The Hierarchical Hidden Markov Model: Analysis and Applications

SHAI FINE fshai@cs.huji.ac.il
Institute of Computer Science, Hebrew University, Jerusalem 91904, Israel

YORAM SINGER singer@research.att.com
AT&T Labs, 180 Park Avenue, Florham Park, NJ 07932

NAFTALI TISHBY tishby@cs.huji.ac.il
Institute of Computer Science, Hebrew University, Jerusalem 91904, Israel

Abstract. We introduce, analyze and demonstrate a recursive hierarchical generalization of the widely used hidden Markov models, which we name Hierarchical Hidden Markov Models (HHMM). Our model is motivated by the complex multi-scale structure which appears in many natural sequences, particularly in language, handwriting and speech. We seek a systematic unsupervised approach to the modeling of such structures. By extending the standard forward-backward (Baum-Welch) algorithm, we derive an efficient procedure for estimating the model parameters from unlabeled data. We then use the trained model for automatic hierarchical parsing of observation sequences. We describe two applications of our model and its parameter estimation procedure. In the first application we show how to construct hierarchical models of natural English text. In these models different levels of the hierarchy correspond to structures on different length scales in the text. In the second application we demonstrate how HHMMs can be used to automatically identify repeated strokes that represent combination of letters in cursive handwriting.

1. Introduction

Hidden Markov models (HMMs) have become the method of choice for modeling stochastic processes and sequences in applications such as speech and handwriting recognition (Rabiner, 1986), (Nag et al., 1985) and computational molecular biology (Krogh et al., 1993), (Baldi et al., 1994). Hidden Markov models are also used for natural language modeling (see e.g. (Jelinek, 1985)). In most of these applications the model's topology is determined in advance and the model parameters are estimated by an EM procedure (Dempster et al., 1977), known as the forward-backward (or Baum-Welch) algorithm in this context (Baum and Petrie, 1966). Some recent worked has explored the inference of the model structure as well (Stolcke and Omohundro, 1994). In most of the above applications, however, there are difficulties due to the multiplicity of length scales and recursive nature of the sequences. Some of these difficulties can be overcome using stochastic context free grammars (SCFG). The parameters of stochastic grammars are difficult to estimate since typically the likelihood of observed sequences induced by a SCFG varies dramatically with small changes in the parameters of the model. Furthermore, the common algorithm for parameter estimation of SCFGs, called the inside-outside

algorithm (Lari and Young, 1990), has a cubic time complexity in the length of the observed sequences.

In this paper we present a hierarchical generalization of the hidden Markov model. Our primary motivation is to enable better modeling of the different stochastic levels and length scales that are present in the natural language, whether speech, handwriting, or text. Another important property of such models is the ability to infer correlated observations over long periods in the observation sequence via the higher levels of the hierarchy. We show how to efficiently estimate the model parameters through an estimation scheme inspired by the inside-outside algorithm. The structure of the model we propose is fairly general and allows an arbitrary number of activations of its submodels. This estimation procedure can be efficiently approximated so that the overall computation time is only quadratic in the length of the observations. Thus long time correlations can be captured by the model while keeping the running time reasonable. We demonstrate the applicability of the model and its estimation procedure by learning a multi-resolution structure of natural English text. The resulting models exhibit the formation of "temporal experts" of different time scales, such as punctuation marks, frequent combinations of letters, and endings of phrases. We also use the learning algorithm of hierarchical hidden Markov models for unsupervised learning of repeated strokes that represent combinations of letters in cursive handwriting. We then use submodels of the resulting HHMMs to spot new occurrences of the same letters combination in unlabeled data.

The paper is organized as follows: In Section 2 we introduce and describe the hierarchical hidden Markov model. In Section 3 we derive the estimation procedure for the parameters of the hierarchical hidden Markov model. In Section 4 we describe and demonstrate two applications that utilize the model and its estimation scheme. Finally, in Section 5 we discuss related work, describe several possible generalizations of the model, and conclude. In order to keep the presentation simple, most of the technical details are deferred to the technical appendices. A summary of the symbols and variables used in the paper is given in Appendix C.

2. Model description

Hierarchical hidden Markov models (HHMM) are structured multi-level stochastic processes. HHMMs generalize the standard HMMs by making each of the hidden states an "autonomous" probabilistic model on its own, that is, each state is an HHMM as well. Therefore, the states of an HHMM emit sequences rather than a single symbol. An HHMM generates sequences by a recursive activation of one of the substates of a state. This substate might also be composed of substates and would thus activate one of its substates, etc. This process of recursive activations ends when we reach a special state which we term a production state. The production states are the only states which actually emit output symbols through the usual HMM state output mechanism: an output symbol emitted in a production state is chosen according to a probability distribution over the set of output

symbols. Hidden states that do not emit observable symbols directly are called internal states. We term the activation of a substate by an internal state a vertical transition. Upon the completion of a vertical transition (which may include further vertical transitions to lower level states), control returns to the state which originated the recursive activation chain. Then, a state transition within the same level, which we call a horizontal transition, is performed. The set of states and vertical transitions induces a tree structure where the root state is the node at the top of the hierarchy and the leaves are the production states. To simplify notation we restrict our analysis to HHMMs with a full underlying tree structure, i.e., all the leaves are at the same distance from the root state. The analysis of HHMMs with a general structure is a straightforward generalization of the analysis presented here. The experiments described in this paper were performed with a general topology.

We would like to note in passing that every HHMM can be represented as a standard single level HMM. The states of the HMM are the production states of the corresponding HHMM with a fully connected structure, i.e., there is a non-zero probability of moving from any of the states to any other state. The equivalent HMM lacks, however, the multi-level structure which we exploit in the applications described in Section 4.

We now give a formal description of an HHMM. Let Σ be a finite alphabet. We denote by Σ^* the set of all possible strings over Σ. An observation sequence is a finite string from Σ^* denoted by $\bar{O} = o_1 o_2 \cdots o_T$. A state of an HHMM is denoted by q_i^d $(d \in \{1, \ldots, D\})$ where i is the state index and d is the hierarchy index. The hierarchy index of the root is 1 and of the production states is D. The internal states need not have the same number of substates. We therefore denote the number of substates of an internal state q_i^d by $|q_i^d|$. Whenever it is clear from the context, we omit the state index and denote a state at level d by q^d. In addition to its model structure (topology), an HHMM is characterized by the state transition probability between the internal states and the output distribution vector of the production states. That is, for each internal state q_i^d $(d \in \{1, \ldots, D-1\})$, there is a state transition probability matrix denoted by $A^{q^d} = (a_{ij}^{q^d})$, where $a_{ij}^{q^d} = P(q_j^{d+1} | q_i^{d+1})$ is the probability of making a horizontal transition from the ith state to the jth, both of which are substates of q^d. Similarly, $\Pi^{q^d} = \{\pi^{q^d}(q_i^{d+1})\} = \{P(q_i^{d+1} | q^d)\}$ is the initial distribution vector over the substates of q^d, which is the probability that state q^d will initially activate the state q_i^{d+1}. If q_i^{d+1} is in turn an internal state, then $\pi^d(q_i^{d+1})$ may also be interpreted as the probability of making a vertical transition: entering substate q_i^{d+1} from its parent state q^d. Each production state q^D is solely parameterized by its output probability vector $B^{q^D} = \{b^{q^D}(k)\}$, where $b^{q^D}(k) = P(\sigma_k | q^D)$ is the probability that the production state q^D will output the symbol $\sigma_k \in \Sigma$. The entire set of parameters is denoted by

$$\lambda = \{\lambda^{q^d}\}_{d \in \{1,\ldots,D\}} = \{\{A^{q^d}\}_{d \in \{1,\ldots,D-1\}}, \{\Pi^{q^d}\}_{d \in \{1,\ldots,D-1\}}, \{B^{q^D}\}\} \ .$$

An illustration of an HHMM with an arbitrary topology and parameters is given in Figure 1.

