Coupled hidden Markov models for modeling interacting processes

Matthew Brand
The Media Lab, MIT
20 Ames Street
Cambridge, MA 02139 USA
brand@media.mit.edu
http://www.media.mit.edu/~brand

Abstract

We present methods for coupling hidden Markov models (HMMs) to model systems of multiple interacting processes. The resulting models have multiple state variables that are temporally coupled via matrices of conditional probabilities. We introduce a deterministic $O(T(CN)^2)$ approximation for maximum *a posterior* (MAP) state estimation which enables fast classification and parameter estimation via expectation maximization. An "N-heads" dynamic programming algorithm samples from the highest probability paths through a compact state trellis, minimizing an upper bound on the cross entropy with the full (combinatoric) dynamic programming problem. The complexity is $O(T(CN)^2)$ for C chains of N states apiece observing T data points, compared with $O(TN^{2C})$ for naive (Cartesian product), exact (state clustering), and stochastic (Monte Carlo) methods applied to the same inference problem. In several experiments examining training time, model likelihoods, classification accuracy, and robustness to initial conditions, coupled HMMs compared favorably with conventional HMMs and with energy-based approaches to coupled inference chains. We demonstrate and compare these algorithms on synthetic and real data, including interpretation of video.

1. Introduction

Hidden Markov models (HMMs) are a popular probabilistic framework for modeling processes that have structure in time. They have a clear Bayesian semantics, efficient algorithms for state and parameter estimation, and they automatically perform dynamic time warping for signals that are locally squashed and stretched. An HMM is essentially a quantization of a system's configuration space into a small number of discrete states, together with probabilities for transitions between states. A single finite discrete variable indexes the current state of the system. Any information about the history of the process needed for future inferences must be reflected in the current value of this state variable. However, many interesting systems are composed of multiple interacting processes, and thus merit a compositional representation of two or more variables. This is typically the case for systems that have structure both in time and space. With a single state variable, Markov models are ill-suited to these problems.

We present methods for coupling hidden Markov models to capture these interactions. Chains are coupled via matrices of conditional probabilities modeling causal (temporal) influences between their hidden state variables. Coupled HMMs are densely connected inference graphs for which no efficient exact inference algorithms are known, principally because the implicit state space is exponential in the number of state variables We introduce a deterministic approximation for maximum *a posterior* (MAP) state estimation which enables fast classification and parameter estimation via expectation maximization. We obtain an upper bound on the cross entropy with the full (combinatoric) posterior which can be minimized using a subspace that is linear in the number of state variables. An "N-heads" dynamic programming algorithm samples from the $O(N)$ highest probability paths through a compacted state trellis, with complexity $O(T(CN)^2)$ for C chains of N states apiece observing T data points. For interesting cases with limited couplings the complexity falls further to $O(TCN^2)$. Naive (Cartesian product), exact (state clustering), and stochastic (Monte Carlo) approaches to coupled HMM inference problems involve a combinatoric number of states, typically requiring $O(TN^{2C})$ computations. Much HMM modeling practice is in fact an optimization of the naive strategy, which by definition fails to model interprocess interactions and usually consists of retreats from state counts that overfit the data. We also present a maximum-entropy projection between coupled HMMs and Cartesian product HMMs which also supports estimation and classification. N-heads and projected HMMs compare favorably conventional HMMs and with reduced complexity energy-coupling methods such as mean field approximations. We demonstrate and compare these algorithms on synthetic and real data, including interpretation of video. In several hundred experiments with real and simulated data, CHMMs trained fastest, modeled best, classified most accurately, and performed most robustly in sensitivity analyses. The N-heads algorithm also readily lends itself to variations such as chains with different state structures, time scales, and degrees of influence on each other, and even superimposed outputs.

2. HMMs and Markov representation

We begin with a short review of hidden Markov models. Readers familiar with the model may wish to skip to section 3 on page 5, but will find the key in figure 1 useful. An extended tutorial is available in [Rabiner, 1989]. An HMM is a defined by a set of discrete states $S = \{a_1, a_2, a_3, \ldots, a_N\}$; a time indexed discrete variable $s_t \in S$; state-to-state transition probabilities $P_{i|j} \doteq P_{s_t=a_i|s_{t-1}=a_j}$; prior probabilities for the first state $P_i \doteq P_{s_1=a_i}$; and output probabilities for each state $p_i(o_t) \doteq p_{s_t=a_i}(o_t)$. The output probabilities are usually drawn from a continuous exponential distribution such as a Gaussian $\mathcal{N}(x; \mu_i, \Sigma_i) \doteq (2\pi)^{-d/2} e^{(x-\mu_i)\Sigma_i^{-1}(x-\mu_i)^T} / \sqrt{|\Sigma_i|}$.

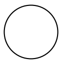
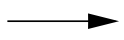
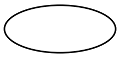
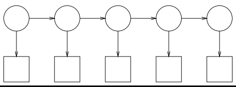
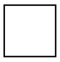
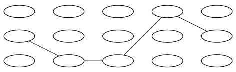
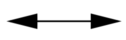
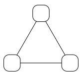
	a. state variable		e. conditional probability (table or exponential p.d.f.)
	b. state label		f. probabilistic inference graph for 5 time slices of an HMM
	c. output variable		g. path through state trellis of a 3-state HMM for 5 time slices
	d. joint probability		h. coupling of 3 HMMs

Figure 1: Key for interpreting graphical notation used in this article.

Probabilistic inference graphs are a convenient way of representing the independence structure of a distribution over random variables [Smyth et al., 1997], which we will use throughout this article. Graphically, hidden Markov models are depicted "rolled out" in time with the state variable taking on a new value in each time slice (figure 1f). In tree-structured graphs such as (figure 1f), a node is independent of its ancestors given its sole parent. Thus, in an HMM, future states are independent of past states given the present state, and outputs are independent of each other given the states. Th assertion that the present state is all the context one needs for continued inference is the essence of the *Markov property.* The Markov property does not actually mandate a single state variable with a single parent, but all classical Markov models are formulated this way, because it is the traditional boundary between practicable and impracticable inference graphs.

HMMs are used for classification in two ways: Given a sequence of observations $O = \{o_1, o_2, o_3, \ldots, o_n\}$, we would like to know the most likely model among a collection of HMMs or the most likely sequence of states $\hat{S}$ within a model. Using Bayes' rule:

$$\hat{S} = \operatorname*{argmax}_S P[S|O] \quad = \quad \operatorname*{argmax}_S \frac{P[O|S]P[S]}{P(O)} \tag{1}$$

where the numerator is the prior probability of the state sequence:

$$\begin{aligned} P[S] &= p(s_1, s_2, s_3, \ldots, s_{T-1}, s_T) \\ &= P_{s_1} P_{s_2|s_1} P_{s_3|s_2,s_1} \ldots P_{s_T|s_{T-1},s_{T-2},\ldots,s_1} \\ &= P_{s_1} P_{s_2|s_1} P_{s_3|s_2} \ldots P_{s_T|s_{T-1}} \qquad \text{(Markov property)} \\ &= P_{s_1} \prod_{t=2}^{T} P_{s_t|s_{t-1}} \end{aligned} \tag{2}$$

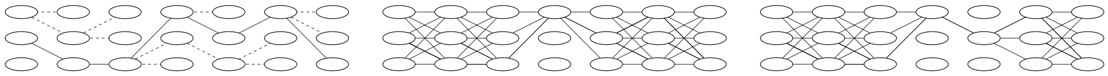


Figure 2: Searches through the state trellis of a 3-state HMM for (a) most likely path; (b) probability of a state ($= \alpha_{i,t}\beta_{i,t}$); (c) probability of a transition.

times the joint density of the observations given that sequence:

$$\begin{aligned} P[O|S] &= p(o_1, o_2, \ldots, o_t | s_1, s_2, \ldots, s_T) \\ &= p_{s_1}(o_1) p_{s_2}(o_2) \ldots p_{s_t}(o_t) \qquad \text{(Markov property)} \\ &= \prod_{t=1}^{T} p_{s_t}(o_t) \end{aligned} \tag{3}$$

The denominator $P(O)$ is usually unknown, presumed constant across sequences, and thus ignored. Thus we seek a state sequence that maximizes the numerator.

