Ghahramani, Z. (2001) An Introduction to Hidden Markov Models and Bayesian Networks. *International Journal of Pattern Recognition and Artificial Intelligence.* **15**(1):9-42.

An Introduction to Hidden Markov Models and Bayesian Networks

Zoubin Ghahramani
Gatsby Computational Neuroscience Unit
University College London
London WC1N 3AR, England
http://www.gatsby.ucl.ac.uk/~zoubin/
zoubin@gatsby.ucl.ac.uk

Abstract

We provide a tutorial on learning and inference in hidden Markov models in the context of the recent literature on Bayesian networks. This perspective makes it possible to consider novel generalizations of hidden Markov models with multiple hidden state variables, multiscale representations, and mixed discrete and continuous variables. Although exact inference in these generalizations is usually intractable, one can use approximate inference algorithms such as Markov chain sampling and variational methods. We conclude this review with a discussion of Bayesian methods for model selection in generalized HMMs.

Keywords: Dynamic Bayesian networks; hidden Markov models; state space models; EM algorithms; variational methods; Gibbs sampling.

1 Introduction

Hidden Markov models (HMMs) are a ubiquitous tool for modelling time series data. They are used in almost all current speech recognition systems, in numerous applications in computational molecular biology, in data compression, and in other areas of artificial intelligence and pattern recognition. Recently HMMs have also been used in computer vision applications—the topic of this special issue—such as image sequence modelling and object tracking. The goal of this paper is to answer the following questions about HMMs, independently of what application the HMM is used for:

- What are hidden Markov models? HMMs will be defined in section 2.
- How do they relate to other Markov models and Bayesian networks in general? Simply stated, hidden Markov models are a particular kind of Bayesian network. In section 3 we will provide a short tutorial on Bayesian networks and describe how HMMs and other Markov models relate to them.
- What are the algorithms for inference and learning in HMMs and Bayesian networks? In order to use an HMM to track an object, segment speech, or group amino-acid sequences into protein families, we need solutions to the inference and learning problems. In section 4 we will describe inference and learning algorithms for HMMs and how they relate to the general belief propagation and EM algorithms for Bayesian networks.
- What are the limitations of HMMs and how can these be overcome? While a lot of mileage has been obtained out of HMMs, these models are quite limited. In section 5 we discuss these limitations, and some generalizations of HMMs that overcome these limitations. Unfortunately, more complex models also require more complex (and sometimes approximate) algorithms for inference and learning. These will be described in section 6.
- How does one avoid overfitting and select model structures in HMMs? One of the last frontiers in the study of HMMs is how to select model structures and how to fit a complex model to a small data set

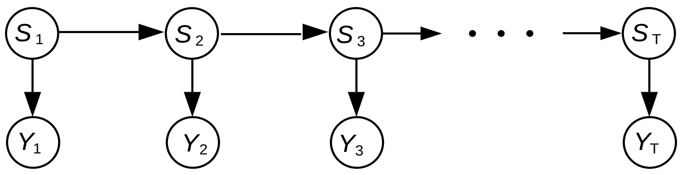

Figure 1: A Bayesian network specifying conditional independence relations for a hidden Markov model.

without fitting noise in the data. A Bayesian solution to these two (closely related) problems is given in section 7.

While there have been several tutorials and review articles written about HMMs (e.g. Rabiner and Juang, 1986), our understanding of HMMs has changed considerably since the realisation that they are a kind of Bayesian network [54]. Namely, we can now relate them to more complex and interesting models, and we can discuss general solutions to the problems of approximate inference, parameter learning, and model selection. The hope is that this article will introduce the reader to a state-of-the-art understanding of HMMs.

2 What are Hidden Markov models?

A hidden Markov model is a tool for representing probability distributions over sequences of observations. Let us denote the observation at time t by the variable Y_t. This can be a symbol from a discrete alphabet, a real-valued variable, an integer, or any other object, as long as we can define a probability distribution over it. We assume that the observations are sampled at discrete, equally-spaced time intervals, so t can be an integer-valued time index.

The hidden Markov model gets its name from two defining properties. First, it assumes that the observation at time t was generated by some process whose state S_t is *hidden* from the observer. Second, it assumes that the state of this hidden process satisfies the *Markov property*: that is, given the value of S_{t-1}, the current state S_t is independent of all the states prior to $t-1$.[1] In other words, the state at some time encapsulates all we need to know about the history of the process in order to predict the future of the process. The outputs also satisfy a Markov property with respect to the states: given S_t, Y_t is independent of the states and observations at all other time indices.

Taken together, these Markov properties mean that the joint distribution of a sequence of states and observations can be factored in the following way:

$$P(S_{1:T}, Y_{1:T}) = P(S_1)P(Y_1|S_1)\prod_{t=2}^{T} P(S_t|S_{t-1})P(Y_t|S_t), \tag{1}$$

where we have used the notation $X_{1:T}$ to mean $X_1, \ldots, X_T$. This factorization of the joint probability can be drawn graphically in the form shown in Figure 1. This graph, known as a Bayesian network, belief network, probabilistic graphical model, or probabilistic independence network, shows the dependencies between the variables in the model. We will define Bayesian networks more fully in the following section, but for now it is sufficient to note that each variable is represented by a node in the graph, each node receives directed arcs from nodes which it is conditionally dependent on in the factorization of the joint distribution.