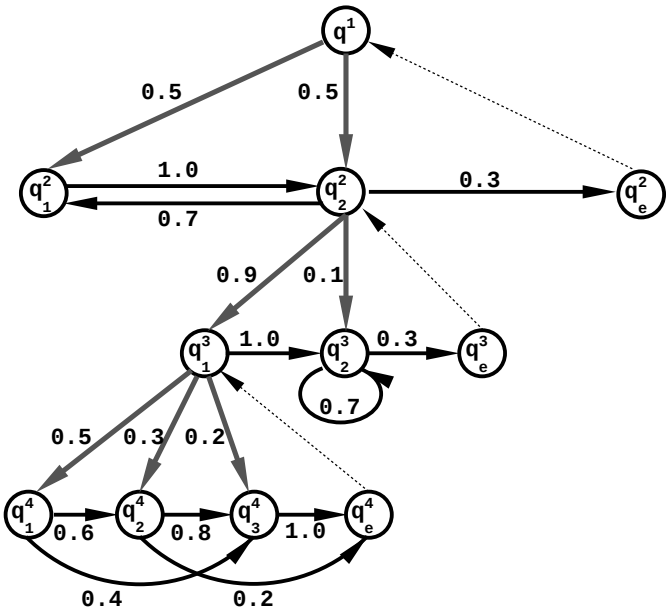

Figure 1. An illustration of an HHMM of four levels. Gray and black edges respectively denote vertical and horizontal transitions. Dashed thin edges denote (forced) returns from the end state of each level to the level's parent state. For simplicity, the production states are omitted from the figure.

To summarize, a string is generated by starting from the root state and choosing one of the root's substates at random according to Π^{q^1}. Similarly, for each internal state q that is entered, one of q's substates is randomly chosen according to q's initial probability vector Π^q. The operation proceeds with the chosen substate which recursively activates one of its substates. These recursive operations are carried out until a production state, q^D, is reached at which point a single symbol is generated according to a state output probability vector, B^{q^D}. Then control returns to the state activated q^D. Upon the completion of a recursive string generation, the internal state that started the recursion chooses the next state in the same level according to the level's state transition matrix and the newly chosen state starts a new recursive string generation process. Each level has a terminal state, denoted q^d_{end}, which is the actual means of terminating the stochastic state activation process. When a terminal state is reached, control returns to the parent state of the whole hierarchy. The generation of the observation sequence is completed when control of all the recursive activations is returned to the root state. We assume that all stats can be reached by a finite number of steps from the root state, that is, the model is strongly connected.

3. Inference and learning

As in the case with HMMs, three natural problems typically arise in applications that use HHMMs:

Calculating the likelihood of a sequence: Given an HHMM and its parameter set $\lambda = \{\lambda^{q^d}\}$, find the probability $P(\bar{O}|\lambda)$ of a sequence $\bar{O}$ to be generated by the model λ.

Finding the most probable state sequence: Given an HHMM, its parameter set $\lambda = \{\lambda^{q^d}\}$, and an observation sequence $\bar{O}$, find the single state activation sequence that is most likely to generate the observation sequence.

Estimating the parameters of a model: Given the structure of an HHMM and one or more observation sequences $\{\bar{O}_t\}$, find the most probable parameter set $\lambda^\star$ of the model, $\lambda^\star = \arg\max_\lambda P(\{\bar{O}_t\}|\lambda)$.

Solutions for the above problems for HHMMs are more involved than for HMMs, due to the hierarchical structure and multi-scale properties. For instance, the most probable state sequence given an observation sequence is a multi-resolution structure of state activations instead of a simple sequence of indices of the mostly probable states to be reached. We now present the solutions to these problems, starting with the simplest. We will be using the following terminology: we say that state q^d started its operation at time t if the (possibly empty) sub-sequence $o_1 \cdots o_{t-1}$ was generated before q^d was activated by its parent state, and the symbol o_t was generated by one of the production states reached from q^d. Analogously, we say that state finished its operation at time t if o_t was the last symbol generated by any of the production states reached from q^d, and control was returned to q^d from q^{d+1}_{end}.

3.1. Calculating the likelihood of a sequence

Since each of the internal states of an HHMM can be viewed as an autonomous model which can generate a substring of the observation using its substates, an efficient likelihood evaluation procedure should be recursive. For each state q^d we calculate the likelihood of generating a substring ω, denoted by $P(\omega|\lambda, q^d)$. Assume for the moment that these probabilities are provided except for the the root state q^1. Let $\bar{i} = (i_1, i_2, \ldots, i_l)$ be the indices of the states at the second level that were visited during the generation of the observation sequence $\bar{O} = o_1, o_2, \ldots, o_T$ of length T. Note that the last state entered at the second level is q^2_{end}, thus $q^2_{i_l} = q^2_{\text{end}}$. Let τ_j be the temporal position of the first symbol generated by state $q^2_{i_j}$, and let the entire list of these indices be denoted by $\bar{\tau} = (\tau_1, \tau_2, \ldots, \tau_l)$. Since $q^2_{i_1}$ was activated by q^1 at the first time step and q^2_{end} was the last state from the second level that was activated, we have $\tau_1 = 1$ and $\tau_l = T$. The likelihood of the entire sequence given the above information is,

$$
\begin{aligned}
&P(\bar{O}|\bar{\tau}, \bar{i}, \lambda) = \\
&\quad \pi^{q^1}(q^2_{i_1})\ P(o_1 \cdots o_{\tau_2 - 1}|q^2_{i_1}, \lambda) a^{q^1}_{i_1 i_2}\ P(o_{\tau_2} \cdots o_{\tau_3 - 1}|q^2_{i_2}, \lambda) a^{q^1}_{i_2 i_3} \\
&\quad \cdots P(o_{\tau_{l-2}} \cdots o_{\tau_{l-1} - 1}|q^2_{i_{l-2}}, \lambda) a^{q^1}_{i_{l-2} i_{l-1}}\ P(o_{\tau_{l-1}} \cdots o_T|q^2_{i_{l-1}}, \lambda)\ a^{q^1}_{i_{l-1}\text{end}} \quad .
\end{aligned}
$$

In order to calculate the unconditioned likelihood we need to sum over all possible switching times τ and state indices I. Clearly this is not feasible since there are exponentially many such combinations. Fortunately, the structure of HHMMs enables us to use dynamic programming to devise a generalized version of the forward-backward algorithm. The generalized forward probabilities, $\alpha(\cdot)$, are defined to be

$$\alpha(t, t+k, q_i^d, q^{d-1}) = P(o_t \cdots o_{t+k},\ q_i^d \text{ finished at } t+k \mid q^{d-1} \text{ started at } t) \quad .$$

That is, $\alpha(t, t+k, q_i^d, q^{d-1})$ is the probability that the partial observation sequence $o_t \cdots o_{t+k}$ was generated by state q^{d-1} and that q_i^d was the last state activated by q^{d-1} during the generation of $o_t \cdots o_{t+k}$. Note that the operation of each substate q^{d-1} does not necessarily end at time $t+k$ and that $o_t \cdots o_{t+k}$ can be a prefix of a longer sequence generated by q^{d-1}. To calculate the probability that the sequence $o_t \cdots o_{t+k}$ was generated by q^{d-1}, we need to sum over all possible states at level d ending at q_{end}^{d-1},

$$P(o_t \cdots o_{t+k} | q^{d-1}) = \sum_{i=1}^{|q^{d-1}|} \alpha(t, t+k, q_i^d, q^{d-1})\, a_{i\,end}^{q^{d-1}} \quad .$$

Finally, the likelihood of the whole observation sequence is obtained by summing over all possible starting states (called by the root state q^1),

$$P(\bar{O}|\lambda) = \sum_{i=1}^{|q^1|} \alpha(1, T, q_i^2, q^1) \quad .$$

The definition of the generalized α variables for the states at level $D-1$, $\alpha(t, t+k, q_i^D, q^{D-1})$, is equivalent to the definition of the forward variable $\alpha_{t+k}(i)$ of an HMM that consists of only this level and whose output probability vectors are defined by the production states q_i^D. The evaluation of the α variables is done in a recursive bottom-up manner such that the α values calculated for the substates of an internal state q are used to determine the α values of q.

In summary, for each internal state q we need to calculate its α value for each possible subsequence of the observation sequence using a recursive decomposition of each subsequence based on the α values of q's substates. Therefore the time complexity of evaluating the α values for all states of an HHMM is $O(NT^3)$, where N is the total number of states and T is the length of the observation sequence. In a similar manner, a generalized backward variable β is defined,

$$\beta(t, t+k, q_i^d, q^{d-1}) = P(o_t \cdots o_{t+k} |\ q_i^d \text{ started at } t,\ q^{d-1} \text{ finished at } t+k) \quad .$$

A detailed description of the calculation of α and β is provided in Appendix A.

3.2. Finding the most probable state sequence

The most probable state sequence is a multi-scale list of states: if state q had generated the string $o_i \cdots o_j$, then its parent state generated the string $o_k \cdots o_l$, such

that $k \leq i$ and $j \leq l$. Thus the string $o_i \cdots o_j$ is subdivided by the substates of state q to non-overlapping subsequences. This list can be computed efficiently following the same line of reasoning used to derive the α variables, replacing summation by maximization. Since the process which finds the most probable state sequence for HMMs is known as the Viterbi algorithm (Viterbi, 1967), we term the modified algorithm for HHMMs the generalized Viterbi algorithm.