Substituting these forms into equation (1), we obtain a tractable statement of the state estimation problem:

$$\hat{S}|O = \operatorname*{argmax}_S P_{s_1} p_{s_1}(o_1) \prod_{t=2}^{T} p_{s_t}(o_t) P_{s_t|s_{t-1}} \tag{4}$$

$\hat{S}$ can be found by exploring a state trellis such as that shown in figure 2a. Each column corresponds to possible states for a time slice. A path across the trellis multiplies the probabilities associated with each traversed state and transition. The most likely path can be found via brute force search through all N^T possible paths, but dynamic programming provides a recursive $O(TN^2)$ procedure: Given the most likely incomplete path $S_{j,t-1}|O$ leading up to each state j at time $t-1$, the most likely path leading up to each state i at time t is

$$\hat{S}_{i,t}|O = \operatorname*{argmax}_j p_i(o_t) \cdot P_{i|j} \cdot \hat{S}_{j,t-1}|O \tag{5}$$

E.g., each state elects to continue one of the N best paths from the previous time slice. The most likely full state sequence is then $\hat{S}|O = \operatorname{argmax}_i \hat{S}_{i,T}|O$. This is the Viterbi algorithm [Forney, 1973]. If we replace "argmax" with "$\sum$" and consider the sum likelihood for all possible paths we obtain $P[M|O]$, the probability of the model given an observation . Instead of $\hat{S}_{i,t}|O$, summing obtains the "forward variable" $\alpha_{i,t}$, the joint density of observations up to time t and $s_t = i$. Similarly the "backward variable" $\beta_{i,t}$ is the joint density of observations after time t and $s_t = i$. Given these two values, we can estimate the probability of any transition given an observation (figure 2c):

$$\hat{P}_{s_t=i,s_{t-1}=j|O} \quad = \quad \frac{\alpha_{j,t-1} P_{i|j} p_{s_t=i}(o_t) \beta_{i,t}}{P(O)} \tag{6}$$

from which we can re-estimate the conditional transition probabilities:

$$\hat{P}_{i|j} \quad = \quad \frac{\sum_{t=2}^{T} \hat{P}_{s_t=i,s_{t-1}=j|O}}{\sum_{t=2}^{T} \alpha_{j,t-1} \beta_{j,t-1}} \tag{7}$$

This is Baum-Welch re-estimation [Baum, 1972]. Forward-backward analysis provides the expected state values given the observations and model parameters; re-estimation adjusts the parameters to maximize model likelihood given the state probabilities and the observation. Together, they form an expectation-maximization procedure that, when repeated until convergence, finds HMM parameters corresponding to a local maximum in model likelihood [Dempster et al., 1977]. The efficiency of this training procedure makes HMMs very attractive for applications time-series modeling and classification.

3. Multiple channels versus multiple processes

The Markov property holds that the current state is all the context one needs for modeling; given that, future states are independent of past states. In Markov models this has been realized in a representation of context that is limited to a single state variable. To accommodate multiple channels of data, HMMs are usually formulated with multivariate p.d.f's on the output variables. However, if there are multiple processes generating those channels, one must hope that these processes evolve in lockstep, since any variation between them is modeled as noise. If that variation carries information, e.g., the processes interact, HMMs may prove inappropriate models, because with context limited to a single variable, a distinct state must be reserved for each possible combination of signals on all the channels. This Cartesian product solution rapidly becomes untenable. Other variations on HMM structure aimed at multiple channel modeling include using neural networks for outputs [Baldi and Chauvin, 1996], input-output models [Bengio and Frasconi, 1996], and per-state mixture models. However, these are all based on the traditional Markov formulation of a single process dynamic.

Signals from systems of multiple processes are common and interesting modeling problems. Nearly any signal produced by human behavior can be usefully decomposed into a group of interacting processes: There are several interacting articulatory centers in speech production, and multiple levels of control at work in motor behaviors such as gesture and dextrous action. Modeling such a system as single lumped process may yield passable recognition rates, but there is a performance ceiling that can only be broken by considering the system in more detail, e.g., by teasing apart some of its process structure into multiple state variables. The naive solution mentioned above — modeling with a large HMM having the Cartesian product of all possible process states — is rarely satisfactory. The computational cost is prohibitive, a surfeit of parameters leads to overfitting, and there is often insufficient data for a large number of states, usually resulting in undersampling and numerical underflow errors. Retreating to smaller state spaces ameliorates the problem somewhat but introduces new problems of underfitting where states that should be distinct are fused. In either case, the interaction is being crudely tabulated but not explicity modeled.

Even with the correct number of states and vast amounts of data, large HMMs generally train poorly because the data is partitioned among states early (and incorrectly) during training; the Markov independence structure then ensures that the data is not shared by states, thus reinforcing any mistakes in the initial partitioning. Systems with multiple processes have states that share properties and thus emit similar signals; the Markovian framework works against this systematicity. Even though an HMM can model any system in principle, in practice, the simple independence structure is a liability for large systems and for systems with compositional state.

Models with compositional state representations would offer conceptual advantages of parsimony and clarity, with consequent computational benefits in efficiency and accuracy. Using graphical notation we can construct various architectures for multi-HMM couplings offering compositional state under various assumptions of independence. Although the Markov property is conserved, a basic assumption of Markov *models* is violated and we will require new inference and estimation procedures. Because these algorithms will follow directly from regularities of the inference graph, we will consider a few varieties of coupling before selecting one for algorithmic development.

4. Varieties of couplings

There are two basic kinds of couplings obtainable by extending the HMM framework. The weakest is when two independent processes are nominally coupled at the output, superimposing their outputs in a single signal (figure 3a). This is generally known as the source separation problem: Signals with zero mutual information are overlaid in a single channel, e.g., unrelated voices at a cocktail party. In true couplings, the processes are dependent and interact by influencing each other's states (figure 3b). This is generally known as the sensor fusion problem: Multiple channels carry complementary information about different components of a system, e.g., acoustical signals from speech and visual features from lip-tracking [Stork and Hennecke, 1996]. (Source separation can also be done with dependent processes; at risk of splitting hairs, this is probably more appropriate for interesting cocktail parties.)

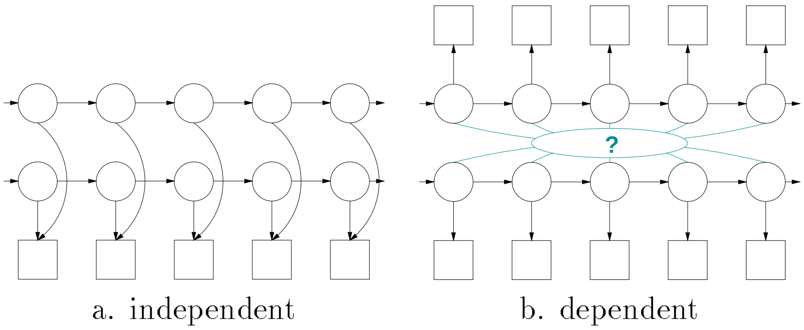
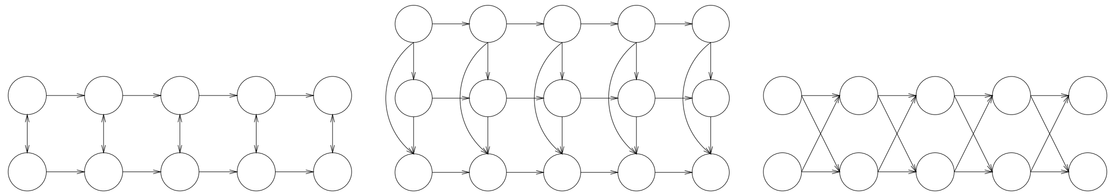


Figure 3: Inference graphs for couplings of two HMMs chains, rolled out to five time slices.

In the case of nominal couplings (figure 3a), it is possible to simultaneously factor the signal into component processes and model each process independently. These have come to be known as factorial hidden Markov models (FHMMs). They have a clear representational advantage over HMMs: to model C processes each with N states each would require an HMM with N^C joint states, typically intractable in both space and time. FHMMs are tractable in space, taking NC states, but present an inference problem equivalent to that of a combinatoric HMM, thus exact solutions are intractable in time. Tractable approximations are available using mean field theory from statistical physics [Williams and Hinton, 1990, Parisi, 1988]. Ghahramani and Jordan [Ghahramani and Jordan, 1996] have developed an elegant "structured mean field" approximation in which the inference graph is partitioned into subgraphs that can be tractably estimated via exact methods (e.g., forward-backward analysis of the independent HMMs). A free energy function is defined over the whole graph. "Severed" links are replaced by coupling terms in the energy function. The mean field approximation alternately minimizes the energy function, then re-estimates the subgraphs. In principle the mean field approximation can be used to train virtually any elaboration on HMM structure in $O(TCN^2)$ time, simply by severing the appropriate links. However, if there are strong and varied interactions across the severed links, the approximation will be quite poor.

Conventional HMMs excel for processes that evolve in lockstep; FHMMs are meant for processes that evolve independently. Many problems tend to lie between the two extremes. Two processes may interact without wholly determining each other, as do voices in polyphonic music, players in a tennis game, or arm motions in gesture. Each process has its own internal dynamic and is influenced by others, possibly causally. A process may be more or less influential depending on its state: E.g., in tennis, "playing net" substantially constrains the range and returns of the other player, while



a. LHMM b. HMDT c. CHMM

Figure 4: Varieties of couplings for dependent processes. Output nodes are omitted.

playing midcourt is less constraining. A variety of inference architectures have been proposed to model these phenomena (figure 4). These classes of graphs license special algorithms and are appropriate for certain kinds of problems, so we shall discuss them briefly.

Figure 4a shows a graph with joint probabilities between synchronous states, encoding which state pairs co-occur more or less frequently. We may call this inference graph a linked hidden Markov models (LHMMs); they are equivalent to Cartesian product HMMs with a bias probability on each joint state. Its independence structure is suitable for expressing symmetrical synchronous constraints such as the fact that it is rare to see tennis opponents playing net at the same time. [Saul and Jordan, 1995] present an $O(TN^3)$ exact algorithm for training an equivalent two-chain Boltzmann machine based on *decimation*, a method from statistical mechanics in which the marginal distributions of singly or doubly connected nodes are integrated out. A limited class of graphs can be recursively decimated, obtaining correlations for any connected pair of nodes. This includes the two-chain ladder graph shown in the figure.