A third assumption of the hidden Markov model is that the hidden state variable is *discrete*: S_t can take on K values which we will denote by the integers $\{1, \ldots, K\}$.

To define a probability distribution over sequences of observations, all that is left to specify is a probability distribution over the initial state $P(S_1)$, the $K \times K$ state transition matrix defining $P(S_t|S_{t-1})$ and the output model defining $P(Y_t|S_t)$. HMMs usually assume that the state transition matrices and output models are not dependent on t—in other words the model is *time invariant* (except for the initial state). If the observables are discrete symbols taking on one of L values, the output model can be fully specified by a $K \times L$ observation (or emission) matrix.

[1]This is a *first-order* Markov property. An n-th order Markov process is one in which S_t given $S_{t-1} \ldots S_{t-n}$ is independent of S_τ for $\tau < t-n$.

For real-valued observation vectors, $P(Y_t|S_t)$ can be modeled in many different forms, such as a Gaussian, mixture of Gaussians, or a neural network. For high-dimensional real-valued observations, a very useful output model is obtained by replacing the Gaussian by a factor analyser [50]. Factor analysis (FA) is a method for modeling correlations in high dimensional data, and is closely related to principal components analysis (PCA). The relationships between FA, PCA, mixture models, HMMs, and other models are reviewed in [48].

HMMs can be augmented to allow for input variables, U_t, in such a way that there is an input dependent state transition probability, $P(S_t|S_{t-1}, U_t)$ [9, 5, 40]. The system then models the conditional distribution of a sequence of output observations given a sequence of input observations. HMMs have been applied extensively to problems in speech recognition [32], computational biology [36, 3], and fault detection [53].

We now turn to Bayesian networks, a more general framework than hidden Markov models which will allow us both to understand the algorithms for inference and learning in HMMs and to formulate natural extensions to the HMM.

3 A Bayesian network tutorial

A Bayesian network is a graphical model for representing conditional independencies between a set of random variables. Consider four random variables, W, X, Y, and Z. From basic probability theory we know that we can factor the joint probability as a product of conditional probabilities:

$$P(W, X, Y, Z) = P(W)P(X|W)P(Y|W, X)P(Z|W, X, Y).$$

This factorization does not tell us anything useful about the joint probability distribution: each variable can potentially depend on every other variable. However, consider the following factorization:

$$P(W, X, Y, Z) = P(W)P(X)P(Y|W)P(Z|X, Y). \tag{2}$$

The above factorization implies a set of conditional independence relations. A variable (or set of variables) A is *conditionally independent* from B given C if $P(A, B|C) = P(A|C)P(B|C)$ for all A,B and C such that $P(C) \neq 0$. From the above factorization we can show that given the values of X and Y, W and Z are independent:

$$\begin{aligned} P(W, Z|X, Y) &= \frac{P(W, X, Y, Z)}{P(X, Y)} \\ &= \frac{P(W)P(X)P(Y|W)P(Z|X, Y)}{\int P(W)P(X)P(Y|W)P(Z|X, Y)\, dW\, dZ} \\ &= \frac{P(W)P(Y|W)P(Z|X, Y)}{P(Y)} \\ &= P(W|Y)P(Z|X, Y). \end{aligned}$$

A Bayesian network is a graphical way to represent a particular factorization of a joint distribution. Each variable is represented by a node in the network. A directed arc is drawn from node A to node B if B is conditioned on A in the factorization of the joint distribution. For example, to represent the factorization (2) we would draw an arc from W to Y but not from W to Z. The Bayesian network representing the factorization (2) is shown in Figure 2.

Some basic definitions from graph theory will be necessary at this point. The node A is a *parent* of another node B if there is a directed arc from A to B; if so, B is a *child* of A. The *descendents* of a node are its children, children's children, and so on. A *directed path* from A to B is a sequence of nodes starting from A and ending in B such that each node in the sequence is a parent of the following node in the sequence. An *undirected path* from A to B is a sequence of nodes starting from A and ending in B such that each node in the sequence is a parent *or child* of the following node.

The semantics of a Bayesian network are simple: each node is conditionally independent from its non-descendents given its parents.[2] More generally, two disjoint sets of nodes A and B are conditionally inde-

[2]Since there is a one-to-one correspondence between nodes and variables, we will often talk about conditional independence relations between nodes meaning conditional independence relations between the variables associated with the nodes.