Similar to the definition of the α variables we define $\delta(t, t+k, q_i^d, q^{d-1})$ to be the likelihood of the most probable (hierarchical) state sequence generating $o_t \cdots o_{t+k}$ given that q^{d-1} was entered at time t, its substate q_i^d was the last state to be activated by q^d, and control returned to q^d at time $t+k$. Since we are interested in the actual hierarchical parsing of the sequence into states we also maintain two additional variables: $\psi(t, t+k, q_i^d, q_d)$ is the index of the most probable state to be activated by q^{d-1} before activating q_i^d, and $t' = \tau(t, t+k, q_i^d, q_d)$ $(t \leq t' \leq t+k)$ is the time when q_i^d was activated by q^d. Given these two variables the most probable hierarchical state sequence is obtained by scanning the lists ψ and τ from the root state to the production states. If a breadth-first-search is used for scanning then the states are listed by their level index from top to bottom. If a depth-first-search is used then the states are listed by their activation time. Since we simply replaced summation with maximization the time complexity of the generalized Viterbi algorithm is the same as the time of the generalized forward-backward, namely $O(NT^3)$. The pseudo-code describing this algorithm is given in Appendix B.

We have also devised a heuristic that finds an approximation to the most probable state sequence in $O(NT^2)$ time. This heuristic assumes that the distributions over sequences induced by the different states are substantially different from each other. Hence the influence of the horizontal transitions on finding the most probable state sequence is negligible. We therefore conduct an approximated search that ignores the transition probabilities. In other words, we treat each state q^d of the HHMM as an autonomous model ignoring the influence of the neighboring states of q at level d. Therefore, only one maximization operation is performed at each internal node, reducing the overall running time to $O(NT^2)$. Although there is no theoretical justification for this approximation, we found in our experiments that the most probable state sequence found by the approximated search greatly resembles the state sequence found by the exact generalized Viterbi algorithm (see the experiments described in Section 4).

3.3. Estimating the parameters of an HHMM

The maximum-likelihood parameter estimation procedure for HHMMs is a generalization of the forward-backward algorithms since we also need to consider stochastic vertical transitions which recursively generate observations. Therefore, in addition to the path variables α and β which correspond to 'forward' and 'backward' transitions, we add additional path variables which correspond to 'downward' and 'upward' transitions. The variables used in the expectation step are as follows:

$\xi(t, q_i^d, q_j^d, q^{d-1})$ is the probability of performing a horizontal transition from q_i^d to q_j^d, both substates of q^{d-1}, at time t after the production of o_t and before the production of o_{t+1},

$$\xi(t, q_i^d, q_j^d, q^{d-1}) = P(o_1 \cdots o_t,\ q_i^d \longrightarrow q_j^d,\ o_{t+1} \cdots o_T \mid \lambda) \ .$$

Based on ξ we define two auxiliary variables γ_{in} and γ_{out} which simplify the re-estimation step:

$\gamma_{in}(t, q_i^d, q^{d-1})$ is the probability of performing a horizontal transition to state q_i^d before o_t was generated. γ_{in} is calculated using ξ by summing over all substates of q^{d-1} which can perform a horizontal transition to q_i^d,

$$\gamma_{in}(t, q_i^d, q^{d-1}) = \sum_{k=1}^{|q^{d-1}|} \xi(t-1, q_k^d, q_i^d, q^{d-1}) \ .$$

$\gamma_{out}(t, q_i^d, q^{d-1})$ is the probability of leaving state q_i^d by performing a horizontal transition to any of the states in the same level d after the generation of o_t. Analogous to γ_{in}, γ_{out} is calculated using ξ by summing over all substates of q^{d-1} that can be reached from q_i^d by a single horizontal transition,

$$\gamma_{out}(t, q_i^d, q^{d-1}) = \sum_{k=1}^{|q^{d-1}|} \xi(t, q_i^d, q_k^d, q^{d-1}) \ .$$

The path variable used to estimate the probability of a vertical transition is χ.

$\chi(t, q_i^d, q^{d-1})$ is the probability that state q^{d-1} was entered at time t before o_t was generated and initially chose to activate state q_i^d,

$$\chi(t, q_i^d, q^{d-1}) = P(o_1 \cdots o_{t-1},\ \begin{array}{c} q^{d-1} \\ \downarrow \\ q_i^d \end{array},\ o_t \cdots o_T \mid \lambda) \ .$$

Based on the above path variables and given the current set of parameters, the following expectations are calculated:

$\sum_{t=1}^{T-1} \xi(t, q_i^d, q_j^d, q^{d-1})$**:** the expected number of horizontal transitions from q_i^d to q_j^d, both are substates of the q^{d-1}.

$\sum_{t=2}^{T} \gamma_{in}(t, q_i^d, q^{d-1}) = \sum_{k=1}^{|q^{d-1}|} \sum_{t=2}^{T} \xi(t-1, q_k^d, q_i^d, q^{d-1})$**:** the expected number of horizontal transitions to state q_i^d from any of the states in level d.

$\sum_{t=1}^{T-1} \gamma_{out}(t, q_i^d, q^{d-1}) = \sum_{k=1}^{|q^{d-1}|} \sum_{t=1}^{T-1} \xi(t, q_i^d, q_k^d, q^{d-1})$**:** the expected number of horizontal transition out of state q_i^d to any of the states in level d.

$\sum_{t=1}^{T} \chi(t, q_i^d, q^{d-1})$**:** the expected number of vertical transitions from q^{d-1} to q_i^d.

$\sum_{i=1}^{|q^{d-1}|} \sum_{t=1}^{T-1} \chi(t, q_i^d, q^{d-1})$**:** the expected number of vertical transitions from q^{d-1} to any of its substates in level d.

$\sum_{t=1}^{T} \chi(t, q_i^D, q^{D-1}) + \sum_{t=2}^{T} \gamma_{in}(t, q_i^D, q^{D-1}) = \sum_{t=1}^{T-1} \gamma_{out}(t, q_i^D, q^{D-1})$ the expected number of vertical transitions to the production state q_i^D from state q^{D-1}.

A complete derivation of $\xi, \gamma_{in}, \gamma_{out}$, and χ is given in Appendix A. After the above expectations are calculated from the current parameters, a new set of parameters is re-estimated as follows:

$$\hat{\pi}^{q^1}(q_i^2) = \chi(t, q_i^2, q^1) \ , \tag{1}$$

$$\hat{\pi}^{q^{d-1}}(q_i^d) = \frac{\sum_{t=1}^{T} \chi(t, q_i^d, q^{d-1})}{\sum_{i=1}^{|q^{d-1}|} \sum_{t=1}^{T} \chi(t, q_i^d, q^{d-1})} \quad (2 < d < D) \ , \tag{2}$$

$$\hat{a}_{ij}^{q^{d-1}} = \frac{\sum_{t=1}^{T} \xi(t, q_i^d, q_j^d, q^{d-1})}{\sum_{k=1}^{|q^{d-1}|} \sum_{t=1}^{T} \xi(t, q_i^d, q_k^d, q^{d-1})} = \frac{\sum_{t=1}^{T} \xi(t, q_i^d, q_j^d, q^{d-1})}{\sum_{t=1}^{T} \gamma_{out}(t, q_i^d, q^{d-1})} \ , \tag{3}$$

$$\hat{b}_{q_i^D}^{q^{D-1}}(v_k) = \frac{\sum_{o_t = v_k} \chi(t, q_i^D, q^{D-1}) + \sum_{t>1, o_t = v_k} \gamma_{in}(t, q_i^D, q^{D-1})}{\sum_{t=1}^{T} \chi(t, q_i^D, q^{D-1}) + \sum_{t=2}^{T} \gamma_{in}(t, q_i^D, q^{D-1})} \ . \tag{4}$$

In order to find a good set of parameters we iterate of the expectation step that calculates τ, χ, and the auxiliary path variables, and then we use Equ. (1)-(4) to find a new estimate of the parameters. Although tedious, it is fairly simple to verify that the above steps in this iterative procedure correspond to the Expectation and Maximization steps of the EM algorithm. Hence, this procedure is guaranteed to converge to a stationary point (typically a local maximum) of the likelihood function.

4. Applications

In this section we discuss and give two examples of the use of HHMMs and their parameter estimation for the following complex sequence modeling tasks: building a multi-level structure for English text, and unsupervised identification of repeated strokes in cursive handwriting.

