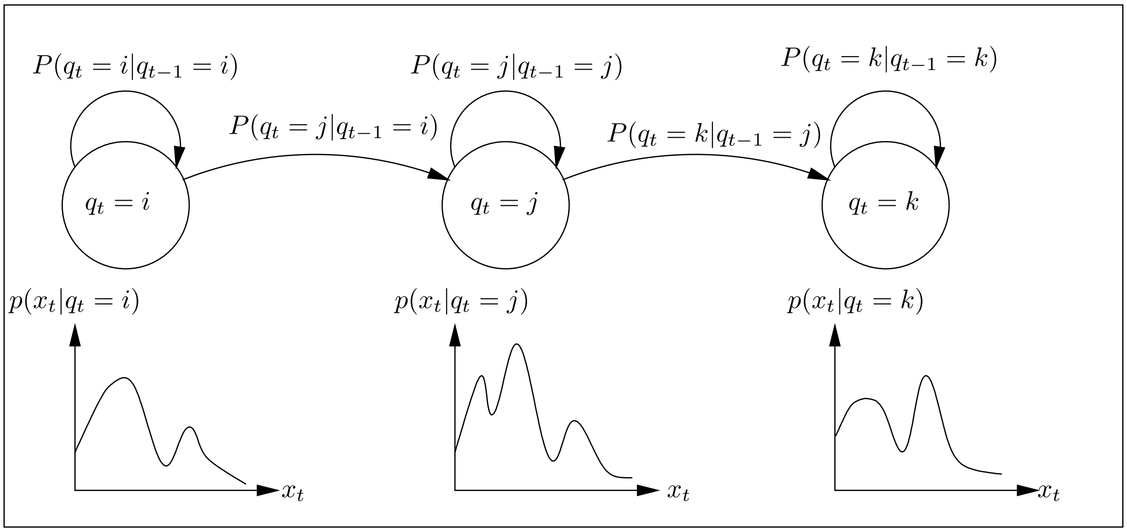
$P(q_t = i | q_{t-1} = i)$
$P(q_t = j | q_{t-1} = j)$
$P(q_t = k | q_{t-1} = k)$
$P(q_t = j | q_{t-1} = i)$
$P(q_t = k | q_{t-1} = j)$
$q_t = i$
$q_t = j$
$q_t = k$
$p(x_t | q_t = i)$
$p(x_t | q_t = j)$
$p(x_t | q_t = k)$
x_t
x_t
x_t

Figure 1: A schematic of a three state, left-to-right hidden Markov model (HMM).

there is an underlying stochastic process (i.e., the sequence of states) that is not observable, but affects the observed sequence of events.

Although sequential signals, such as speech and handwriting, are non-stationary processes, HMMs thus assume that the sequence of observation vectors is a *piecewise stationary* process. That is, a sequence $X = x_1^T$ is modeled as a succession of discrete stationary states $Q = \{1, \dots, k, \dots, K\}$, with instantaneous transitions between these states. In this case, a HMM is defined as a stochastic finite state automata with a particular (generally strictly left-to-right for speech data) topology. An example of a simple HMM is given in Figure 1. In speech recognition, this could be the model of a word or phoneme which is assumed to be composed of three stationary parts. In cursive handwriting recognition, this could be the model of a letter.

Once the topology of the HMM has been defined (usually arbitrarily!), the main criterion used for training and decoding is based on the likelihood $p(X|M, \Theta)$, i.e., the probability that the observed vector sequence X was produced by Markov model M. In this case, the HMM is thus considered as a *production model*, and the observation vectors x_t are thus considered as output variables z_t of the HMM. It can be shown that, provided several assumptions (Bourlard and Morgan, 1993), the likelihood $p(X|M, \Theta)$ can be expressed and computed in terms of transition probabilities $p(q_t{=}\ell | q_{t-1}{=}k, \Theta)$ and *emission probabilities*, which can be of the Mealy type (emission on transitions) $p(x_t | q_t, q_{t-1}, \Theta)$ or of the Moore type (emission on states) $p(x_t | q_t, \Theta)$. In the case of multivariate continuous observations, these emission probabilities are estimated by assuming that they follow a particular functional distribution, usually (mixtures of) multivariate Gaussian densities. In this case, the set of parameters Θ comprises all the Gaussian means and variances, mixing coefficients, and transition probabilities. These parameters are then usually trained according to the maximum likelihood criterion (1), resulting in the efficient Expectation-Maximization (EM) algorithm (Gold and Morgan, 2000; Liporace, 1982).