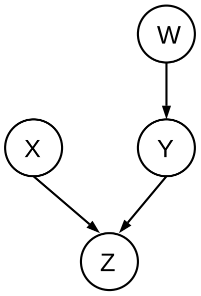

Figure 2: A directed acyclic graph (DAG) consistent with the conditional independence relations in $P(W, X, Y, Z)$.

pendent given C, if C *d-separates* A and B, that is, if along every undirected path between a node in A and a node in B there is a node D such that: (1) D has converging arrows[3] and neither D nor its descendents are in C, or (2) D does not have converging arrow and D is in C [44]. From visual inspection of the graphical model it is therefore easy to infer many independence relations without explicitly grinding through Bayes rule. For example, W is conditionally independent from X given the set $C = \{Y, Z\}$, since $Y \in C$ is along the only path between W and X, and Y does not have converging arrows. However, we cannot infer from the graph that W is conditionally independent from X given Z.

Notice that since each factorization implies a strict ordering of the variables, the connections obtained in this manner define a directed *acyclic* graph[4]. Furthermore, there are many ways to factorise a joint distribution, and consequently there are many Bayesian networks consistent with a particular joint. A Bayesian network G is said to be an independency map *I-map* for a distribution P if every d-separation displayed in G corresponds to a valid conditional independence relation in P. G is a *minimal I-map* if no arc can be deleted from G without removing the *I-map* property.

The absence of arcs in a Bayesian network implies conditional independence relations which can be exploited to obtain efficient algorithms for computing marginal and conditional probabilities. For *singly connected* networks, in which the underlying undirected graph has no loops, there exists a general algorithm called *belief propagation* [35, 44]. For *multiply connected* networks, in which there can be more than one undirected path between any two nodes, there exists a more general algorithm known as the *junction tree algorithm* [37, 29]. I will provide the essence of the belief propagation algorithm (since the exact inference methods used throughout this paper are based on it) and refer the reader to relevant texts [44, 28, 24] for details.

Assume we observe some *evidence*: the value of some variables in the network. The goal of belief propagation is to update the marginal probabilities of all the variables in the network to incorporate this new evidence. This is achieved by local message passing: each node, n sends a message to its parents and to its children. Since the graph is singly connected, n separates the graph, and therefore the evidence, into two mutually exclusive sets: $e^+(n)$, consisting of the parents of n, the nodes connected to n through its parents[5], and n itself, and $e^-(n)$ consisting of the children of n and the nodes connected to n through its children (Figure 3). The message from n to each of its children is the probability of each setting of n given the evidence observed in the set $e^+(n)$. If n is a K-valued discrete variable, then this message is a K-dimensional vector. For real-valued variables, the message is a probability density over the domain of values n can take. The message from n to each of its parents is the probability, given every setting of the parent, of the evidence observed in the set $e^-(n) \cup \{n\}$. The marginal probability of a node is proportional to the product of the messages obtained from its parents, weighted by the conditional probability of the node given its parents, and the message obtained from its children. If the parents of n are $\{p_1, \ldots, p_k\}$ and

[3]That is, D is a child of both the previous and following nodes in the path.

[4]Undirected graphical models (Markov networks) are another important tool for representing probability distributions, and have a different set of semantics [6, 17]. We will deal exclusively with directed graphical models in this paper.

[5]That is, the nodes for which the undirected path to n goes through a parent of n.

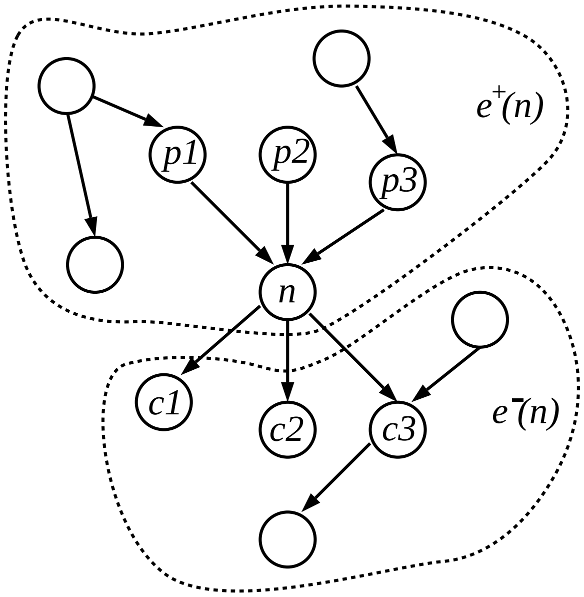

Figure 3: Separation of evidence in singly connected graphs.

the children of n are $\{c_1, \ldots, c_\ell\}$, then

$$P(n|e) \propto \left[\sum_{\{p_1,\ldots,p_k\}} P(n|p_1,\ldots,p_k) \prod_{i=1}^{k} P(p_i|e^+(p_i)) \right] \prod_{j=1}^{\ell} P(c_j, e^-(c_j)|n) \tag{3}$$

where the summation (or more generally the integral) extends over all settings of $\{p_1, \ldots, p_k\}$. For example, for the Bayesian network in Figure 2, given the evidence $e = \{X = x, Z = z\}$,

$$\begin{aligned} P(Y|X=x, Z=z) &\propto \left[\int P(Y|W)P(W)\, dW \right] P(Z=z, X=x|Y) \quad (4) \\ &\propto P(Y)\; P(Z=z|X=x,Y)\; P(X=x) \quad (5) \end{aligned}$$

where $P(W)$ is the message passed from W to Y since $e^+(W) = \emptyset$, and $P(Z = z, X = x|Y)$ is the message passed from Z to Y. Variables in the evidence set are referred to as *observable* variables, while those not in the evidence set are referred to as *hidden* variables.