4.1. A multi-level structure for English text

One of the primary goals in stochastic analysis of complex sequences such as natural text is to design a model that captures correlations between events appearing far apart in the sequence. Observable Markov models have been widely used for such tasks (see for instance (Ron et al., 1996) and the references therein). Since the states of these models are constructed based on observable sub-sequences, however,

they cannot capture implicit long-distance statistical correlations. Here we suggest an alternative approach based on HHMMs and give some experimental evidence that this approach is able to partly overcome the above difficulty.

We built two HHMMs and trained them on natural text consisting of classical English stories. The observation alphabet included the lower and upper case letters, blanks, and punctuation marks. For training we used approximately 5000 sentences of an average length of 50 characters. We trained the two HHMMs on exactly the same text. The HHMMs were as follows:

- A shallow HHMM consisting of two levels. The root state of this HHMM had 4 substates. Each of the states at the second level had 7 substates which were all production states. Thus, all the production states were at the same level. The structure of this HHMM is shown at the top part of Figure 3.
- An unbalanced HHMM consisting of three levels. This HHMM had a variable number of substates at each internal state. The structure of this HHMM was unbalanced as it had production states at all levels. Illustrations of the second HHMM are given in Figure 2 and the bottom part of Figure 3.

We applied the generalized forward-backward parameter estimation procedure to the above HHMMs. We found that after training the distributions over strings induced by the substates of the first HHMM greatly resembled the distribution induced by a standard HMM trained on the same data. In contrast, the distribution induced by the second HHMM was substantially different and revealed several interesting phenomena. First, the distribution induced by the second HHMM greatly varied across its different substates. The sets of strings which are most probable to be produced by the each of the states turned to have very little overlap. Second, we observed a multi-scale behavior of the states. Specifically, we found that the most probable strings to be produced by the deep states roughly correspond to phonetic units, namely, strings such as `ing`, `th`, `wh`, and `ou`. Going up the hierarchy, the states at the second and third level produce strings which are frequent words and phrases such as: `is not`, `will` and `where`. Finally, at the top of the hierarchy, the root state induced a distribution that corresponds to a sentence scale. For instance, the strings produced by the root state (and hence the entire HHMM) are likely to end with a punctuation mark. We also found that the horizontal transition probabilities at the end of training of the unbalanced HHMM got highly peaked. This reflects strong Markov dependencies between states at the same level. Thus, not only is the distribution induced by each state highly concentrated on few strings, but also the set of strings that can be generated by recursive activations of the deep HHMM is strongly biased towards syntactic structures that frequently appear in natural texts. As we demonstrate in the next application, the trained HHMM or any of its submodels can be now used as a building block in more complex tasks such as text classification. These highly biased distributions are illustrated in Figures 2 and 3. In Figure 2 we give the horizontal transition probabilities at the beginning and the end of the training. In Figure 3 we list the most probable strings produced by each substate of the two HHMMs: the deep unbalanced HHMM at the bottom

and the shallow balanced HHMM at the top. It is clear from the figure that the richer model developed a much larger variety of strings which include whole words and fragments of sentences.

4.2. Unsupervised learning of cursive handwriting

In (Singer and Tishby, 1994), a dynamic encoding scheme for cursive handwriting based on an oscillatory model of handwriting was proposed and analyzed. This scheme performs an inverse mapping from continuous pen trajectories to strings over a discrete set of symbols which efficiently encode cursive handwriting. These symbols are named motor control commands. The motor control commands can be transformed back into pen trajectories using a generative model, and the handwriting can be reconstructed without the noise that is eliminated by the dynamic encoding scheme. Each possible control command is composed of a Cartesian product of the form $\mathbf{x} \times \mathbf{y}$ where $\mathbf{x}, \mathbf{y} \in \{0, 1, 2, 3, 4, 5\}$, hence the alphabet consists of 36 different symbols. These symbols represent quantized horizontal and vertical amplitude modulation and their phase-lags.

Different Roman letters map to different sequences over the above symbols. Moreover, since there are different writing styles and due to the existence of noise in the human motor system, the same cursive letter can be written in many different ways. This results in different encodings that represent the same cursively written word. A desirable first step in a system that analyzes and recognizes cursive scripts is to build stochastic models that approximate the distribution of the sequences that correspond to complete cursive pen-trajectories. We used the motor control commands as the observation alphabet and built HHMMs corresponding to different cursive words in the training set. For example, we used 60 examples of the word `maintain` to estimate the parameters of an HHMM which had five levels. This HHMM had an unbalanced structure and it had production states at all levels. In the design of the topology of the HHMMs we took into account additional knowledge such as repetitions of letters and combination of letters in cursively written words. The structure of the HHMM used for the word `maintain` is shown in Figure 4. We used the generalized forward-backward algorithm to estimate the parameters of the HHMM. We then used the trained HHMM to identify repeated strokes that represent combination of letters in cursive handwriting.

In order to verify that the resulting HHMMs indeed learned the distribution and the internal structure of the words, we used the generalized Viterbi algorithm for HHMMs to perform multi-scale segmentation of the motor control sequences. An example result of such a segmentation is given in Figure 5. In the figure the cursive word `maintain`, reconstructed from the motor control commands, is shown together with its hierarchical segmentation. We used the trained HHMM of depth five whose structure is shown in Figure 4 to segment the word.

In Figure 5 we show the temporal segmentation into states from the first two hierarchies of the HHMM. It is clear from the figure that the gerneralized Viterbi algorithm assigned different states of the HHMM to different cursive strokes. Fur-

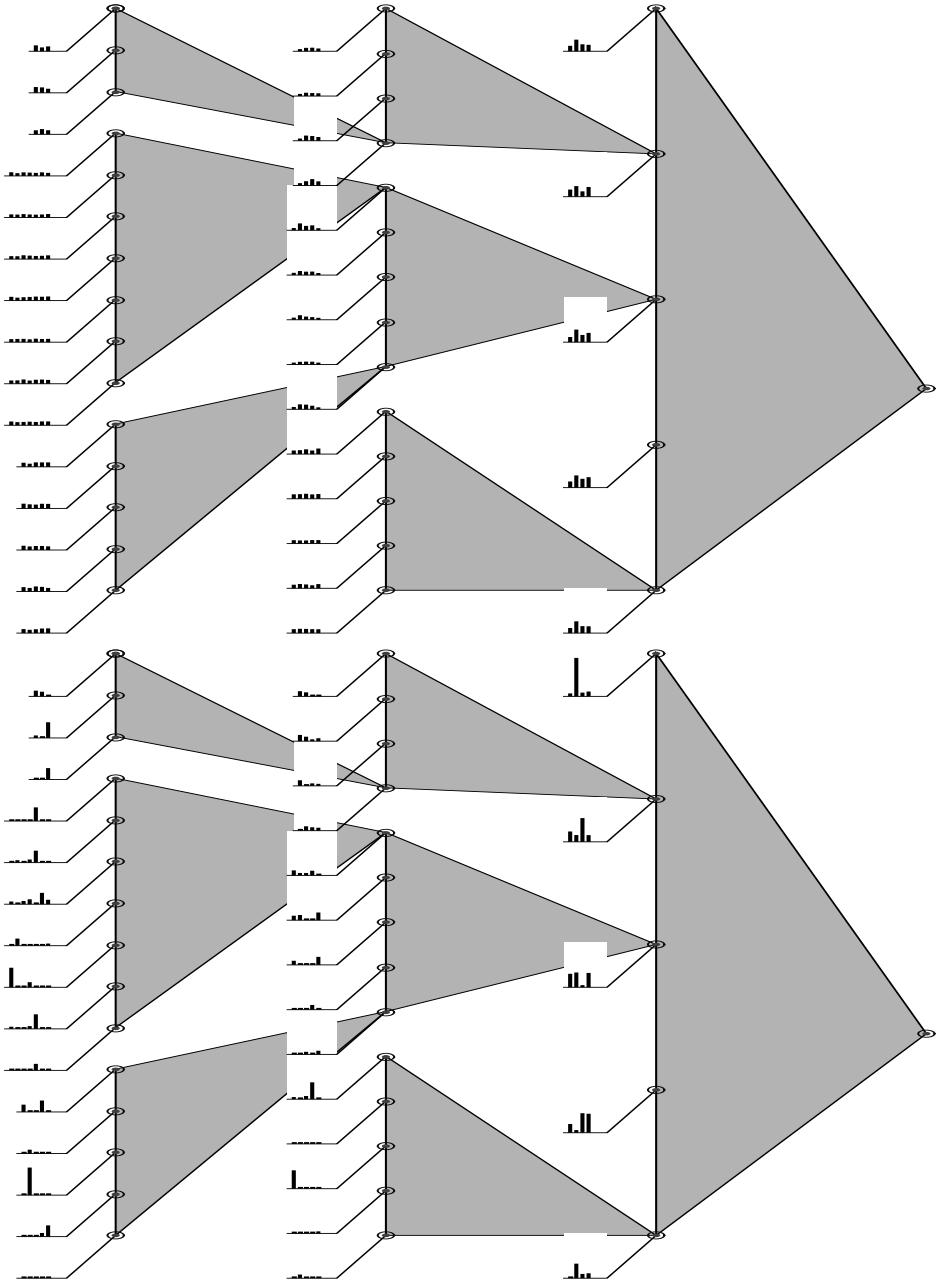

Figure 2. The transition distribution at the beginning (top) and the end of the training for an unbalanced HHMM of depth 3 that was trained on English texts. While the initial distribution is almost uniform, the final distribution is sharply peaked around different states at different levels.