Figure 4b shows a cascade of synchronous conditional probabilities down an ordered hierarchy of HMMs, meant to model the constraints imposed by "master" processes on "slave" processes. Hidden Markov decision trees [Jordan et al., 1996] use structured mean field approximations to break the graph into HMM *or* decision tree subgraphs. This independence structure is suitable for representing hierarchical structure in a signal, for example the baseline of a song constrains the melody, and both constrain the harmony.

In general, inference architectures with synchronous links assume lockstep processes that do not have causal (temporal) influences on each other. To capture interprocess influences across time, the coupling must bridge time slices, e.g., the crosswork of conditional probabilities in figure 4c. Arguably, this offers the strongest model of interprocess influences. Intuitively and empirically, it is appropriate for processes that influence each other asymmetrically and possibly causally, e.g., in tennis, going up to the net will drive the opponent back, weak serves will pull her forward, etc. We shall call this architecture and its generalizations coupled hidden Markov models[1] (CHMMs).

Inference in CHMMs has the same time complexity as the Cartesian product HMMs, thus exact algorithms are not attractive. E.g., graphical analysis of the independence structure of a two-chain CHMM reveals that exact maximum *a posterior* (MAP) inference is an $O(TN^4)$ computation [Jensen et al., 1990, Kjaerulff, 1992, Smyth et al., 1997]. In recent years researchers have adopted Monte Carlo sampling methods for estimation problems in a class of CHMM-like inference graphs called dynamic probabilistic networks [Dean and Kanazawa, 1989]. These algorithms improve over random sampling by discarding, weighting, and/or varying sample state sequences according to their posteriors ([Henrion, 1988, Fung and Chang, 1989, Kanazawa et al., 1995], respectively). The space and time complexities are also typically exponential in the number of simultaneous state variables [Binder et al., 1997b]. While some of these algorithms are *consistent* (they converge asymptotically to the true distribution), it is not known whether they are *efficient* (they produce the best estimates given the order of computation).

We turn now to developing a deterministic $O(TN^2)$ algorithm for maximum-entropy approximations to state and parameter values in CHMMs.

5. CHMM posteriors

The posterior of a state sequence through fully coupled C-chain HMM is

$$P(S^{\{C\}}|O) \quad = \quad \prod_{(c)}^{C} \left(\underbrace{P_{s_1^{(c)}}}_{\text{prior}} \underbrace{p_{s_1^{(c)}}(o_1^{(c)})}_{\text{1st output}} \prod_{t=2}^{T} \left(\underbrace{p_{s_t^{(c)}}(o_t^{(c)})}_{\text{outputs}} \prod_{(d)}^{C} \underbrace{P_{s_t^{(c)}|s_{t-1}^{(d)}}}_{\substack{\text{c=d: transition} \\ \text{else couplings}}} \right) \right) \Big/ P(O) \tag{8}$$

where superscripting indexes a chain, such that $p_{s_t^{(c)}}(o_t^{(c)})$ is the probability of the output given a state in chain c and $P_{s_t^{(c)}|s_{t-1}^{(d)}}$ is the probability of a state in chain c given a previous state in chain d. In most of this article we will concentrate on two-chain couplings, where the most dramatic gain in modeling power is to be expected. We write its posterior as

$$P(S|O) \quad = \quad \frac{P_{s_1} p_{s_1}(o_1) P_{s_1'} p_{s_1'}(o_1')}{P(O)} \prod_{t=2}^{T} P_{s_t|s_{t-1}} P_{s_t'|s_{t-1}'} P_{s_t'|s_{t-1}} P_{s_t|s_{t-1}'} p_{s_t}(o_t) p_{s_t'}(o_t') \tag{9}$$

1. "Coupled hidden Markov model" is a mild misnomer because we have jettisoned a key restriction associated with the Markov models. However, the Markov *property* is preserved and the name helps to stave off the a proliferation of unrelated names for related models.

where $s_t, s'_t; o_t, o'_t$ denote states and observations of the opposing HMMs. An important feature of this kind of distribution and its "shoelace" graph is that it licenses an efficient algorithm for constructing and sampling a reduced-complexity state trellis, which we develop in the next section.

6. N-heads dynamic programming

Dynamic programming is the engine at the heart of many expectation-maximization algorithms, including the classic HMM procedures for calculating model likelihoods (forward analysis) and estimating states (Viterbi analysis). Dynamic programming allows us to collect statistics on an exponential number of possible paths through an HMM state trellis in polynomial time, because evidence from all paths that share a state at time t may be combined without loss of information. Thus N^T paths can be explored by tracking only N path "heads." To collect statistics for estimation, it is necessary to pass through every state and every transition at every time slice, thus dynamic programming in a trellis of length T and width N takes $O(TN^2)$ time.

A coupled HMM of C chains has a joint state trellis that is in principle N^C states wide; the associated dynamic programming problem is $O(TN^{2C})$. Here we show that it is possible to relax the assumption that every transition must be visited, thereby obtaining an $O(T(CN)^2)$ algorithm that closely approximates the full combinatoric result.

We wish to formulate a policy for sampling a small subset of the of possible state sequences while obtaining as much information as possible. One might have the intuition that a random sampling of state sequences would give the most representative picture of the model's statistics, but this is in fact not desirable. The posterior probability mass of an HMM is not distributed evenly among all possible state sequences; it is by definition concentrated in state sequences that are close to the "true" or MAP state sequence. Low-probability sequences carry relatively little information for estimation problems. We now show this formally.

Given some training data X, the posterior of the model M is the sum of the posteriors of all possible paths $S^{\{\}}$ through the model; let us denote the expectation of this value in each time slice $\langle P(M|X)\rangle_{S\{\}}$. Let us assume now only some subset of paths $S^{\{Q\}}$ is explored; what policy for path selection should we formulate so that the least information is lost? We may cast this as a problem of greedily minimizing the cross entropy with the all-paths posterior:

$$
\begin{aligned}
D(S^{\{\}}\|S^{\{Q\}}) &= \sum_X \langle P(M|X)\rangle_{S\{\}} \log \frac{\langle P(M|X)\rangle_{S\{\}}}{\langle P(M|X)\rangle_{S\{Q\}}} && (10)\\
&= \sum_X \langle P(M|X)\rangle_{S\{\}} \log \langle P(M|X)\rangle_{S\{\}} && (11)\\
&\quad - \sum_X \langle P(M|X)\rangle_{S\{\}} \log \langle P(M|X)\rangle_{S\{Q\}} && (12)
\end{aligned}
$$

$$
\begin{aligned}
&= f(H(X)) - \sum_X \langle P(M|X) \rangle_{S\{\}} \log \langle P(M|X) \rangle_{S\{Q\}} && (13) \\
&\leq f(H(X)) - \sum_X \langle P(M|X) \rangle_{S\{\}} \langle \log P(M|X) \rangle_{S\{Q\}} && (14)
\end{aligned}
$$

In the last step we have used Jensen's inequality push the log into the expectation and thereby obtain an upper bound on the cross entropy. The first term is a monotonic function of the entropy and thus constant, so we concentrate on the second term. Carrying through the standard assumption that all training sequences X are equiprobable, we may also treat the first expectation in equation (14) as a constant and pull it out of the sum:

$$
D(S^{\{\}}||S^{\{Q\}}) \leq f(H(X)) - c \sum_X \langle \log P(M|X) \rangle_{S\{Q\}} \tag{15}
$$

Minimizing the upper bound is then strictly a matter of maximizing the expectation in the second term by choosing the set of paths $S^{\{Q\}}$ with the greatest probability mass. We shall expand this expectation with the posterior for a two-chain CHMM:

$$
\begin{aligned}
Q &= \operatorname*{argmax}^{N}_{s \in S^{\{\}}} \sum_X \langle \log P(M|X) \rangle_s && (16) \\
&= \operatorname*{argmax}^{N}_{s \in S^{\{\}}} \sum_X \left\langle \log P_{s_1} P_{s'_1} p_{s_1}(o_1) p_{s'_1}(o'_1) \prod_{t=2}^{T} P_{s_t|s_{t-1}} P_{s'_t|s'_{t-1}} P_{s_t|s'_{t-1}} P_{s'_t|s_{t-1}} p_{s_t}(o_t) p_{s'_t}(o'_t) \right\rangle_s \\
&= \operatorname*{argmax}^{N}_{s \in S^{\{\}}} \sum_X \left\langle \begin{pmatrix} \log P_{s_1} + & \log P_{s'_1} + \\ \log p_{s_1}(o_1) + & \log p_{s'_1}(o'_1) \end{pmatrix} + \sum_{t=2}^{T} \begin{pmatrix} \log P_{s_t|s_{t-1}} + \log P_{s'_t|s'_{t-1}} + \\ \log P_{s_t|s'_{t-1}} + \log P_{s'_t|s_{t-1}} + \\ \log p_{s_t}(o_t) + \log p_{s'_t}(o'_t) \end{pmatrix} \right\rangle_s && (17)
\end{aligned}
$$

A practical procedure that finds these $S^{\{Q\}}$ paths in $O(TN^2)$ time will have to satisfy the following requirements. **Time/space**: no more than $O(N)$ path heads can be tracked. **Visiting**: Every component state must be visited so that statistics may be collected for re-estimating transition and output probabilities.