3.1 Dynamic Bayesian networks

Hidden Markov models fall in a subclass of Bayesian networks known as dynamic Bayesian networks, which are simply Bayesian networks for modeling time series data. In time series modeling, the assumption that an event can cause another event in the future, but not vice-versa, simplifies the design of the Bayesian network: directed arcs should flow forward in time. Assigning a time index t to each variable, one of the simplest causal models for a sequence of data $\{Y_1, \ldots, Y_T\}$ is the first-order Markov model we have already mentioned, in which each variable is directly influenced only by the previous variable (Figure 4)::

$$P(Y_{1:T}) = P(Y_1)P(Y_2|Y_1) \cdots P(Y_T|Y_{T-1})$$

Having observed $\{Y_1, \ldots, Y_t\}$, the model will only make use of Y_t to predict the value of Y_{t+1}. One simple way of extending Markov models is to allow higher order interactions between variables, for example, an n^{th}-order Markov model allows arcs from $\{Y_{t-n}, \ldots, Y_{t-1}\}$ to Y_t. Another way to extend Markov models is to posit that the observations are dependent on a hidden variable, which we will call the *state*, and that the sequence of *states* is a Markov process (Figure 1). Hidden Markov models fall into this class of dynamic Bayesian network. Another very well-known model in this class is the linear-Gaussian state-space model, also known as the Kalman filter, which can be thought of as the continuous-state version of HMMs.

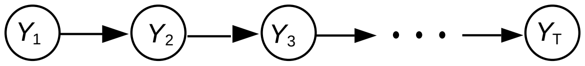

Figure 4: A Bayesian network representing a first-order Markov process.

3.2 State-space models

In state-space models, a sequence of D-dimensional real-valued observation vectors $\{Y_1, \dots, Y_T\}$, is modeled by assuming that at each time step Y_t was generated from a K-dimensional real-valued hidden state variable X_t, and that the sequence of X's define a first-order Markov process:

$$P(X_{1:T}, Y_{1:T}) = P(X_1)P(Y_1|X_1)\prod_{t=2}^{T} P(X_t|X_{t-1})P(Y_t|X_t). \tag{6}$$

This factorization of the joint probability means that the Bayesian network for state-space models is identical to that of HMMs (Figure 1) except that the hidden S variables are replaced by hidden X variables.

The state transition probability $P(X_t|X_{t-1})$ can be decomposed into deterministic and stochastic components:

$$X_t = f_t(X_{t-1}) + w_t$$

where f_t is the deterministic transition function determining the mean of X_t given X_{t-1}, and w_t is a zero-mean random noise vector. Similarly, the observation probability $P(Y_t|X_t)$ can be decomposed as

$$Y_t = g_t(X_t) + v_t.$$

If both the transition and output functions are linear and time-invariant and the distribution of the state and observation noise variables is Gaussian, the model becomes a linear-Gaussian state-space model:

$$X_t = AX_{t-1} + w_t \tag{7}$$
$$Y_t = CX_t + v_t \tag{8}$$

where A is the state transition matrix and C is the observation matrix.

Often, the observations can be divided into a set of input (or predictor) variables and output (or response) variables. Again, assuming linearity and Gaussian noise we can write the state transition function as

$$X_t = AX_{t-1} + BU_t + w_t, \tag{9}$$

where U_t is the input observation vector and B is the input matrix. The Bayesian network corresponding to this model would include a sequence of nodes $U_{1:T}$ each of which is a parent of the corresponding X_t. Linear-Gaussian state-space models are used extensively in all areas of control and signal processing.

What makes hidden Markov models and state-space models special is that their hidden state spaces are closed under their respective state transition probabilities and output models. In HMMs, the hidden state is assumed to have a K-valued discrete (a.k.a. multinomial) distribution. Under a $K \times K$ transition matrix one obtains another K-valued multinomial distribution. In SSMs the hidden state is assumed to be Gaussian distributed. After dynamics consisting of a linear transformation with Gaussian noise added, one again obtains a Gaussian distributed hidden state. This closed property of HMMs and SSMs makes inference and learning particularly simple and appealing in these models.