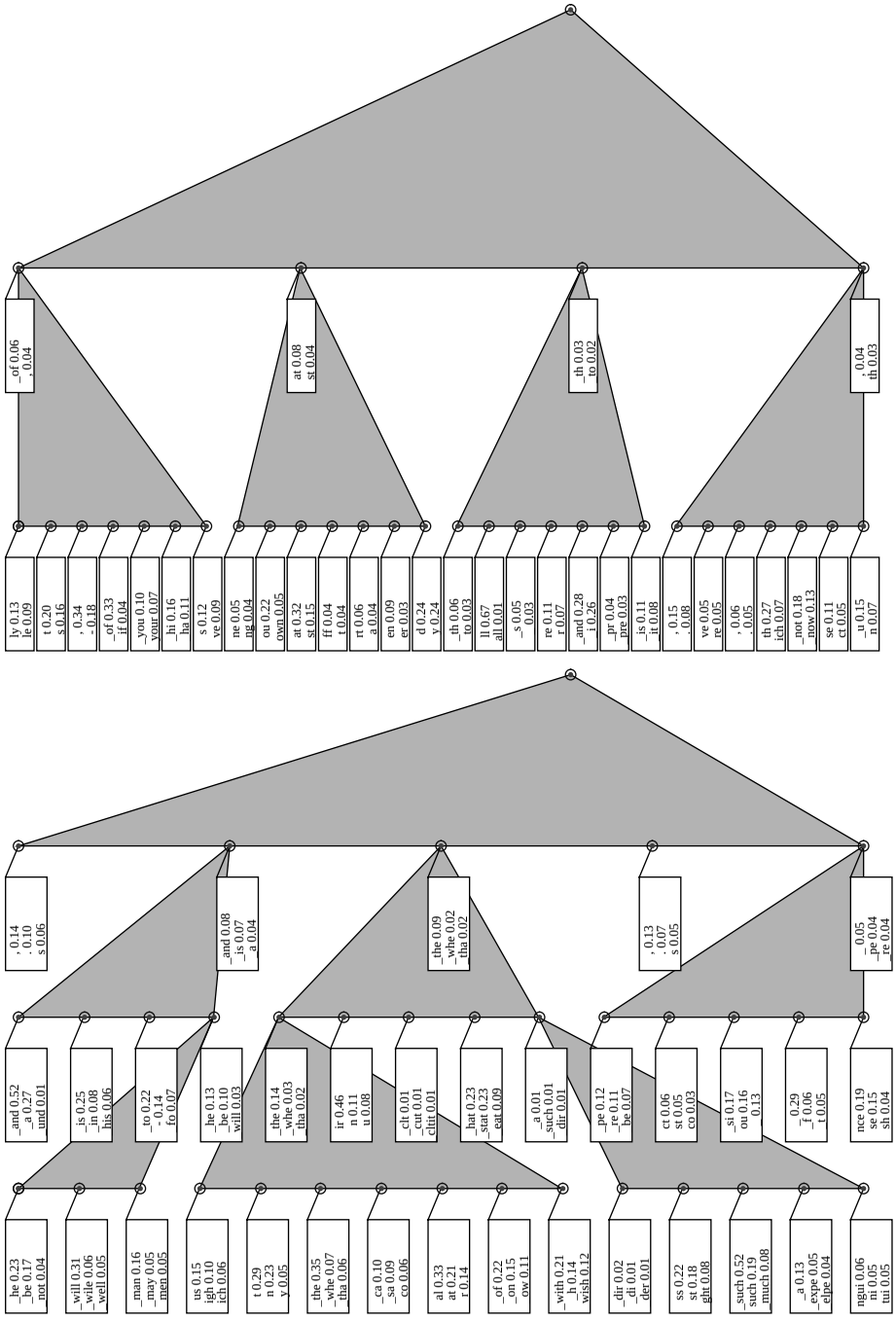

Figure 3. The most probable strings to be generated at each state for two different HHMMs.

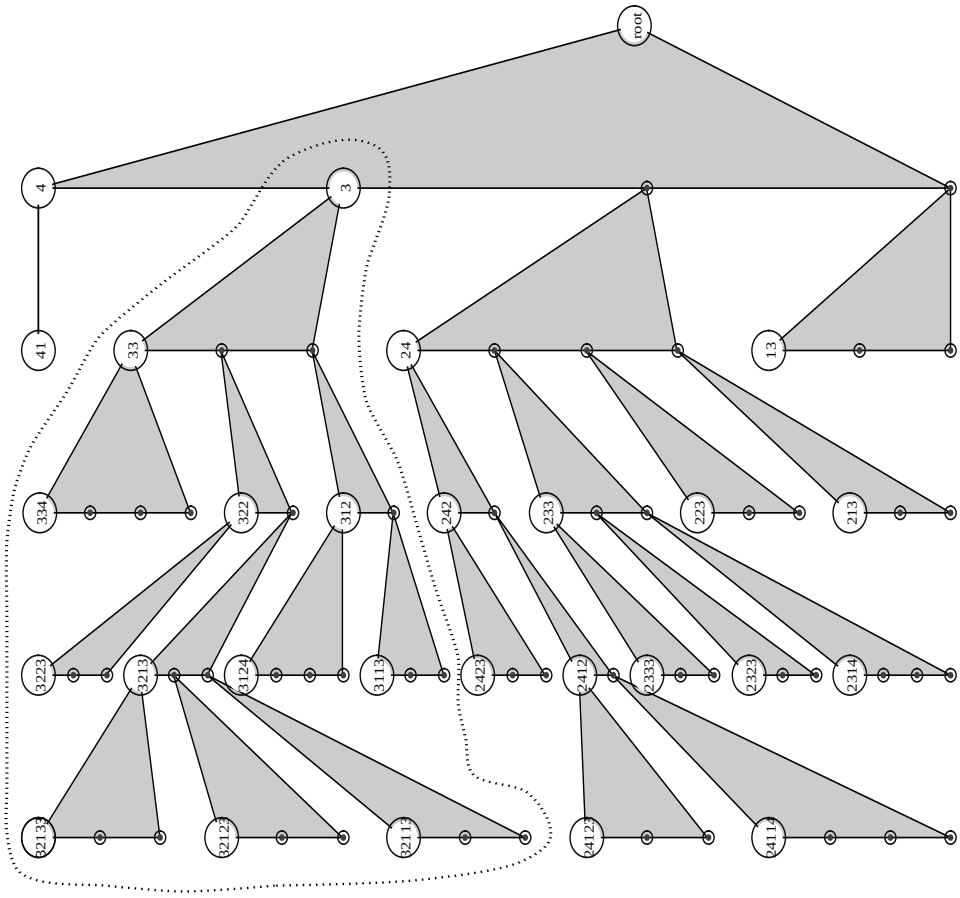

Figure 4. The HHMM used in the cursive handwriting spotting experiments. The full model was trained on complete words. The submodel denoted by a dotted line was used to locate the occurrences of the letter combination `ai`.

thermore, the same state is consistently used to generate strokes that represent the same letter combination. For instance, state 3 in the first hierarchy is responsible for producing the combination `ai`. Its substates further split the letters `ai` into sub-strokes: the first substate, denoted by 3_1, generates the first part of the cursive letter `a`, the second, 3_2, generates the middle part (the articulated stroke connecting the letters `a` and `i`), and the third, 3_3, generates the letter `i`. Similar phenomena can also be observed in other states of the HHMM.