To visualize such a policy, we will graphically represent the problem using two coupled state trellises for the component HMMs in a two-chain CHMM. Each state sequence through the trellis is double-tracked, having a head in one HMM $h_{i,t}$ and an associated "sidekick" $k_{i,t}$ in the opposite HMM (figure 5). We will call a sequence of {head,sidekick} pairs a "path," and the subsequence leading up to any particular {head,sidekick} pair the "antecedent path."

To satisfy the visiting criterion, we must guarantee that each component state at time t is a head of some path, e.g., $h_{i,t} = i$ (as in conventional trellis algorithms). Coupling two HMMs of N, M states then takes $N + M$ heads each with a sidekick; coupling three HMMs of N, M, L states takes $N + M + L$ heads each with two sidekicks, and so on. Since every component state must be a head, equation (17) can be

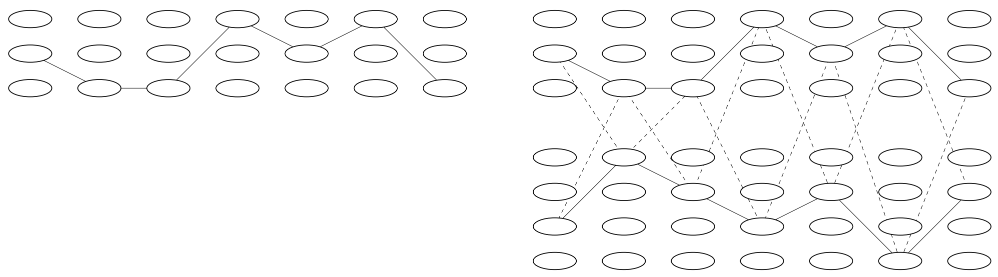


Figure 5: A single path through a standard 3-state conventional HMM trellis and through a reduced trellis for a CHMM of $3 \times 4 = 12$ joint states. Solid lines represent transition probabilities; dashed lines, coupling probabilities.

simplified: For the set of paths that pass through all heads in chain one, the left hand column in both parenthesized sums in equation (17) is constant ($\sum_i^N \log P_{s_t=i|s_{t-1}} = C$, etc.). Consequently, for these paths we only need to maximize the right hand columns, which amounts to choosing the MAP sidekick:

$$Q_1 = \underset{s \in S'\{\}}{\overset{N}{\operatorname{argmax}}} \sum_X \left\langle \begin{pmatrix} \log P_{s'_1} + \\ \log p_{s'_1}(o'_1) \end{pmatrix} + \sum_{t=2}^{T} \begin{pmatrix} \log P_{s'_t|s'_{t-1}} + \\ \log P_{s'_t|s_{t-1}} + \\ \log p_{s'_t}(o'_t) \end{pmatrix} \right\rangle_s \tag{18}$$

The converse applies for heads in the opposite chain; we choose a sidekick to maximize the left hand columns. Thus, even though the choices of heads, sidekicks, and antecedents are mutually contingent, of the $O(N^3)$ {head, sidekick, antecedent} tuples in a two-chain CHMM, we can find $O(N)$ MAP tuples in $O(N^2)$ time by first associating MAP sidekicks to each antecedent paths, then associating antecedent paths to each new head. This directly leads to the approximate forward- and Viterbi-analyses developed below.

6.1 Forward-backward analysis

In each step of the forward analysis we seek the MAP mass {head, sidekick} pairs given all antecedent paths. Every head in a chain sums over the same set of antecendent paths, and thus shares the same sidekick. This sidekick simply has the maximum marginal posterior given all antecedent paths. Fortunately, it can be found directly without marginalizing: (1) In each chain, choose the MAP state given all antecedent paths. This will be the sidekick to heads in other chains. (2) For each head, calculate a new path posterior given antecedent paths and sidekicks. To calculate the forward variable $\alpha_{i,t}$ associated with each head, we marginalize out the sidekicks.

The backward variable $\beta_{i,t}$ is similarly calculated by tracing back through the paths selected by the forward analysis. These procedures are detailed in appendix A.

6.2 Viterbi analysis

In each step of the Viterbi analysis we seek the MAP density {head, sidekick} pairs given all antecedent paths. In ordinary Viterbi algorithms, for each head at time t we choose an antecedent path in $t-1$; in the reduced trellis algorithm we must also choose a sidekick in t, and in contrast to the forward-backward analysis, each head can have a different sidekick. Though mutually contingent, these choices can be done in two steps: (1) for each antecedent path in $t-1$, select MAP sidekicks in t; (2) for each head in t, select the antecedent path and associated sidekick that maximizes the new head's posterior. If we have prior knowledge that intrachain dynamics dominate interchain dynamics, we may also use the heuristic method of [Jordan et al., 1996], in which we perform a conventional Viterbi analysis in one chain while holding state assignments constant in all others, and cycle through the chains until convergence to a local maximum.

6.3 Re-estimation

After collecting statistics using the N-heads method, transition matrices within chains are re-estimated according to the conventional HMM formulæ(equation (7)); the formula for coupling matrices between HMMs is similar:

$$P_{s'_t=i,s_{t-1}=j|O} = \frac{\alpha_{j,t-1} P_{i'|j} p_{s'_t=i}(o'_t) \beta_{i',t}}{P(O)} \tag{19}$$

$$\hat{P}_{i'|j} = \frac{\sum_{t=2}^{T} P_{s'_t=i,s_{t-1}=j|O}}{\sum_{t=2}^{T} \alpha_{j,t-1}\beta_{j,t-1}} \tag{20}$$

Alternately, the statistics obtained by the N-heads algorithm can be used by gradient descent methods [Baldi and Chauvin, 1994, Binder et al., 1997a] to yield an on-line training procedure.

6.4 Complexity of multiple couplings

These procedures generalize directly to graphs containing multiple HMMs (figure 6): first associate sidekicks to antecedent paths, then associate antecedent paths to new heads. With CN paths, the complexity of a fully coupled set of HMMs is $O(T(CN)^2)$, for C chains each having N states; $O(TC\lceil G\rceil N^2)$ for a system of varied couplings where $\lceil G\rceil$ is the largest coupling; and therefore $O(TCN^2)$ models where each chain has a constant number of couplings, e.g., rings (see figure 7).

Full couplings of large numbers of HMMs result in very densely connected graphs. Exact estimation in these graphs is NP-hard [Cooper, 1990]; for major classes of dense graphs approximation methods are NP-hard as well [Dagum and Luby, 1993]. The

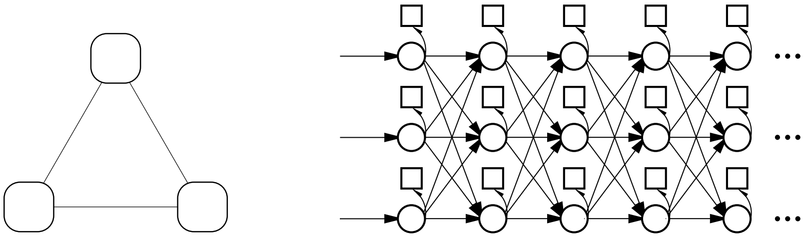


Figure 6: A coupling of three HMM chains, equivalent to the inference graph at right.

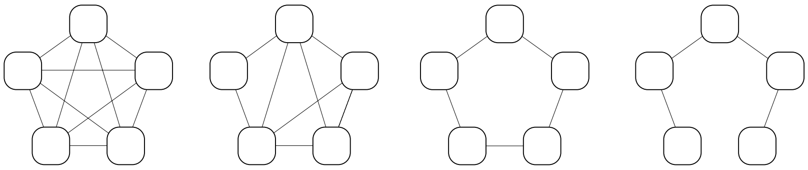


Figure 7: Full, partial, ring, and sheet couplings between five HMM chains.

N-heads method exploits a regularity in the structure of the inference graph to avoid these problems. The approximation will of course weaken as more and more HMMs are coupled, since an exponentially small fraction of the paths are being sampled. However, in practice it is rarely necessary to employ as many chains and couplings as one's intuitions about a system might suggest. Ten signal channels with a suspected $\binom{10}{2} = 45$ pairwise interactions does not necessarily require a fully coupled ten-chain model; the interactions between processes on several channels may be similar and thus amenable to modeling with just a few couplings, by allowing each chain to observe several channels. In section 8.3 we will illustrate an application in which eighteen channels from three processes are successfully modeled with only a single coupling. Nature appears to prefer to decompose into pairwise interactions and even the rare higher-order interaction can often be profitably be modeled by "currying" the effect into a set of pairwise interactions.