4 Learning and Inference

A Bayesian approach to learning starts with some *a priori* knowledge about the model structure—the set of arcs in the Bayesian network—and model parameters. This initial knowledge is represented in the form of a prior probability distribution over model structures and parameters, and is updated using the data to obtain a posterior probability distribution over models and parameters. More formally, assuming a prior distribution over models structures $P(\mathcal{M})$ and a prior distribution over parameters for each model structure $P(\theta|\mathcal{M})$, a data set $\mathcal{D}$ is used to form a posterior distribution over models using Bayes rule

$$P(\mathcal{M}|\mathcal{D}) = \frac{\int P(\mathcal{D}|\theta,\mathcal{M})P(\theta|\mathcal{M})\, d\theta\; P(\mathcal{M})}{P(\mathcal{D})} \tag{10}$$

which averages over the uncertainty in the parameters. For a given model structure, we can compute the posterior distribution over the parameters:

$$P(\theta|\mathcal{M},\mathcal{D}) = \frac{P(\mathcal{D}|\theta,\mathcal{M})P(\theta|\mathcal{M})}{P(\mathcal{D}|\mathcal{M})}.$$

If the data set is some sequence of observations $\mathcal{D} = \{Y_1, \ldots, Y_T\}$ and we wish to predict the next observation, Y_{T+1} based on our data and models, then the Bayesian prediction

$$P(Y_{T+1}|\mathcal{D}) = \int P(Y_{T+1}|\theta,\mathcal{M},\mathcal{D})P(\theta|\mathcal{M},\mathcal{D})P(\mathcal{M}|\mathcal{D})\, d\theta\, d\mathcal{M}$$

averages over both the uncertainty in the model structure and in the parameters. This is known as the *predictive distribution* for the model.

We obtain a somewhat impoverished by nonetheless useful limiting case of the Bayesian approach to learning if we assume a single model structure $\mathcal{M}$ and we estimate the parameter vector $\hat{\theta}$ that maximises the likelihood $P(\mathcal{D}|\theta,\mathcal{M})$ under that model. In the limit of a large data set and an uninformative (e.g. uniform) prior over the parameters, the posterior $P(\theta|\mathcal{M},\mathcal{D})$ will be sharply peaked around the maxima of the likelihood, and therefore the predictions of a single maximum likelihood (ML) model will be similar to those obtained by Bayesian integration over the parameters.

For now we focus on the problem of estimating ML parameters for a given model structure. Although this is the most widely used approach to learning hidden Markov models, it only crudely approximates Bayesian learning, and can perform catastrophically when data is scarce and/or the model is complex. In practice an exact Bayesian analysis of HMMs is impractical, which is why most research has focused on ML approaches or regularised ML approaches. In section 7 we will tackle practical approaches to full-fledged Bayesian learning.

4.1 ML Estimation with Complete Data

Assume we are given a data set of independent and identically distributed observations $\mathcal{D} = \{Y^{(1)}, \ldots, Y^{(N)}\}$, each of which can be a vector or time series of vectors. The likelihood of the data set is a function of the parameters of the model:

$$P(\mathcal{D}|\theta,\mathcal{M}) = \prod_{i=1}^{N} P(Y^{(i)}|\theta,\mathcal{M})$$

For notational convenience we henceforth drop the implicit conditioning on the model structure, $\mathcal{M}$. The ML parameters are obtained by maximising the likelihood, or equivalently the log likelihood:

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} \log P(Y^{(i)}|\theta).$$

If the observation vector includes all the variables in the Bayesian network, then each term in the log likelihood further factors as:

$$\begin{aligned}\log P(Y^{(i)}|\theta) &= \log \prod_j P(Y_j^{(i)}|Y_{\mathrm{pa}(j)}^{(i)}, \theta_j) &(11)\\ &= \sum_j \log P(Y_j^{(i)}|Y_{\mathrm{pa}(j)}^{(i)}, \theta_j), &(12)\end{aligned}$$

where j indexes over the nodes in the Bayesian network, $\mathrm{pa}(j)$ is the set of parents of j, and θ_j are the parameters that define the conditional probability of Y_j given its parents. The likelihood therefore decouples into local terms involving each node and its parents, simplifying the ML estimation problem. For example, if the Y variables are discrete and θ_j is the conditional probability table for Y_j given its parents, then the ML estimate of θ_j is simply a normalised table containing counts of each setting of Y_j given each setting of its parents in the data set.

4.2 ML Estimation with Hidden Variables: The EM algorithm

With hidden variables the log likelihood cannot be decomposed as in (12). Rather, we find:

$$\mathcal{L}(\theta) = \log P(Y|\theta) = \log \sum_X P(Y, X|\theta) \tag{13}$$

where X is the set of hidden variables, and $\sum_X$ is the sum (or integral) over X required to obtain the marginal probability of the data. (We have dropped the superscript (i) in (13) by evaluating the log likelihood for a single observation, again for notational convenience.) Maximising (13) directly is often difficult because the log of the sum can potentially couple all of the parameters of the model. We can simplify the problem of maximising $\mathcal{L}$ with respect to θ by making use of the following insight. Any distribution $Q(X)$ over the hidden variables defines a *lower bound* on $\mathcal{L}$:

$$\begin{aligned}
\log \sum_X P(Y, X|\theta) &= \log \sum_X Q(X) \frac{P(Y, X|\theta)}{Q(X)} && (14)\\
&\geq \sum_X Q(X) \log \frac{P(X, Y|\theta)}{Q(X)} && (15)\\
&= \sum_X Q(X) \log P(X, Y|\theta) - \sum_X Q(X) \log Q(X) && (16)\\
&= \mathcal{F}(Q, \theta) && (17)
\end{aligned}$$

where the inequality is known as Jensen's inequality and follows from the fact that the log function is concave. If we define the *energy* of a global configuration (X, Y) to be $\log P(X, Y|\theta)$, then some readers may notice that the lower bound $\mathcal{F}(Q, \theta) \leq \mathcal{L}(\theta)$ is the negative of a quantity known in statistical physics as the *free energy*: the expected energy under Q minus the entropy of Q [42]. The Expectation-Maximization (EM) algorithm [4, 12] alternates between maximising $\mathcal{F}$ with respect to Q and θ, respectively, holding the other fixed. Starting from some initial parameters θ_0:

$$\textbf{E step:} \qquad Q_{k+1} \leftarrow \underset{Q}{\arg\max}\ \mathcal{F}(Q, \theta_k) \tag{18}$$

$$\textbf{M step:} \qquad \theta_{k+1} \leftarrow \underset{\theta}{\arg\max}\ \mathcal{F}(Q_{k+1}, \theta) \tag{19}$$

It is easy to show that the maximum in the E step is obtained by setting $Q_{k+1}(X) = P(X|Y, \theta_k)$, at which point the bound becomes an equality: $\mathcal{F}(Q_{k+1}, \theta_k) = \mathcal{L}(\theta_k)$. The maximum in the M step is obtained by maximising the first term in (16), since the entropy of Q does not depend on θ:

$$\textbf{M step:} \qquad \theta_{k+1} \leftarrow \underset{\theta}{\arg\max}\ \sum_X P(X|Y, \theta_k) \log P(X, Y|\theta).$$

This is the expression most often associated with the EM algorithm [12], but it obscures the elegant interpretation of EM as coordinate ascent in $\mathcal{F}$. Since $\mathcal{F} = \mathcal{L}$ at the beginning of each M step, and since the E step does not change θ, we are guaranteed not to decrease the likelihood after each combined EM step.

It is worthwhile to point out that it is usually not necessary to explicitly evaluate the posterior distribution $P(X|Y, \theta_k)$. Since $\log P(X, Y|\theta)$ contains both hidden and observed variables in the network, it can be factored as before as the sum of log probabilities of each node given its parents. Consequently, the quantities required for the M step are the expected values, under the posterior distribution $P(X|Y, \theta_k)$, of the analogous sufficient statistics required for ML estimation in the complete data case.

4.3 Example 1: Learning hidden Markov models using EM

To derive the EM algorithm for learning the parameters of an HMM we first need to write out the log probability of the hidden variables and observations:

$$\log P(S_{1:T}, Y_{1:T}) = \log P(S_1) + \sum_{t=1}^{T} \log P(Y_t|S_t) + \sum_{t=2}^{T} \log P(S_t|S_{t-1}). \tag{20}$$

Let us represent the K-valued discrete state S_t using K-dimensional unit column vectors, e.g. the state at time t taking on the value "2" is represented as $S_t = [010\ldots0]^\top$. Each of the terms in (20) can be decomposed into summations over S. For example, the transition probability is

$$P(S_t|S_{t-1}) = \prod_{i=1}^{K}\prod_{j=1}^{K} (\Phi_{ij})^{S_{t,i}S_{t-1,j}}$$

where Φ_{ij} is the probability of transitioning from state j to state i, arranged in a $K \times K$ matrix Φ. Then

$$\begin{aligned} \log P(S_t|S_{t-1}) &= \sum_{i=1}^{K}\sum_{j=1}^{K} S_{t,i}S_{t-1,j} \log \Phi_{ij} && (21) \\ &= S_t^\top (\log \Phi) S_{t-1} && (22) \end{aligned}$$

using matrix notation, where $^\top$ is the matrix transpose (not to be confused with the sequence length T), and logarithms of vectors and matrices are taken elementwise. Similarly, if the initial state probabilities are arranged in a vector $\boldsymbol{\pi}$, then

$$\log P(S_1) = S_1^\top \log \boldsymbol{\pi}. \tag{23}$$

Finally, the emission probabilities depend on the form of the observations. If Y_t is a discrete variable which can take on D values, then we again represent it using D-dimensional unit vectors and obtain

$$\log P(Y_t|S_t) = Y_t^\top (\log E) S_t$$

where E is a $D \times K$ emission probability matrix. The parameter set for the HMM is $\theta = \{\Phi, \boldsymbol{\pi}, E\}$.