Given this formalism, the likelihood of an observation sequence X given the model M can be calculated by extending the forward recurrence (4) defined for Markov models to also include the emission probabilities. Assuming a Moore automaton (emission on states), we thus have the *forward recurrence*

$$\alpha_t(\ell) = p(x_1^t, q_t{=}\ell) = p(x_t|q_t{=}\ell) \sum_k \alpha_{t-1}(k) p(q_t{=}\ell|q_{t-1}{=}k) \tag{5}$$

which will be applied over all possible t, and where $\sum_k$ is applied over all possible predecessor states of ℓ, thus resulting in:

$$p(X|M,\Theta) = \alpha_{T+1}(F) \ .$$

Replacing the sum operator in (5) by the max operator is equivalent to finding the most probable path of length T generating the sequence X, and then yields the well known dynamic programming recurrence, also referred to as the *Viterbi recurrence* in the case of HMM:

$$\overline{p}(x_1^t, q_t{=}\ell) = p(x_t|q_t{=}\ell) \ \max_{\{k\}} \left\{ \overline{p}(x_1^{t-1}, q_{t-1}{=}k) p(q_t{=}\ell|q_{t-1}{=}k) \right\} \tag{6}$$

where $\overline{p}(x_1^t, q_t{=}\ell)$ represents the probability of having produced the partial observation sequence x_1^t while being in state ℓ at time t and having followed the most probable path; $\{k\}$ represents the set of possible predecessor states of ℓ (given by the topology of the HMM); the likelihood $\overline{p}(X|M,\Theta)$ of the most probable path is obtained at the end of the sequence and is equal to $\overline{p}(x_1^T, F)$.

During training, the HMM parameters Θ are optimized to maximize the likelihood of a set of training utterances given their associated (and known during training) HMM model, according to (1) where $p(X|M,\Theta)$ is computed by taking all possible paths into account (forward recurrence) or only the most probable path (Viterbi recurrence). Powerful iterative training procedures, based on the Expectation-Maximization (EM) algorithm, exist for both criteria, and have been proved to converge to a local optimum. At each iteration of the EM algorithm, the "Expectation" step estimates the most probable segmentation or the best state posterior distribution (referred to as "hidden variables") based on the current values of the parameters, while the "Maximization" step re-estimates the optimal value of these parameters assuming that the current estimate of the hidden variables is correct.

For further reading of the HMM training and decoding algorithms, see (Bourlard and Morgan, 1993; Deller, Proakis and Hansen, 1993; Gold and Morgan, 2000; Jelinek, 1998).

HMM Advantages and Drawbacks

The most successful application of HMMs is speech recognition. Given a sequence of acoustic, the goal is to produce a sequence of associated phoneme or word transcription. In order to solve such a problem, one usually associates one HMM per different phoneme (or word). During training, a new HMM is created for each training sentence as the concatenation of the corresponding target phoneme models, and its parameters are maximized. Over the last few years, a number of laboratories have demonstrated large-vocabulary (at least 1,000 words), speaker-independent, continuous speech recognition systems based on HMMs.

HMMs are models that can deal efficiently with the temporal aspect of speech (time warping) as well as with frequency distortion. They also benefit from powerful and efficient training and decoding algorithms. For training, only the transcription in terms of the speech units which are trained is necessary and no explicit segmentation of the training material is required. Also, HMMs can easily be extended to include phonological and syntactical rules (at least when these are using the same statistical formalism).

However, the assumptions that make the efficiency of these models and their optimization possible limit their generality. As a consequence, they also suffer from several drawbacks including, among others:

- Poor discrimination due to the training algorithm which maximizes likelihoods instead of *a posteriori* probabilities $p(M|X)$ (i.e., the HMM associated with each speech unit is trained independently of the other models).

- *A priori* choice of model topology and statistical distributions, e.g., assuming that the probability density functions associated with the HMM state can be described as (mixtures of) multivariate Gaussian densities, each with a diagonal-only covariance matrix (i.e., the possible correlation between the components of the acoustic vectors is disregarded).

- Assumption that the state sequences are first-order Markov chains.

- Assumption that the input observations are not correlated over time. Thus, apart through the HMM topology, the possible temporal correlation across features associated with the same HMM state is simply disregarded.

In order to overcome some of these problems, many researchers have concentrated on integrating Artificial Neural Networks into the formalism of HMMs. In the next section, we expose some of the most promising approaches.

ANN-BASED STOCHASTIC FINITE STATE AUTOMATA

The idea of combining HMMs and ANNs was motivated by the observation that HMMs and ANNs had complementary properties: (1) HMMs are clearly dynamic and very well suited to sequential data, but several assumptions limit their generality; (2) ANNs can approximate any kind of nonlinear discriminant functions, are very flexible and do not need strong assumptions about the distribution of the input data, but they cannot properly handle time sequences (although recurrent neural networks can indeed handle time, they are known to be difficult to train long term dependencies, and cannot easily incorporate knowledge in their structure as it is the case for HMMs). Therefore a number of hybrid models have been proposed in the literature.