7. Conversion to and from Cartesian product HMMs

The transition probabilities of a coupled HMM can be converted to and from those of a Cartesian-product HMM. This is useful for comparison with other methods and formed the basis of earlier heuristic training procedures. We briefly sketch the mathematical rationale here for a two-chain coupling. The Cartesian product HMM transition matrix is the normalized product of two tensor products $(P_{S|S} \otimes P_{S'|S'}) \cdot R(P_{S'|S} \otimes P_{S|S'})$, where R is a row permute operator that swaps fast and slow indices. Element-wise, it is the product of probabilities:

$$
\begin{aligned}
P_{ss'} &= P_s P_{s'} && (21)\\
P_{ss'|ss'} &\propto P_{s|s} P_{s'|s'} P_{s|s'} P_{s'|s} && (22)
\end{aligned}
$$

where s is generically a state from a component HMM. The transition matrix must normalized be because it is not stochastic, in fact $\sum_i^{N_1} \sum_k^{N_2} P_{ik'|ss'} \ll 1$. To factor the matrix back into the two transition and two coupling matrices, we seek a probability-preserving projection from the $O(N^4)$ parameter space of the Cartesian product HMM to the $O(N^2)$ parameter space of the coupled HMMs. In appendix B we show that the projection that minimizes the cross entropy between the coupled and Cartesian posteriors is

$$
\begin{aligned}
P_i &= \sum_k P_{ik'} && (23)\\
P_{i|j} &\propto \sqrt{\sum_{k,l} P_{ik'|jl'}} && (24)
\end{aligned}
$$

Prior to N-heads dynamic programming, this formed the basis for an algorithm for training coupled HMMs in which a Cartesian product HMM was factored and recoupled between each re-estimation. Compared to the N-heads method, this algorithm has inferior complexity and accuracy, but it compares favorably to other algorithms and it has the interesting property that it yields the smallest train-test divergence of all evaluated algorithms in cross-validation trials.

An earlier and even less accurate factor-and-couple algorithm based on arithmetic factoring performed well enough enable the applications discussed in section 8.3. In this variant, we couple with the sum of the tensor products and thus can factor without the square root or normalization. The corresponding distribution is in fact not a properly formed independence network but corresponds to the sum of two proper posteriors, one the product of the two HMM posteriors, the other the product of their coupling matrices.

8. Experiments

We tested coupled HMMs in a battery of experiments with synthetic data to gauge its performance, then with real data from computer vision to verify its advantages over existing HMM algorithms.

8.1 Benchmarks with synthetic data

To test the coupled HMM and variations, Cartesian product HMMs were generated from random intra-HMM transition probabilities and coupling matrices that were structured so that state i in either HMM favored state $j \geq i$ in the other. The Cartesian product matrices were then perturbed with uniform noise, renormalized, and used to generate data via random walks. This ensured that a some of the dynamics arose from interactions between processes, but that the generating system was not conformant to the CHMM distributional form. 3-dimensional output means for two processes were drawn randomly from a Gaussian distribution ($\mu = 0, \sigma{=}1$). Covariances were set to $\mathbf{I}/4$. Forty sequences of thirty observations were taken, half for training and half for testing.

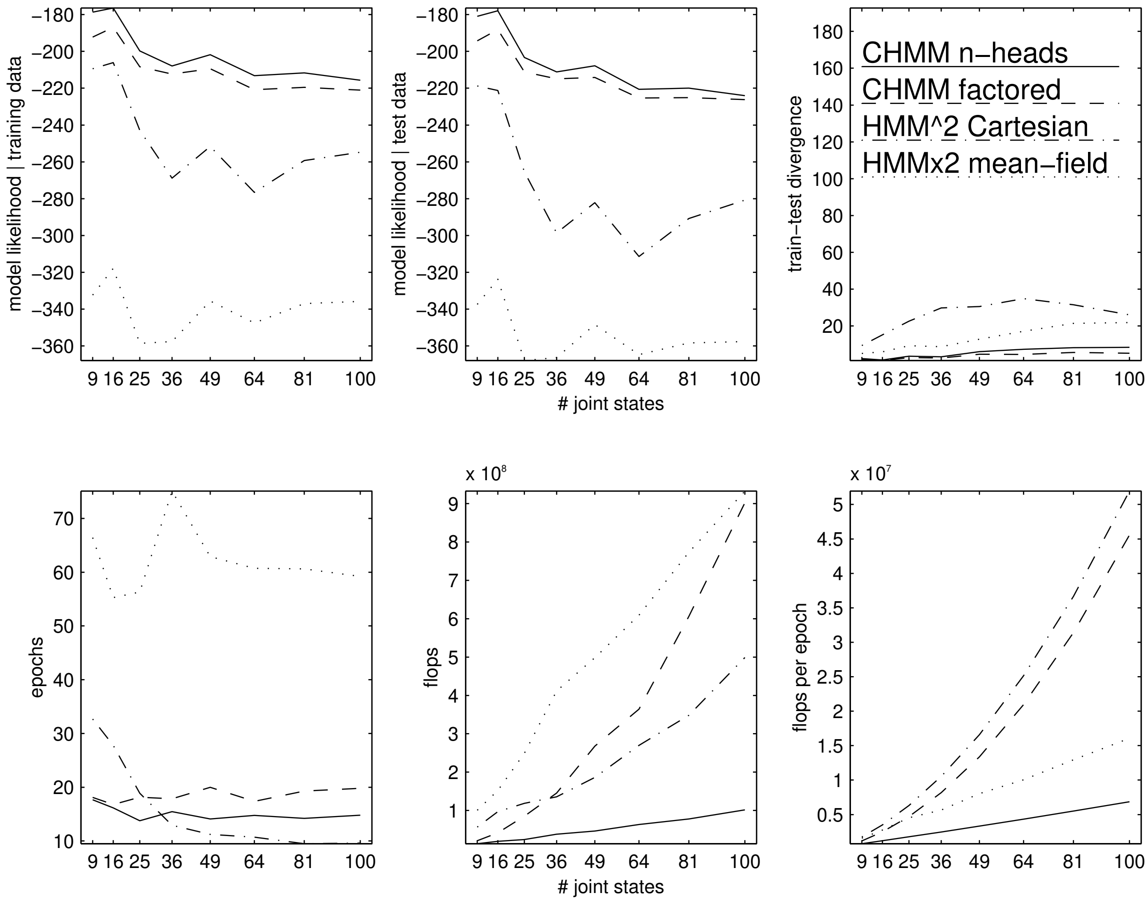
The data generators were then discarded, and a variety of higher-order HMM algorithms were used to model the data: (1) The $O(TN^2)$ N-heads CHMM; (2) an $O(TN^4)$ factor-and-couple CHMM; (3) an $O(TN^4)$ Cartesian product HMM; and (4) an $O(TN^2)$ factorial HMM with structured mean field approximation modified to include coupling terms in the energy function[2]. Algorithms 1&2 take two observation vectors per time slice while 3&4 take one; they received the concatenation of the two vectors and the appropriate terms in their initial means and covariances were set accordingly.

Each model was trained until the slope of the log likelihood curve fell below 0.01 or stopped when 100 epochs had passed. The result was then tested on the training sequences and cross-validated on the test sequences. The entire procedure was repeated 50 times for each of 3+3, 4+4 ... 10+10 (=100 joint) state coupled HMMs, for a total 400 trials training 1600 models on 32000 sequences.

8.2 Results

Figure 8 shows a graph of the log likelihoods of (a) the training data and (b) the test data. The disparity between the two, which indicates overfitting, is shown in (c). The N-heads CHMM yielded the most accurate and general models, followed closely by the factored CHMM. The Cartesian product HMM failed to model the interaction, but managed a remote third because it has a polynomial order of magnitude more free parameters with which to fit the data. The mean field FHMM, having neither abundant free parameters nor a strong prior for coupling, did poorly. From the graph of train-test divergence, it is clear that the Cartesian product HMM grossly overfitted

2. Mean field approximations of CHMM inference graphs are $O(TN^4)$; we followed the suggestion, "Rather than specifying these higher-order couplings through probability transition matrices, one can introduce second-order interaction terms in the energy function." [Ghahramani and Jordan, 1996] We independently compared the mean field algorithm with and without coupling terms in its energy function, and found that these terms reduce train-test divergence in cross-validation, increasing test data likelihoods but worsening training data likelihoods by a somewhat larger amount. This suggests that the coupling terms keep the component HMMs from overfitting the data.



a. training log likelihoods b. test log likelihoods c. train-test divergence
d. training epochs e. total FLOPs f. FLOPs per epoch

Figure 8: Comparison of CHMM, HMM, and FHMM algorithms.

the training data and generalized very poorly to the test data; the same system with the addition of factoring generalized best. The N-heads CHMM placed a close second, well ahead of the mean field.

The bottom row of graphs compares the time-performance of the algorithms, in (d) epochs, (e) total FLOPs, and (f) FLOPs per epoch. From (f) it is readily apparent that the N-heads CHMM and mean field FHMM are $O(TN^2)$ algorithms while the Cartesian and factored algorithms are $O(TN^4)$. What is surprising is that the N-heads and mean field algorithms are the best and worst, respectively, in terms of overall computation. This is largely because the the mean field took a very large number of iterations to converge (or terminate), while the N-heads converged fairly rapidly.