A common question that arises in stochastic modeling of sequences such as speech signals and handwritten text is what are the 'natural' units that constitute the sequences. A widely used approach is to manually define these units, e.g. phonemes

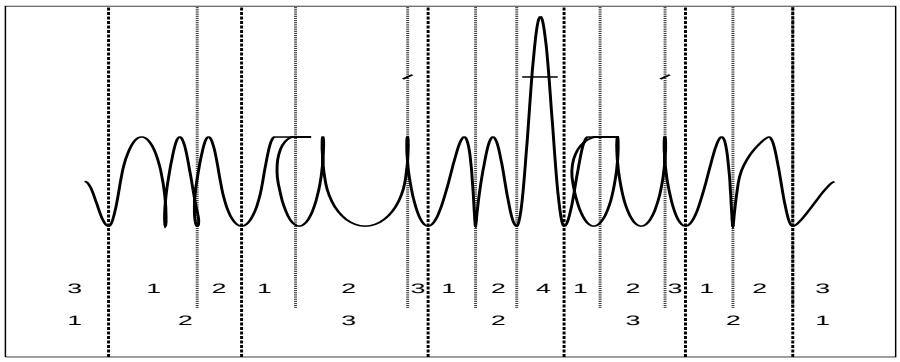

Figure 5. Hierarchical segmentation of the word `maintain` obtained by the Viterbi algorithm for HHMMs. The temporal segmentation according to the first two levels of the HHMM is shown. The word was reconstructed from its encoded dynamical representation.

in spoken language, letters in written text, etc. There are several drawbacks to this approach: it requires a manual segmentation and it does not take into account the temporal interaction (such as co-articulation in speech) between consecutive 'units'. We now propose and briefly demonstrate an alternative approach that uses the substates of a trained HHMM to provide a partial answer to the above question. Due to the self-similar structure of HHMMs we can use each substate as an autonomous model. We used the substates at the second level of the HHMM described above and calculated the probabilities they induce for each sub-sequence of an observation sequence. We also defined a simple HMM that induce a uniform distribution over all the possible symbols. This simple model, denoted by U, serves as a null hypothesis and competes against submodels that were pulled out from the full HHMM. The probability of a subsequence $\bar{O}$ to be generated by a submodel M compared to the null hypothesis (assuming an equal prior for the alternatives) is, $P(M|\bar{O}) = \frac{P(\bar{O}|M)}{P(\bar{O}|M)+P(\bar{O}|U)}$. High values of $P(M|\bar{O})$ indicate the occurrence of the letters that correspond to the pulled-out model M. Hence, thresholding this value can be used to identify the locations of combination of letters in unsegmented data. An example result of letter combination spotting using an HHMM is given in Figure 6. In the figure we show the logarithm of the conditional probability $P(M|\bar{O})$ normalized by the length of the string $|\bar{O}|$. This probability was calculated over all possible start locations. The submodel M corresponding to the combination `ai`, that is, the submodel rooted at state 3, denoted by a dotted line in Figure 4, was pulled out from the HHMM which was constructed for the word `maintain`. Clearly, all the occurrences of the letters `ai` were correctly located. The combination of letters `oi` in the word `pointers` also received a high likelihood. This and other ambiguities can be resolved by further refining the set of submodels and by employing a higher level stochastic language model.

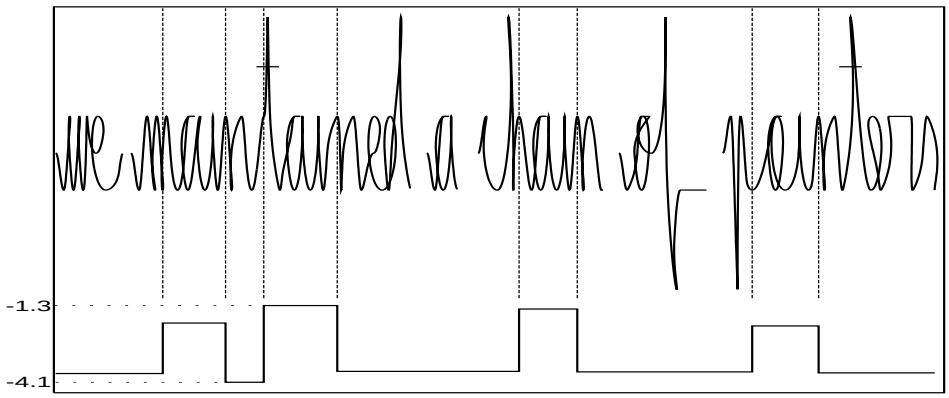

Figure 6. Spotting the occurrences of the letters `ai` in the sentence `we maintained a chain of pointers`. The submodel corresponding to this combination of letters was pulled out from the HHMM shown in Figure 5 which was built for the word `maintain`. Occurrences of the letters are found by letting the model compete against a null hypothesis that induces a uniform distribution over all possible symbols.

5. Conclusions

Hierarchical hidden Markov models are a generalization of HMMs which provide a partial answer to two fundamental problems that arise in complex sequence modeling. First, HHMMs are able to correlate structures accuring relatively far apart in observation sequences, while maintaining the simplicity and computational tractability of simple Markov processes. Second, they are able to handle statistical inhomogeneities common in speech and natural language. The maximum likelihood parameter estimation procedure and the Viterbi most probable state decoding, are both naturally generalized to this richer structure. However, there is still a missing component: HHMMs lack the ability to adapt their topology, that is, to allow for self-organized merging and growth of the submodels. There are also several natural generalization of HHMMs. For instance, using the framework introduced by Bengio and Frasconi (1995) for input-output HMMs, hierarchical HMMs can be generalized to describe input-output mappings between strings over two different alphabets.

The experiments with HHMMs described in this paper are an initial step towards a better understanding of hierarchical stochastic models for natural complex sequences. There are other models, such as factorial hidden Markov models (Ghahramani and Jordan, 1997) and alternative parameter estimation techniques (Singer and Warmuth, 1997) that can be used. Understanding the connections between these different approaches and conducting a formal analysis of hierarchical stochastic modeling is an important research direction that is now in progress.

Appendix A
Generalized forward-backward algorithm

To remind the reader, we calculate four path variables in the expectation step: α, β, χ, and ξ, which informally correspond to 'forward','backward', 'downward', and 'upward' stochastic transitions in the given HHMM. The variables α and β are calculated in a bottom-up manner since the probability induced by a state q depends only on the substates that belong to the tree (submodel) rooted at q. Given the variables α and β, the variables χ and ξ are calculated in a top-down manner. To simplify the derivation of the path variables, we also define two auxiliary variables, η_{in} and η_{out}. We now give a detailed derivation of these variables.

Definition:

$$\alpha(t, t+k, q_i^d, q^{d-1}) = P(o_t \cdots o_{t+k},\ q_i^d \text{ finished at } t+k \mid q^{d-1} \text{ started at } t)$$

Estimation:

$$\alpha(t, t, q_i^D, q^{D-1}) = \pi^{q^{D-1}}(q_i^D)\ b^{q_i^{D-1}}(o_t)$$

$$\alpha(t, t+k, q_i^D, q^{D-1}) = \left[\sum_{j=1}^{|q^{D-1}|} \alpha(t, t+k-1, q_j^D, q^{D-1})\, a_{ji}^{q^{D-1}}\right] b^{q_i^{D-1}}(o_{t+k})$$

$$\alpha(t, t, q_i^d, q^{d-1}) = \pi^{q^{d-1}}(q_i^d) \left[\sum_{s=1}^{|q_i^d|} \alpha(t, t, q_s^{d+1}, q_i^d)\, a_{s\ end}^{q_i^d}\right]$$

$$\begin{aligned}\alpha(t, t+k, q_i^d, q^{d-1}) = {} & \sum_{l=0}^{k-1} \left[\sum_{j=1}^{|q^{d-1}|} \alpha(t, t+l, q_j^d, q^{d-1})\, a_{ji}^{q^{d-1}}\right] \\ & \left[\sum_{s=1}^{|q_i^d|} \alpha(t+l+1, t+k, q_s^{d+1}, q_i^d)\, a_{s\ end}^{q_i^d}\right] \\ & + \pi^{q^{d-1}}(q_i^d) \left[\sum_{s=1}^{|q_i^d|} \alpha(t, t+k, q_s^{d+1}, q_i^d)\, a_{s\ end}^{q_i^d}\right]\end{aligned}$$

Definition:

$$\beta(t, t+k, q_i^d, q^{d-1}) = P(o_t \cdots o_{t+k} \mid q_i^d \text{ started at } t,\ q^{d-1} \text{ finished at } t+k)$$