Hybrid HMM/ANN Systems

HMMs are based on a strict probabilistic formalism, making them difficult to interface with other modules in a heterogeneous system. However, it has indeed been shown (Bourlard and Morgan, 1993; Richard and Lippmann, 1991) that if each output unit of an ANN (typically a multilayer perceptron) is associated with a state k of the set of states $\mathcal{Q} = \{1, 2, \ldots, K\}$ on which the SFSA are defined, it is possible to train the ANN (e.g., according to the usual least means square or relative entropy criteria) to generate good estimates of *a posteriori* probabilities of the output classes conditioned on the input. In other words, if $g_k(x_t|\Theta)$ represents the output function observed on the k-th ANN output unit when the ANN is being presented with the input observation vector x_t, we will have:

$$g_k(x_t|\Theta^*) \approx p(q_t{=}k|x_t) \tag{7}$$

where Θ^* represents the optimal set of ANN parameters.

When using these posterior probabilities (instead of local likelihoods) in SFSA, the model becomes a recognition model (sometimes referred to as stochastic finite state acceptor), where the observation sequence is an *input* to the system, and where all local and global measures are based on a posteriori probabilities. It was thus necessary the revisit the SFSA basis to accommodate this formalism. In (Bourlard and Morgan, 1993; Bourlard, Konig and Morgan, 1996), it is shown that $p(M|X, \Theta)$ can be expressed in terms *conditional transition probabilities* $p(q_t|x_t, q_{t-1})$ and that it is possible to train the optimum ANN parameter set Θ according to the MAP criterion (2). The resulting training algorithm (Bourlard, Konig and Morgan, 1994), referred to as REMAP (Recursive Estimation and Maximization of A Posteriori Probabilities) is a particular form of EM training, directly involving posteriors, where the "Maximization" step involves the (gradient-based) training of the ANN, and where the desired target distribution (required to train the ANN) has been estimated in the previous "Expectation" step. Since this EM version includes an iterative "Maximization" step, it is also sometimes referred to as Generalized EM (GEM). As for standard HMMs, there is a full likelihood version (taking all possible paths into account) as well as a Viterbi version of the training procedure.

Another popular solution in using hybrid HMM/ANN as a sequence recognizer is to turn the local posterior probabilities $p(q_t{=}k|x_t)$ into *scaled likelihoods* by dividing these by the estimated value of the class priors as observed on the training data, i.e.:

$$\frac{p(q_t{=}k|x_t)}{p(q_t{=}k)} = \frac{p(x_t|q_t{=}k)}{p(x_t)} \tag{8}$$

These scaled likelihoods are trained discriminatively (using the discriminant properties of ANN); during decoding though, the denominator of the resulting scaled likelihoods $\frac{p(x_t|q_t=k)}{p(x_t)}$ is independent of the class and simply appears as a normalization constant. The scale likelihoods can thus be simply used in a regular Viterbi or forward recurrence to yield an estimator of the global scaled likelihood (Hennebert et al., 1997):

$$\frac{p(X|M,\Theta)}{p(X)} = \sum_{paths} \prod_{t=1}^{T} \frac{p(x_t|q_t)}{p(x_t)} p(q_t|q_{t-1}) \tag{9}$$

where the sum extends over all possible paths of length T in model M.

These hybrid HMM/ANN approaches provide more discriminant estimates of the emission probabilities needed for HMMs, without requiring strong hypotheses about the statistical distribution of the data. Since this result still holds with modified ANN architectures, the approach has been extended in a number of ways, including:

- Extending the input field to accommodate not only the current input vector but also its right and left contexts, leading to HMM systems that take into account the correlation between acoustic vectors (Bourlard and Morgan, 1993).

- Partially recurrent ANN (Robinson, Hochberg and Renals, 1996) feeding back previous activation vectors on the hidden or output units, leading to some kind of higher-order HMM.

Input/Output HMMs

Input/Output Hidden Markov Models (IOHMMs) (Bengio and Frasconi, 1995) are an extension of classical HMMs where the emission and transition probability distributions are conditioned on another sequence, called the input sequence, and noted $Y = y_1^T$. The emitted sequence is now called the output sequence, noted $Z = z_1^T$. Hence, in the simplest case of the Moore model as presented in Table 1, the emission distribution now models $p(z_t|q_t, y_t)$ while the transition distribution models $p(q_t|q_{t-1}, y_t)$.

While this looks like an apparently simple modification, it has structural impacts on the resulting model and hence on the hypotheses of the problems to solve. For instance, while in classical HMMs the emission and transition distributions do not depend on t (we say that HMMs are homogeneous), this is not the case anymore for IOHMMs, which are hence called inhomogeneous, as the distributions are now conditioned on y_t, which changes with time t.