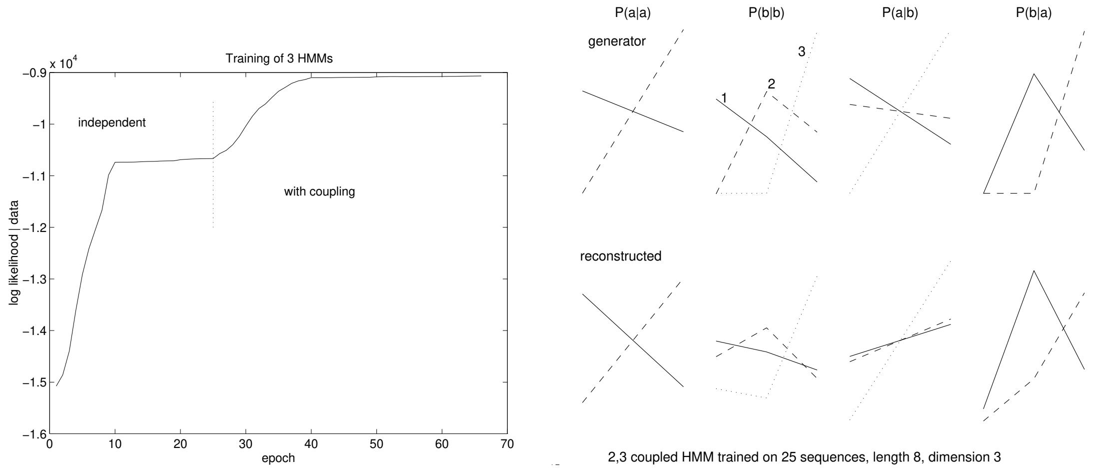
a 

Figure 9: (a) Log likelihood curve of three HMMs trained, coupled, and further trained.
(b) Transition probabilities reconstructed from data by training a CHMM with 2 states in one component HMM and 3 in the other. Each line depicts the probabilities for all transitions out of a given state. The qualitative structure of the transition matrices is largely recovered.

To demonstrate that additional information is being captured by the coupling, we trained three HMMs independently, then coupled them and continued training. Figure 9a shows the log likelihood of the model through twenty five epochs of independent training as conventional HMM, then for twenty more epochs trained as a CHMM. The gain from coupling will be even more concretely demonstrated below in applications with real data. Interestingly, we found that such two stage training does not improve in any way over training with couplings at the start. With small couplings, the data contains enough information for a reasonable reconstruction of the original transition probabilities, shown in figure 9b.

To test classification power, we compared several models in a task where they must distinguish between test sequences and the same sequences reversed, having the same stationary densities but opposite dynamics. As an estimate of the probability of classification error, we computed the log of the likelihood ratio of the *least* likely test sequence to the *most* likely reversal. Large log-ratios are desirable; a nonpositive log-ratio implies one or more misclassified sequences. In fifty train-and-test trials we compared the N-heads CHMM log-ratio with that of other algorithms, with results in table 1. The difference between the two CHMM algorithms was not statistically significant, while CHMMs advantages over conventional HMMs and factorial HMMs are both substantial and significant.

	Log discrimination ratios: N-head CHMM versus		
	factored CHMM	conventional HMM	factorial HMM
mean	11	52	124
p-value	> 0.4	0.001	$\ll 10^{-20}$

Table 1: Relative log discrimination ratio of CHMM and other models, estimated from the likelihood ratios of the least likely sequence and the most likely *reversed* sequence. Means (rounded) are a measure of how much more successfully CHMMs separated the two classes.

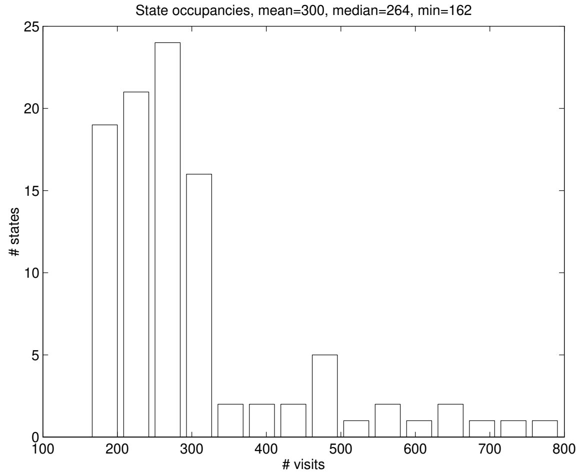


Figure 10: Histogram of state visits while generating training data for the 10x10 trials. Most joint states have approximately equal measure.

It may be objected that efforts should be made to optimize or at least reduce the number of states in the conventional HMMs, on the principle that many of the (joint) states may have negligible measure and should be ommitted from the representation. In fact, a histogram of the state visit counts from data generation (figure 10) shows that this is not the case: most of the probability mass is spread out over all states. Even the least visited state was occupied at least half as often as the mean. Any reduced-state HMM would underfit this data. Although no efforts were made to optimize any of the architectures in the simulations, we did seek optimal models in the real-data task below.

	Single HMMs	Linked HMMs	Coupled HMMs	N-head Coupled HMMs
accuracy	69.2%	36.5%	**94.2%**	**100%**
# parameters	25+30+180	27+18+54	36+18+54	36+18+54

Table 2: Accuracies and free parameter counts for classifying two-handed T'ai chi gestures. The parameter counts are transitions+means+covariances.

8.3 Tests with real-time computer vision data

T'ai chi ch'uan is a Chinese martial art and meditative exercise, consisting of stylized full-body and upper-body gestures that emphasize balance and fluidity. Although the limbs may in principle move independently, maintaining balance induces a coupling. Using a stereo vision blob-tracker, we obtained 3D hand-tracking data for both hands for three T'ai chi gestures with elaborate arm motions: the single whip, the cobra, and the brush knee. We collected 52 sequences of six channels at ≈10Hz, each lasting several hundred time steps. After hundreds of trials with varying state counts, we found that the data was best modeled by 3+3-state CHMMs, 3+3-state LHMMs, and 5-state HMMs. These architectures correspond to physical hypotheses: CHMM – there is spatial coupling in time (dynamic balance); LHMM – there is only spatial coupling (instantaneous balance); HMM – the data is best modeled just as an arbitrary space-time curve (balance not informative).

Five examples of each gesture were used to train each model. All models were then used to classify all 52 sequences of mixed novel and training data. The results are summarized in table 2, showing that the CHMM yielded the best classifier, an expected result since T'ai chi does in fact emphasize dynamic balance. The LHMM proved to be inappropriate; two LHMMs classified well but the third claimed most of the sequences. Although hundreds of different conventional HMMs were tried, none yielded accuracies over 70%. In fact, there was a considerable variation in classification power even given the same state count and training data.

The CHMM proved to be far less sensitive to initial conditions, e.g., it produced better and more consistent models regardless of the initial random parameter values. We ran fifty cross-validation trials with each algorithm, and collected statistics on the model posteriors given all the data. The results are plotted as fitted normal curves in figure 11, where the CHMM is distinguished by its sharp, high-likelihood peak—contrasting the broad spread of low-likelihood HMM results. Further details can be found in [Brand and Oliver, 1997].

In a second video interpretation task, CHMMs were trained to recognize continuous sequences of dextrous manipulation—picking up, putting down, pushing, batting, tool

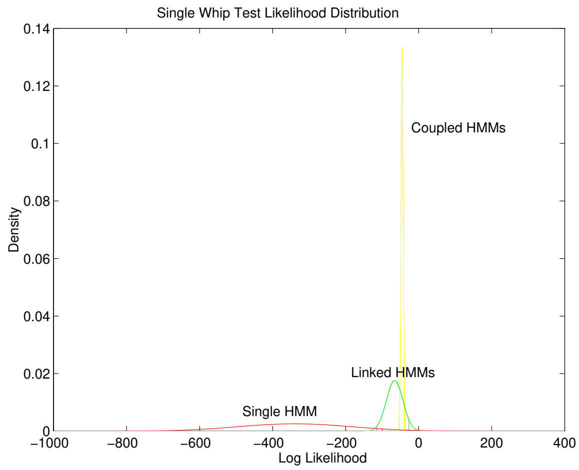


Figure 11: Distribution of models trained and cross-validated on gesture data.

use, etc. A vision front-end tracked three surfaces in the image plane, corresponding to the hand, the tool, and the affected object. Five ellipse parameters for each surface plus their three mutual spatial relations were collected as observation vectors for the CHMM. Half of vector was observed by each component chain. Since several interactions between these eighteen channels are distinctive for various actions, six of the channels were observed by both chains; through training the CHMM discovered which interactions between channels were useful to model. In this manner it was possible to layer the three spatial interations between the agent, patient, and instrument onto one coupling[3]. After the models were trained, they were chained into a finite-state machine in accordance with rules governing event sequences (e.g., nothing can be put down twice without being picked up in between). Novel action sequences were processed and the MAP state sequence through the composite CHMM was examined to obtain a sequence of recognized actions. Hit, miss, false alarms, precision, and recall statistics were compiled for four variants of this system systems, all shown in table 3. Further details are in [Brand, 1997b].

9. Variations and extensions

A number of interesting CHMM variations are possible within the $O(T(CN)^2)$ complexity of the N-heads algorithm. We briefly mention two extensions motivated by problems in acoustical signal processing, relating to the inference graphs in figure 12.