Estimation:

$$\beta(t, t, q_i^D, q^{D-1}) = b^{q_i^{D-1}}(o_t)\ a_{i\ end}^{q^{D-1}}$$

$$\beta(t, t+k, q_i^D, q^{D-1}) = b^{q_i^{D-1}}(o_t) \left[\sum_{j \neq end}^{|q^{D-1}|} a_{ij}^{q^{D-1}}\, \beta(t+1, t+k, q_j^D, q^{D-1})\right]$$

$$\begin{aligned}
\beta(t,t,q_i^d,q^{d-1}) &= \left[\sum_{s=1}^{|q_i^d|}\pi^{q_i^d}(q_s^{d+1})\ \beta(t,t,q_s^{d+1},q_i^d)\right] a_{i\,end}^{q_i^d} \\
\beta(t,t+k,q_i^d,q^{d-1}) &= \sum_{l=0}^{k-1}\left[\sum_{s=1}^{|q_i^d|}\pi^{q_i^d}(q_s^{d+1})\,\beta(t,t+l,q_s^{d+1},q_i^d)\right] \\
&\qquad \left[\sum_{j=1}^{|q^{d-1}|} a_{ij}^{q^{d-1}}\ \beta(t+l+1,t+k,q_j^d,q^{d-1})\right] \\
&+ \left[\sum_{s=1}^{|q_i^d|}\pi^{q_i^d}(q_s^{d+1})\,\beta(t,t+k,q_s^{d+1},q_i^d)\right] a_{i\,end}^{q^{d-1}}
\end{aligned}$$

Definition:

$$\eta_{in}(t,q_i^d,q^{d-1}) = P(o_1\cdots o_{t-1},\ q_i^d \text{ started at } t|\ \lambda)$$

Estimation:

$$\begin{aligned}
\eta_{in}(1,q_i^2,q^1) &= \pi^{q^1}(q_i^2) \\
\eta_{in}(t,q_i^2,q^1) &= \sum_{j=1}^{|q^1|}\alpha(1,t-1,q_j^2,q^1)\ a_{ji}^{q^1} \qquad (1<t) \\
\eta_{in}(1,q_i^d,q_l^{d-1}) &= \eta_{in}(1,q_l^{d-1},q^{d-2})\ \pi^{q_l^{d-1}}(q_i^d) \\
\eta_{in}(t,q_i^d,q_l^{d-1}) &= \sum_{t'=1}^{t-1}\eta_{in}(t',q_l^{d-1},q^{d-2})\left[\sum_{j=1}^{|q_l^{d-1}|}\alpha(t',t-1,q_j^d,q_l^{d-1})\,a_{ji}^{q_l^{d-1}}\right] \\
&+ \eta_{in}(t,q_l^{d-1},q^{d-2})\,\pi^{q_l^{d-1}}(q_i^d) \qquad (1<t)
\end{aligned}$$

Definition:

$$\eta_{out}(t,q_i^d,q^{d-1}) = P(q_i^d \text{ finished at } t,\ o_{t+1}\cdots o_T|\ \lambda)$$

Estimation:

$$\begin{aligned}
\eta_{out}(t,q_i^2,q^1) &= \sum_{j=1}^{|q^1|} a_{ij}^{q^1}\,\beta(t+1,T,q_j^2,q^1) \qquad (t<T) \\
\eta_{out}(t,q_i^d,q_l^{d-1}) &= \sum_{k=t+1}^{T}\left[\sum_{j=1}^{|q_l^{d-1}|} a_{ij}^{q_l^{d-1}}\,\beta(t+1,k,q_j^d,q_l^{d-1})\right]\eta_{out}(k,q_l^{d-1},q^{d-2}) \\
&+ a_{i\,end}^{q_l^{d-1}}\,\eta_{out}(t,q_l^{d-1},q^{d-2}) \qquad (t<T) \\
\eta_{out}(T,q_i^d,q_l^{d-1}) &= a_{i\,end}^{q_l^{d-1}}\,\eta_{out}(T,q_l^{d-1},q^{d-2})
\end{aligned}$$

Definition:

$$\begin{aligned}\xi(t, q_i^d, q_j^d, q^{d-1}) &= P(q_i^d \text{ finished at } t,\ q_j^d \text{ started at } t+1 \,|\, \lambda, \bar{O}) \\ &= P(o_1 \cdots o_t,\ q_i^d \longrightarrow q_j^d,\ o_{t+1} \cdots o_T \,|\, \lambda, \bar{O})\end{aligned}$$

Estimation:

$$\xi(t, q_i^2, q_j^2, q^1) = \frac{\alpha(1, t, q_i^2, q^1)\, a_{ij}^{q^1}\, \beta(t+1, T, q_j^2, q^1)}{P(\bar{O}|\lambda)} \qquad (t < T)$$

$$\xi(T, q_i^2, q_j^2, q^1) = \frac{\alpha(1, T, q_i^2, q^1)\, a_{ij}^{q^1}}{P(\bar{O}|\lambda)}$$

$$\begin{aligned}\xi(t, q_i^d, q_j^d, q_l^{d-1}) &= \frac{1}{P(\bar{O}|\lambda)} \left[\sum_{s=1}^{t} \eta_{in}(s, q_l^{d-1}, q^{d-2})\, \alpha(s, t, q_i^d, q_l^{d-1})\right] a_{ij}^{q_l^{d-1}} \\ &\quad \left[\sum_{e=t+1}^{T} \beta(t+1, e, q_j^d, q_l^{d-1})\, \eta_{out}(e, q_l^{d-1}, q^{d-2})\right] \quad (t < T)\end{aligned}$$

$$\begin{aligned}\xi(t, q_i^d, q_{end}^d, q_l^{d-1}) &= \frac{1}{P(\bar{O}|\lambda)} \left[\sum_{s=1}^{t} \eta_{in}(s, q_l^{d-1}, q^{d-2})\, \alpha(s, t, q_i^d, q_l^{d-1})\right] \\ &\quad a_{i\,end}^{q_l^{d-1}}\, \eta_{out}(t, q_l^{d-1}, q^{d-2}) \qquad (t < T)\end{aligned}$$

Definition:

$$\begin{aligned}\chi(t, q_i^d, q^{d-1}) &= P(q_i^d \text{ started at } t \,|\, \lambda, \bar{O}) \\ &= P(o_1 \cdots o_{t-1},\ \overset{q^{d-1}}{\underset{q_i^d}{\downarrow}},\ o_t \cdots o_T \mid \lambda, \bar{O})\end{aligned}$$

Estimation:

$$\chi(1, q_i^2, q^1) = \frac{\pi^{q^1}(q_i^2)\ \ \beta(1, T, q_i^2, q^1)}{P(\bar{O}|\lambda)}$$

$$\begin{aligned}\chi(t, q_i^d, q_l^{d-1}) &= \frac{\eta_{in}(t, q_l^{d-1}, q^{d-2})\, \pi^{q_l^{d-1}}(q_i^d)}{P(\bar{O}|\lambda)} \\ &\quad \left[\sum_{e=t}^{T} \beta(t, e, q_i^d, q_l^{d-1})\, \eta_{out}(e, q_l^{d-1}, q^{d-2})\right] \quad (2 < d)\end{aligned}$$

Appendix B
Generalized Viterbi algorithm

To remind the reader, for each pair of states (q^{d-1}, q_i^d) we keep three variables:

- $\delta(t, t+k, q_i^d, q^{d-1})$ is the likelihood of the most probable state sequence generating $o_t \cdots o_{t+k}$ assuming it was solely generated by a recursive activation that started at time step t from state q^{d-1} and ended at q_i^d which returned to q^{d-1} at time step $t+k$.
- $\psi(t, t+k, q_i^d, q^{d-1})$ is the index of the most probable state to be activated by q^{d-1} before q_i^d. If such a state does not exist ($o_t \cdots o_{t+k}$ was solely generated by q_i^d) we set $\psi(t, t+k, q_i^d, q^{d-1}) \stackrel{\text{def}}{=} 0$.
- $\tau(t, t+k, q_i^d, q^{d-1})$ is the time step at which q_i^d was most probable to be called by q^{d-1}. If q_i^d generated the entire subsequence we set $\psi(t, t+k, q_i^d, q^{d-1}) = t$.

To simplify our notation, we define the functional $\mathcal{MAX}$ whose parameters are a function f and a finite set S,

$$\mathcal{MAX}_{l \in S} \{f(l)\} \stackrel{\text{def}}{=} \left(\max_{l \in S} \{f(l)\}, \ \arg\max_{l \in S} \{f(l)\} \right) \quad .$$

The generalized Viterbi algorithm starts from the production states and calculate δ, ψ, and τ in a bottom up manner as follows.