Since the state variables are hidden we cannot compute (20) directly. The EM algorithm, which in the case of HMMs is known as the Baum-Welch algorithm [4], allows us to circumvent this problem by computing the expectation of (20) under the posterior distribution of the hidden states given the observations. We denote the expected value of some quantity $f(X)$ with respect to the posterior distribution of X by $\langle f(X) \rangle$,

$$\langle f(X) \rangle = \int_X f(X)\; P(X|Y,\theta)\; dX. \tag{24}$$

The expected value of (20) can be expressed as a function of $\langle S_t \rangle$ and $\langle S_t S_{t-1}^\top \rangle$ $(1 \le t \le T)$. The first term, $\langle S_t \rangle$, is a vector containing the probability that the HMM was in each of the K states at time t given its current parameters and the entire sequence of observations[6]. The second term, $\langle S_t S_{t-1}^\top \rangle$, is a matrix containing the joint probability that the HMM was in each of the K^2 pairs of states at times $t-1$ and t. In the HMM notation of [45], $\langle S_t \rangle$ corresponds to γ_t and $\langle S_t S_{t-1}^\top \rangle$ corresponds to ξ_t. Given these expectations, the M step is straightforward: we take derivatives of (20) with respect to the parameters, set to zero, and solve subject to the sum-to-one constraints that ensure valid transition, emission and initial state probabilities. For discrete Y_t coded in the same way as S_t (i.e. Y_t is coded as a D-dimensional binary unit vector), the M step is:

$$\Phi_{ij} \quad \propto \quad \sum_{t=2}^{T} \langle S_{t,i} S_{t-1,j} \rangle \tag{25}$$

[6]When learning from a data set containing multiple sequences, this quantity has to be computed separately for each sequence. For clarity, we will describe the single sequence case only.

$$\leftarrow \quad \frac{\sum_{t=2}^{T} \langle S_{t,i}\ S_{t-1,j} \rangle}{\sum_{t=2}^{T} \langle S_{t-1,j} \rangle} \tag{26}$$

$$\boldsymbol{\pi}_i \quad \leftarrow \quad \langle S_{1,i} \rangle \tag{27}$$

$$E_{di} \quad \leftarrow \quad \frac{\sum_{t=1}^{T} Y_{t,d} \langle S_{t,i} \rangle}{\sum_{t=1}^{T} \langle S_{t,i} \rangle}. \tag{28}$$

The necessary expectations are computed using the forward–backward algorithm.

4.4 The forward–backward algorithm

The forward–backward algorithm is an instance of belief propagation applied to the Bayesian network corresponding to a hidden Markov model (see [54] for a recent treatment). The forward pass recursively computes α_t, defined as the joint probability of S_t and the sequence of observations Y_1 to Y_t:

$$\alpha_t \quad = \quad P(S_t, Y_{1:t}) \tag{29}$$

$$= \quad \left[\sum_{S_{t-1}} P(S_{t-1}, Y_{1:t-1}) P(S_t | S_{t-1}) \right] P(Y_t | S_t) \tag{30}$$

$$= \quad \left[\sum_{S_{t-1}} \alpha_{t-1} P(S_t | S_{t-1}) \right] P(Y_t | S_t). \tag{31}$$

The backward pass computes the conditional probability of the observations Y_{t+1} to Y_T given S_t:

$$\beta_t \quad = \quad P(Y_{t+1:T} | S_t) \tag{32}$$

$$= \quad \sum_{S_{t+1}} P(Y_{t+2:T} | S_{t+1}) P(S_{t+1} | S_t) P(Y_{t+1} | S_{t+1}) \tag{33}$$

$$= \quad \sum_{S_{t+1}} \beta_{t+1} P(S_{t+1} | S_t) P(Y_{t+1} | S_{t+1}). \tag{34}$$

From these it is easy to compute the expectations needed for EM:

$$\langle S_{t,i} \rangle = \gamma_{ti} \quad = \quad \frac{\alpha_{t,i} \beta_{t,i}}{\sum_j \alpha_{t,j} \beta_{t,j}} \tag{35}$$

$$\langle S_{t,i} S_{t-1,j} \rangle = \xi_{tij} \quad = \quad \frac{\alpha_{t-1,j} \Phi_{ij} P(Y_t | S_{t,i}) \beta_{t,i}}{\sum_{k,\ell} \alpha_{t-1,k} \Phi_{k\ell} P(Y_t | S_{t,\ell}) \beta_{t,\ell}}. \tag{36}$$

In practice for long sequences both the α_t and β_t become vanishingly small as the recursions progress. They are therefore usually renormalised to sum to one at each step of the recursions. This makes the computation of the relevant expectations much more numerically well-behaved, and has the nice side-effect that the sum of the log normalizations in the forward pass is the log likelihood of the observation sequence.

Occasionally, it is also useful to compute the single most probable state sequence. The solution to this problem is given by the *Viterbi algorithm* [57], which is very similar to the forward–backward algorithm except that some of the summations are replaced by maximisations (see [45] for a tutorial on HMMs, especially as applied to speech recognition).