3. This representational trick works well for two- and three-body systems, but gives out at four, when the combinatorics begins to explode with six pairwise spatial relations and myriad tertiary interactions to consider. Notably, there are no four-participant action verbs in natural language.

model	hits	misses	false alarm	recall	precision
HMM	20	17	15	54%	57%
HMM+rules	28	9	9	75%	75%
CHMM	30	7	9	81%	76%
CHMM+rules	36	1	2	**97%**	**94%**

Table 3: Performance statistics for four recognizers tested on the continous action-labeling problem.

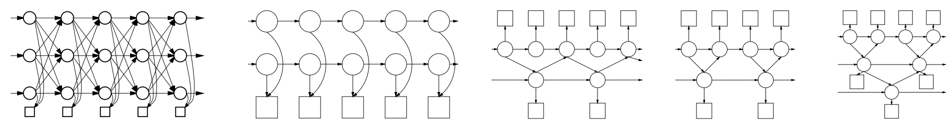


Figure 12: (a) CHMM inference graph for source separation.
(b) FHMM inference graph for source separation.
(c,d,e) CHMM inference graphs for signals at multiple time scales.

9.1 Source separation of superimposed signals

There are systems of interest in which signals from component processes are superimposed in a single or small number of channels (e.g., figure 12a). For example, the acoustical waveform of a string quartet. We have derived an extension to N-heads algorithm that performs source separation hand-in-hand with state estimation [Brand, 1997c]. Briefly, the posterior is expressed in terms of the choice of sidekicks, as before, and also as a choice of component signals, which must sum to the actual observed signal. Constrained maximization of this posterior yields a fixed-point equation for each component signal. The choice of sidekicks and calculation of component signals then alternates in an $O((CN)^2)$ expectation-maximization loop. In the degenerate case, where the chains are wholly uncoupled (figure 12b), the equations yield an $O(TCN^2)$ algorithm for source separation given C independent sources, e.g., unrelated conversations at a cocktail party.

9.2 Processes at different time scales

In many application areas, coupled processes may differ not only in complexity but in time scale as well. Such is the case in speech-reading, where the voicebox modulates the acoustic signal faster than lip articulations modulate the visual signal. Models for this problem have been proposed by [Saul and Jordan, 1996, Hihi and Bengio, 1996]. Two inference graphs for 2-1 time scales are shown in figure (12c,d). Such architectures can be evaluated using $O(T(DN)^2)$ variants of the N-heads algorithm [Brand, 1997a], where D is the number of differently conditioned hidden nodes in the inference graph. The procedure generalizes to multiple chains with doubling time scales (figure 12e).

10. Conclusion

Coupled hidden Markov models (CHMMs) are complex but regularly structured inference graphs that are well-suited for modeling dynamic interactions between processes and combining evidence from disparate sensors and modalities. These graphs have hidden Markov models as subgraphs and add coupling probabilities across chains and across time. This makes them a strong model for causal and potentially asymmetric interactions between processes. Since CHMMs model signals that are at variance with classic Markov formulations, new estimation procedures are needed. We present a general N-heads dynamic programming algorithm in which only $O((CN)^T)$ of the possible $O(N^{CT})$ state paths are explored, resulting in an $O(T(CN)^2)$ algorithm for inference over C fully coupled chains, or $O(TCN^2)$ if the number of couplings is constant. This contrasts with $O(TN^{2C})$ for naive (Cartesian) and exact (clustered) solutions. The N-heads method minimizes cross entropy with the full combinatoric posterior probability. It readily accommodates component HMMs with different couplings and/or state structures, and has been generalized to allow chains operating at different time scales and with superimposed outputs.

In experiments with a range of algorithms trained on real and synthetic data, we found that the N-heads method yielded the highest quality models in the least time, with high posteriors, robustness to initial conditions, good discrimination properties, and good generalization to novel examples. Because CHMMs are the appropriate distributional form for systems that have structure in both time and space, they have proven especially useful in the realm of video, where they have licensed novel applications in the interpretation of human gesture and action.

11. Acknowledgements

Mike Jordan was a source of illuminating conversations and papers. Zoubin Ghahramani provided software for FHMMs. Andy Wilson provided basic HMM software. Nuria Oliver acquired the T'ai chi data and ran the experiments summarized in table 2. Sandy Pentland and Dave Becker originally proposed the T'ai chi task as a domain for HMM methods. Many improvements in this presentation are due to comment from students, colleagues, and reviewers.

Appendix A. Forward-backward analysis

We denote heads and sidekick indices in each time slice t by $h_{i,t}, k_{i,t}$; $\alpha^*_{i,t}$ is the probability mass associated with each head; $q_{i,t}$ is the partial posterior probability (in the absence of sidekicks) of a state given all $\alpha^*_{j,t-1}$ and the output at t. The maximizing policy selects the sidekick that maximizes $\left\langle q_{k_{i,t},t} | \alpha^*_{j,t-1} \right\rangle_i$.

1. Calculate all partial posteriors:

$$q_{i,t} = p_i(o_t) \sum_j P_{i|h_{j,t-1}} P_{i|k_{j',t-1}} \alpha^*_{j,t-1} \tag{25}$$

2. Choose one best sidekick from each chain:

$$k_{i,t} = \operatorname*{argmax}_j q_{j',t} \tag{26}$$

3. The choice of heads is fixed at $h_{i,t} = i$.

4. Calculate full posteriors for each path:

$$\alpha^*_{i,t} = p_i(o_t) p_{k_{i',t}}(o_t) \sum_j P_{i|h_{j,t-1}} P_{i|k_{j',t-1}} P_{k_{i',t}|h_{j,t-1}} P_{k_{i',t}|k_{j,t-1}} \alpha^*_{j,t-1} \tag{27}$$

5. Marginalize out each head (over all possible sidekicks) to obtain forward variables in each chain:

$$\begin{aligned} \alpha_{i,t} &= p_i(o_t) \sum_j P_{i|h_{j,t-1}} P_{i|k_{j',t-1}} \sum_g p_{k_{g',t}}(o_t) P_{k_{g',t}|h_{j,t-1}} P_{k_{g',t}|k_{j',t-1}} \alpha^*_{j,t-1} \\ &= p_i(o_t) \sum_j P_{i|h_{j,t-1}} P_{i|k_{j',t-1}} \sum_g q_{g',t} \end{aligned} \tag{28}$$

Taking the maximum rather than the sum yields an N-heads Viterbi procedure in steps 1–4.

Note that in contrast to conventional forward-backward analysis, we have two different kinds of forward variables: α^* variables for propagating probabilities, and marginalized α variables for re-estimating parameters. The backward variables

$$\beta^*_{i,t} = \sum_j P_{h_{j,t+1}|i} P_{k_{j',t+1}|i} P_{h_{j,t+1}|k_{i',t}} P_{k_{j,t+1}|k_{i',t}} p_i(o_{t+1}) p_{k_{i',t+1}}(o_{t+1}) \beta^*_{j,t+1} \tag{29}$$

are computed using the sidekicks found in the forward analysis, and similarly marginalized.

Without increasing complexity, we can improve slightly over this greedy cross-entropy approximation by conditioning the choice of sidekicks in t on α^*_{t-1} *and* on sidekicks chosen in $t+1$. In practice, this causes the forward calculation to occasionally backtrack one time slice. Similarly, one may recalculate sidekicks in all t given α^*_{t-1} and β^*_{t+1}, then recalculate forward and backward variables. Although both schemes obtain slightly higher posteriors by expanding the temporal scale of the greedy method, neither has substantial impact on parameter estimation, suggesting that the basic algorithm is already near optimum.

A.1 Scaling

Since joint probabilities quickly become vanishingly small, a scaling procedure is used to prevent numerical underflow. Typically, a scaling variable $c_t = \sum_i \alpha^*_{i,t}$ is used to normalize the forward variables ($\alpha^*_{i,t} \leftarrow \alpha^*_{i,t}/c_t$) on each iteration. The backward variables are rescaled using the same values ($\beta^*_{i,t} \leftarrow \beta^*_{i,t} c_t$). Scaling must preserve the invariant that the posterior probabilities of all the states in a chain must sum to one in each time slice ($\sum_i \gamma_{i,t} = \sum_i \alpha_{i,t}\beta_{i,t} = 1$). In a conventional HMM, where dynamic programming exhaustively samples all possible state sequences, this invariant obtains automatically. In the N-heads algorithm only a small fraction of state sequences are sampled, so that $\sum_i \gamma_{i,t} < 1$. Noting that $\alpha_{i,t} = p_i(o_t)\sum_j \alpha_{j,t-1}P_{i|j}$, we obtain a simple procedure for rescaling the backward variables that implicitly restores the invariant:

$$\gamma_{i,t} \leftarrow \frac{\alpha_{i,t}\beta_{i,t}}{\sum_i \alpha_{i,t}\beta_{i,t}} \tag{30}$$

$$\beta_{i,t} \leftarrow \frac{\gamma_{i,t}}{\alpha_{i,t}} = \frac{\gamma_{i,t}}{p_i(o_t)\sum_j \alpha_{j,t-1}P_{i|j}} \tag{31}$$

Appendix B. Factoring Cartesian product transition matrices

The maximum entropy factoring of an $O(N^4)$ transition matrix into two $O(N^2)$ transition matrices and two $O(N^2)$ coupling matrices is that which minimizes the cross entropy between the distributional priors:

$$D(\prod_t P_{s_t s'_t|s_{t-1}s'_{t-1}}; \prod_t P_{s_t|s_{t-1}} P_{s'_t|s'_{t-1}} P_{s_t|s'_{t-1}} P_{s'_t|s_{t-1}}) \tag{32}$$