Production states:

1. Initialization:

$$\delta(t, t, q_i^D, q^{D-1}) = \pi^{q^{D-1}}(q_i^D)\ b^{q_i^D}(o_t) \quad \psi(t, t, q_i^D, q^{D-1}) = 0 \quad \tau(t, t, q_i^D, q^{D-1}) = t$$

2. Recursion:

$$\left(\delta(t, t+k, q_i^D, q^{D-1}),\ \psi(t, t+k, q_i^D, q^{D-1})\right) = \mathcal{MAX}_{1 \le j \le |q^{D-1}|} \left\{ \delta(t, t+k-1, q_j^D, q^{D-1})\ a_{ji}^{q^{D-1}}\ b^{q_i^D}(o_{t+k}) \right\}$$

$$\tau(t, t+k, q_i^D, q^{D-1}) = t+k$$

Internal states:

1. Initialization:

$$\delta(t, t, q_i^d, q^{d-1}) = \max_{1 \le j \le |q_i^d|} \left\{ \pi^{q^{d-1}}(q_i^d)\ \delta(t, t, q_r^{d+1}, q_i^d)\ a_{r\,\text{end}}^{q_i^d} \right\}$$

$$\psi(t, t, q_i^d, q^{d-1}) = 0 \qquad \tau(t, t, q_i^d, q^{d-1}) = t$$

2. Recursion:

   (A) For $t' = t+1, \ldots, t+k$ set:

$$R = \max_{1 \le r \le |q_i^d|} \left\{ \delta(t', t+k, q_r^{d+1}, q_i^d)\, a_{r\,\mathrm{end}}^{q_i^d} \right\}$$

$$(\Delta(t'), \Psi(t')) = \mathcal{MAX}_{1 \le j \le |q^{d-1}|} \left\{ \delta(t, t'-1, q_j^d, q^{d-1})\, a_{ji}^{q^{d-1}}\, R \right\}$$

   (B) For t set:

$$\Delta(t) = \pi^{q^{d-1}}(q_i^d) \max_{1 \le r \le |q_i^{d-1}|} \left\{ \delta(t, t+k, q_r^{d+1}, q_i^d)\, a_{r\,\mathrm{end}}^{q_i^d} \right\}$$

$$\Psi(t) = 0$$

   (C) Find the most probable switching time:

$$\left(\delta(t, t+k, q_i^d, q^{d-1}),\ \tau(t, t+k, q_i^d, q^{d-1})\right) = \mathcal{MAX}_{t \le t' \le t+k}\, \Delta(t')$$

$$\psi(t, t+k, q_i^d, q^{d-1}) = \Psi\left(\tau(t, t+k, q_i^d, q^{d-1})\right)$$

Finally, the probability of the most probable state sequence is found as follows,

$$\left(P^\star,\ q^2_{\mathrm{last}}\right) = \mathcal{MAX}_{q_i^2} \left\{ \delta(1, T, q_i^2, q^1) \right\} \quad ,$$

and the most probable states sequence itself is found by scanning the lists ψ and τ starting from $\tau(1, T, q^2_{\mathrm{last}}, q^1)$ and $\psi(1, T, q^2_{\mathrm{last}}, q^1)$.

Appendix C

Table C.1. List of symbols and variables

Symbol	Definition	Section
Σ	finite alphabet	2
$\bar{O} = o_1 o_2 \ldots o_T \quad (o_i \in \Sigma)$	observation sequence	2
$d \in \{1, \ldots, D\}$	hierarchy depth	2
q_i^d	the ith substate at level d	2
$\Pi^{q^d} = \{\pi^d(q_i^{d+1})\}$ $= \{P(q_i^{d+1} \mid q^d)\}$	initial substate distribution	2
$A^{q^d} = \{a_{ij}^{q^d}\}$ $= \{P(q_j^{d+1} \mid q_i^{d+1})\}$	substate transition probabilities	2
$B^{q_i^D} = \{b^{q_i^D}(k)\}$ $= \{P(\sigma_k \mid q_i^D)\}$	output probability distribution	2
$\lambda = \{\{A^{q^d}\}\{\Pi^{q^d}\}\{B^{q^D}\}\}$	HHMM's set of parameters	2
$\alpha(t, t+k, q_i^d, q^{d-1})$	$P(o_t \cdots o_{t+k},\ q_i^d$ finished at $t+k \mid$ q^{d-1} started at $t)$	3.1
$\beta(t, t+k, q_i^d, q^{d-1})$	$P(o_t \cdots o_{t+k} \mid q_i^d$ started at t, q^{d-1} finished at $t+k)$	3.1
$\xi(t, q_i^d, q_j^d, q^{d-1})$	$P(o_1 \cdots o_t,\ q_i^d \longrightarrow q_j^d,\ o_{t+1} \cdots o_T \mid \lambda)$	3
$\gamma_{in}(t, q_i^d, q^{d-1})$	$\gamma_{in}(t, q_i^d, q^{d-1}) = \sum_{k=1}^{\lvert q^{d-1} \rvert} \xi(t-1, q_k^d, q_i^d, q^{d-1})$	3
$\gamma_{out}(t, q_i^d, q^{d-1})$	$\gamma_{out}(t, q_i^d, q^{d-1}) = \sum_{k=1}^{\lvert q^{d-1} \rvert} \xi(t, q_i^d, q_k^d, q^{d-1})$	3
$\chi(t, q_i^d, q^{d-1})$	$P(o_1 \cdots o_{t-1},\ \begin{matrix} q^{d-1} \\ \downarrow \\ q_i^d \end{matrix},\ o_t \cdots o_T \mid \lambda)$	3
$\eta_{in}(t, q_i^d, q^{d-1})$	$P(o_1 \cdots o_{t-1},\ q_i^d$ started at $t \mid \lambda)$	A
$\eta_{out}(t, q_i^d, q^{d-1})$	$P(q_i^d$ finished at $t,\ o_{t+1} \cdots o_T \mid \lambda)$	A
$\delta(t, t+k, q_i^d, q^{d-1})$ $\psi(t, t+k, q_i^d, q^{d-1})$ $\tau(t, t+k, q_i^d, q^{d-1})$	value (δ), state-list (ψ), transition times (τ) of the most probable generation of $o_t \cdots o_{t+k}$ started by q^{d-1} at t and ended by q_i^d at $t+k$	B

Acknowledgments

We thank Yoshua Bengio and Raj Iyer for helpful comments.

References

N. Abe and M. Warmuth. On the computational complexity of approximating distributions by probabilistic automata. *Machine Learning*, 9:205–260, 1992.

P. Baldi, Y. Chauvin, T. Hunkapiller, and M. McClure. Hidden Markov models of biological primary sequence information. *Proc. Nat. Acd. Sci. (USA)*, 91(3):1059–1063, 1994.

L.E Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, Vol. 37, 1966.

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

T. Cover and J. Thomas. *Elements of Information Theory*. Wiley, 1991.

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

Z. Ghahramani and M.I. Jordan. Factorial hidden Markov models. *Machine Learning*, 1997 (to appear).

D. Gillman and M. Sipser. Inference and minimization of hidden Markov chains. *Proceedings of the Seventh Annual Workshop on Computational Learning Theory*, pages 147–158, 1994.

F. Jelinek. Robust part-of-speech tagging using a hidden Markov model. *IBM T.J. Watson Research Technical Report*, 1985.

F. Jelinek. Markov source modeling of text generation. Technical report, *IBM T.J. Watson Research Center Technical Report*, 1983.

F. Jelinek. Self-organized language modeling for speech recognition. *IBM T.J. Watson Research Center Technical Report*, 1985.

A. Krogh, S.I. Mian, D. Haussler. A hidden Markov model that finds genes in E.coli DNA. NAR, 22"4768-4778, 1994.

K. Lari and S.J. Young. The estimation of stochastic context free grammars using the Inside-Outside algorithm. *Computers Speech and Language*, 4, 1990.

R. Nag, K.H. Wong, F. Fallside. Script recognition using hidden Markov models. *Proceedings of International Conference on Acoustics Speech and Signal Processing*, pages 2071–2074, 1985.

L.R. Rabiner and B.H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, No. 3, 1986.

L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 1989.

J. Rissanen. Complexity of strings in the class of Markov sources. *IEEE Transactions on Information Theory*, 32(4):526–532, 1986.

D. Ron, Y. Singer, N. Tishby. The power of amnesia: learning probabilistic automata with variable memory length. *Machine Learning*, 25:117–149, 1996.

Y. Singer and N. Tishby. Dynamical encoding of cursive handwriting. *Biological Cybernetics*, 71(3):227-237, 1994.

Y. Singer and M.K. Warmuth. Training algorithms for hidden Markov models using entropy based distance functions. In M.C. Mozer, M.I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 641–647. MIT Press, Cambridge, MA, 1997.

A. Stolcke and S.M. Omohundro. Best-first model merging for hidden Markov model induction. *Technical Report ICSI TR-94-003*, 1994.

A.J. Viterbi. Error bounds for convulutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269, 1967.