Applications of IOHMMs range from speech processing to sequence classification tasks, and include time series prediction and robot navigation. For example, for economic time series, the input sequence could represent different economic indicators while the output sequence could be for instance the future values of some target assets or the evolution of a given portfolio.

In order to train IOHMMs, an EM algorithm has been developed (Bengio and Frasconi, 1995) which looks very much as the classical EM algorithm used for HMMs, except that all distributions and posterior estimates are now conditioned on the input sequence. Hence we need to implement conditional distributions, either for transitions or emissions, which can be represented for instance by artificial neural networks (ANNs). The resulting training algorithm is thus a Generalized EM, which is also guaranteed to converge. For transition probabilities, the output of the ANN would represent the posterior probability of each transition $p(q_t|q_{t-1}, y_t)$ with the constraint that all such probabilities from a given state sum to 1. For emission probabilities, the output of the ANN would represent the parameters of a unconditional probability distribution, such as a classical mixture of Gaussians. Another implementation option would be to use an ANN to represent only the expectation $E[z_t|q_t, y_t]$ instead of the probability itself. For some applications such as prediction, this is often sufficient and more efficient.

An interesting extension of IOHMMs has been proposed in (Bengio and Bengio, 1996) in order to handle asynchronous input/output sequences, hence being able to match input sequences that might be shorter or longer than output sequences. An obvious application of asynchronous IOHMMs is speech recognition (or handwritten cursive recognition) where the input sequence represents the acoustic while the output sequence represents the corresponding phoneme transcription.

DISCUSSION

In this paper, we discussed the use of deterministic and stochastic finite state automata for sequence processing, also attempting to present this in an unified framework. As a particularly powerful instantiation of

SFSA, and one of the most popular tools for the processing of complex piecewise stationary sequences, we also discussed hidden Markov models in more detail. Finally, we described a few (certainly not exhaustive) contributions of the artificial neural network (ANN) community to further improve those SFSA, mainly including hybrid HMM/ANN systems and Input/Output HMMs.

Of course, this overview and comparison could have been extended to many other related areas such as transducers, linear dynamical systems, Kalman filters, and others. This, however, would have fell outside the scope of this short introduction.

REFERENCES

Baum, L. and Petrie, T. (1966). Statistical inference for probabilistic functions of finite state markov chains. *Annals of Mathematical Statistics*, 37:1554–1563.

Bengio, S. and Bengio, Y. (1996). An EM algorithm for asynchronous input/output hidden markov models. In *Proceedings of the International Conference on Neural Information Processing, ICONIP*, Honk Kong.

Bengio, Y. and Frasconi, P. (1995). An input/output HMM architecture. In *Advances in Neural Information Processing Systems 7*, pages 427–434. MIT Press, Cambridge, MA.

Bourlard, H., Konig, Y., and Morgan, N. (1994). REMAP: Recursive estimation and maximization of a posteriori probabilities. Technical Report TR-94-064, Intl. Computer Science Institute, Berkeley, CA.

Bourlard, H., Konig, Y., and Morgan, N. (1996). A training algorithm for statistical sequence recognition with applications to transition-based speech recognition. *IEEE Signal Processing Letters*, 3(7):203–205.

*Bourlard, H. and Morgan, N. (1993). *Connectionist Speech Recognition – A Hybrid Approach*. Kluwer Academic Publishers.

*Deller, J., Proakis, J., and Hansen, J. (1993). *Discrete-Time Processing of Speech Signals*. MacMillan.

*Gold, B. and Morgan, N. (2000). *Speech and Audio Signal Processing*. Wiley.

Hennebert, J., Ris, C., Bourlard, H., Renals, S., and Morgan, N. (1997). Estimation of global posteriors and forward-backward training of hybrid HMM/ANN systems. In *Proceedings of Eurospeech'97*, pages 1951–1954.

*Hopcroft, J., Motwani, R., and Ullman, J. (2000). *Introduction to Automata Theory, Language and Computations, second edition*. Addison-Wesley.

*Jelinek, F. (1998). *Statistical Methods for Speech Recognition*. MIT Press.

Liporace, L. (1982). Maximum likelihood estimation for multivariate observations of markov sources. *IEEE Trans. on Information Theory*, IT-28(5):729–734.

Richard, M. and Lippmann, R. (1991). Neural network classifiers estimate bayesian a posteriori probabilities. *Neural Computation*, 3:461–483.

Robinson, T., Hochberg, M., and Renals, S. (1996). The use of recurrent neural networks in continuous speech recognition. In *Automatic Speech and Speaker Recognition*, pages 233–258. Kluwer Academic Publishers.