Taking the partial derivative with respect to a coupled transition probability,

$$\frac{\partial D}{\partial P_{s_t|s_{t-1}}} = \frac{\partial}{\partial P_{s_t|s_{t-1}}} \sum_x \left(\prod_t P_{s_t s'_t|s_{t-1}s'_{t-1}}\right) \log \frac{\prod_t P_{s_t s'_t|s_{t-1}s'_{t-1}}}{\prod_t P_{s_t|s_{t-1}} P_{s'_t|s'_{t-1}} P_{s_t|s'_{t-1}} P_{s'_t|s_{t-1}}} \tag{33}$$

We note here that since x is drawn from the distribution of signals that the transition probabilities are modeling, then $\sum_x \prod_t P_{s_t|s_{t-1}} \approx \sum_{s_t, s_{t-1}} P_{s_t, s_{t-1}}$.

$$\frac{\partial D}{\partial P_{s_t|s_{t-1}}} \approx \frac{\partial}{\partial P_{s_t|s_{t-1}}} \sum_{\begin{pmatrix} s_t, s_{t-1}, \\ s'_t, s'_{t-1} \end{pmatrix}} P_{s_t s'_t|s_{t-1} s'_{t-1}} \begin{pmatrix} \log P_{s_t s'_t|s_{t-1} s'_{t-1}} \\ -\log P_{s_t|s_{t-1}} - \log P_{s'_t|s'_{t-1}} \\ -\log P_{s_t|s'_{t-1}} - \log P_{s'_t|s_{t-1}} \end{pmatrix} \tag{34}$$

$$= \frac{\sum_{s'_t, s'_{t-1}} P_{s_t s'_t|s_{t-1} s'_{t-1}}}{P_{s_t|s_{t-1}}} \tag{35}$$

To obtain $P_{s_t|s_{t-1}}$, we add Lagrange multipliers guaranteeing $\sum_i P_{s_i|s_j} = 1$.

$$\frac{\partial}{\partial P_{s_t|s_{t-1}}} \lambda(\sum P_{s_t|s_{t-1}} - 1) = \lambda P_{s_t|s_{t-1}} \tag{36}$$

Combining with equation (35),

$$\frac{\sum_{s'_t, s'_{t-1}} P_{s_t s'_t|s_{t-1} s'_{t-1}}}{P_{s_t|s_{t-1}}} + \lambda P_{s_t|s_{t-1}} = 0 \tag{37}$$

$$\hat{P}_{s_t|s_{t-1}} = \sqrt{\frac{\sum_{s'_t, s'_{t-1}} P_{s_t s'_t|s_{t-1} s'_{t-1}}}{\lambda}} \tag{38}$$

Since λ renormalizes the probabilities, we finally write $\hat{P}_{s_t|s_{t-1}}$ as

$$\hat{P}_{s_t|s_{t-1}} = \frac{\sqrt{\sum_{s'_t, s'_{t-1}} P_{s_t s'_t|s_{t-1} s'_{t-1}}}}{\sum_{s \in S} P_{s|s_{t-1}}} \tag{39}$$

References

[Baldi and Chauvin, 1994] Baldi, P. and Chauvin, Y. (1994). Smooth on-line learning algorithms for hidden Markov models. *Neural Computation*, 6(2):305–316.

[Baldi and Chauvin, 1996] Baldi, P. and Chauvin, Y. (1996). Hybrid modeling, HMM/NN architectures, and protein applications. *Neural Computation*, 8(7):1541–1565.

[Baum, 1972] Baum, L. (1972). An inequality and associated maximization technique in statistical estimation of probabilistic functions of Markov processes. *Inequalities*, 3:1–8.

[Bengio and Frasconi, 1996] Bengio, Y. and Frasconi, P. (1996). Input/output hmms for sequence processing. *IEEE Transactions on Neural Networks*, 7(5):1231–1249.

[Binder et al., 1997a] Binder, J., Koller, D., Russell, S., and Kanazawa, K. (1997a). Adaptive probabilistic networks with hidden variables. *Machine Learning*. in press.

[Binder et al., 1997b] Binder, J., Murphy, K., and Russell, S. (1997b). Space-efficient inference in dynamic probabilistic networks. In *Proceedings, International Joint Conference on Artificial Intelligence.*

[Brand, 1997a] Brand, M. (1997a). Coupled hidden Markov models at multiple time scales. Vision and Modeling TR 428, MIT Media Lab.

[Brand, 1997b] Brand, M. (1997b). The "Inverse Hollywood Problem": From video to scripts and storyboards via causal analysis. In *Proceedings, Conference on Artificial Intelligence, AAAI.* Also MIT Media Lab Vision and Modeling TR 410, 12/96.

[Brand, 1997c] Brand, M. (1997c). Source separation with coupled hidden Markov models. Vision and Modeling TR 427, MIT Media Lab.

[Brand and Oliver, 1997] Brand, M. and Oliver, N. (1997). Coupled hidden markov models for complex action recognition. In *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition*, pages 994–999. Also MIT Media Lab Vision and Modeling TR 407, 11/96.

[Cooper, 1990] Cooper, G. F. (1990). The computational complexity of probablistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393–405.

[Dagum and Luby, 1993] Dagum, P. and Luby, M. (1993). Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 60:141–153.

[Dean and Kanazawa, 1989] Dean, T. and Kanazawa, K. (1989). Probabilistic temporal reasoning. *Computational Intelligence*, 5(3):142–150.

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

[Forney, 1973] Forney, G. (1973). The Viterbi algorithm. *Proceedings of the IEEE*, 6:268–278.

[Fung and Chang, 1989] Fung, R. and Chang, K. C. (1989). Weighting and integrating evidence for stochastic simulation in Bayesian networks. In *Proceedings, Conference on Uncertainty in Artificial Intelligence*, volume 5, Ontario, Canada. Morgan Kaufmann.

[Ghahramani and Jordan, 1996] Ghahramani, Z. and Jordan, M. I. (1996). Factorial hidden Markov models. In [Touretzky et al., 1996].

[Henrion, 1988] Henrion, M. (1988). Propagation of uncertainty in Bayesian networks by probabilistic logic sampling. In Lemmer, J. F. and Kanal, L. N., editors, *Uncertainty in Artificial Intelligence*, volume 2. Elsevier/North-Holland, Amsterdam.

[Hihi and Bengio, 1996] Hihi, S. E. and Bengio, Y. (1996). Hierarchical recurrent networks for long-term dependencies. In [Touretzky et al., 1996].

[Jensen et al., 1990] Jensen, F. V., Lauritzen, S. L., and Olesen, K. G. (1990). Bayesian updating in recursive graphical models by local computations. *Computational Statistical Quarterly*, 4:269–282.

[Jordan et al., 1996] Jordan, M. I., Ghahramani, Z., and Saul, L. K. (1996). Hidden Markov decision trees. In [Touretzky et al., 1996].

[Kanazawa et al., 1995] Kanazawa, K., Koller, D., and Russell, S. (1995). Stochastic simulation algorithms for dynamic probabilistic networks. In *Proceedings, Conference on Uncertainty in Artificial Intelligence*, volume 11, Montreal, Canada. Morgan Kaufmann.

[Kjaerulff, 1992] Kjaerulff, U. (1992). A computational scheme for reasoning in dynamic probabilistic networks. In *Proceedings, Conference on Uncertainty in Artificial Intelligence*, volume 8, pages 121–129. Morgan Kaufmann.

[Parisi, 1988] Parisi, G. (1988). *Statistical Field Theory.* Addison-Wesley, Redwood City, CA.

[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.

[Saul and Jordan, 1995] Saul, L. K. and Jordan, M. I. (1995). Boltzmann chains and hidden Markov models. In [Tesauro et al., 1995].

[Saul and Jordan, 1996] Saul, L. K. and Jordan, M. I. (1996). Exploiting tractable substructures in intractable networks. In [Touretzky et al., 1996].

[Smyth et al., 1997] Smyth, P., Heckerman, D., and Jordan, M. (1997). Probabilistic independence networks for hidden Markov probability models. *Neural Computation*, 7(2):227–269.

[Stork and Hennecke, 1996] Stork, D. G. and Hennecke, M. E., editors (1996). *Speachreading by humans and machines*, volume 150 of *NATO ASI Series, F: Computer and Systems Sciences.* Springer Verlag, Berlin.

[Tesauro et al., 1995] Tesauro, G., Touretzky, D. S., and Leen, T., editors (1995). *Advances in Neural Information Processing Systems*, volume 7, Cambridge, MA. MIT Press.

[Touretzky et al., 1996] Touretzky, D. S., Mozer, M. C., and Hasselmo, M., editors (1996). *Advances in Neural Information Processing Systems*, volume 8, Cambridge, MA. MIT Press.

[Williams and Hinton, 1990] Williams, C. and Hinton, G. E. (1990). Mean field networks that learn to discriminate temporally distorted strings. In *Proceedings, Connectionist models summer school*, pages 18–22, San Mateo, CA. Morgan Kaufmann.