4.5 Example 2: Learning state-space models using EM

Using equation (6), the log probability of the hidden states and observations for linear-Gaussian state-space models can be written as

$$\log P(X_{1:T}, Y_{1:T}) = \log P(X_1) + \sum_{t=1}^{T} \log P(Y_t | X_t) + \sum_{t=2}^{T} \log P(X_t | X_{t-1}). \tag{37}$$

Each of the above probability densities is Gaussian, and therefore the overall expression is a sum of quadratics. For example, using equation (8):

$$\log P(Y_t|X_t) = -\frac{1}{2}(Y_t - CX_t)^\top R^{-1}(Y_t - CX_t) - \frac{1}{2}|R| + const \tag{38}$$

where R is the covariance of the observation noise v_t and $|\cdot|$ is the matrix determinant.

If all the random variables were observed, then the ML parameters could be solved for by maximising (37). Taking derivatives of (37) we obtain a linear systems of equations. For example, the ML estimate of the matrix C is

$$C \leftarrow \left(\sum_t Y_t X_t^\top\right) \left(\sum_t X_t X_t^\top\right)^{-1}.$$

Since the states are in fact hidden, in the M step we use expected values wherever we don't have access to the actual observed values. Then, the M step for C is

$$C \leftarrow \left(\sum_t Y_t \langle X_t\rangle^\top\right) \left(\sum_t \langle X_t X_t^\top\rangle\right)^{-1}.$$

Similar M steps can be derived for all the other parameters by taking derivatives of the expected log probability [52, 13, 20].[7] In general we require all terms of the kind $\langle X_t\rangle$, $\langle X_t X_t^\top\rangle$ and $\langle X_t X_{t-1}^\top\rangle$. These terms can be computed using the Kalman smoothing algorithm.

4.6 Kalman smoothing

The Kalman smoother solves the problem of estimating the state at time t of a linear-Gaussian state-space model given the model parameters and a sequence of observations $\{Y_1, \ldots, Y_t, \ldots, Y_T\}$. It consists of two parts: a forward recursion which uses the observations from Y_1 to Y_t, known as the *Kalman filter* [33], and a backward recursion which uses the observations from Y_T to Y_{t+1} [46].[8] We have already seen that in order to compute the marginal probability of a variable in a Bayesian network one must take into account both the evidence above and below the variable. In fact, the Kalman smoother is simply a special case of the belief propagation algorithm we have already encountered for Bayesian networks. The Kalman smoothing algorithm and the forward–backward algorithm are conceptually identical, although of course the details differ since in one Gaussian densities are propagated and in the other discrete distributions are propagated.

5 Limitations of HMMs and Generalisations

Linear-Gaussian state-space models and hidden Markov models provide an interesting starting point for designing dynamic Bayesian networks. However, they suffer from important limitations when it comes to modeling real world time series. In the case of linear-Gaussian state-space models the limitations are advertised in the name: in many realistic applications, both the state dynamics and the relation between states and observations can be nonlinear, and the noise can be non-Gaussian. For hidden Markov models, the situation is more subtle. HMMs are a dynamical extension of mixture models, and unconstrained mixture models can be used to model any distribution in the limit of an infinite number of mixture components. Furthermore, if the state transition matrix is unconstrained, any arbitrary nonlinear dynamics can also be modeled. So where does the limitation lie?

Consider the problem of modeling the movement of several objects in a sequence of images. If there are M objects, each of which can occupy K positions and orientations in the image, there are K^M possible states of the system underlying an image. A hidden Markov model would require K^M distinct states to model this system. This representation is not only inefficient but difficult to interpret. We would much prefer that

[7]The parameters of a linear-Gaussian state-space model can also be estimated using methods from on-line recursive identification [38].

[8]The forward and backward recursions together are also known as the Rauch-Tung-Streibel (RTS) smoother. Thorough treatments of Kalman filtering and smoothing can be found in [1, 23].

our "HMM" could capture the underlying state space by using M different K-dimensional variables. More seriously, an unconstrained HMM with K^M states has of order K^{2M} parameters in the transition matrix. Unless the data set captures all these possible transitions or a priori knowledge is used to constrain the parameters, severe over-fitting may result.

In this section, we describe three ways in which HMMs and state-space models can be extended to overcome some of these limitations. The first of these represents the hidden state of an HMM using a set of distinct state variables. We call this HMM with a distributed state representation, a *factorial hidden Markov model* [22].

5.1 Extension 1: Factorial HMMs

We generalise the HMM by representing the state using a collection of discrete state variables

$$S_t = S_t^{(1)}, \ldots S_t^{(m)}, \ldots, S_t^{(M)}, \tag{39}$$

each of which can take on $K^{(m)}$ values. The state space of this model consists of the cross product of these state variables. For simplicity, we will assume that $K^{(m)} = K$, for all m, although the algorithms we present can be trivially generalised to the case of differing $K^{(m)}$. Given that the state space of this factorial HMM consists of all K^M combinations of the $S_t^{(m)}$ variables, placing no constraints on the state transition structure would result in a $K^M \times K^M$ transition matrix. Such an unconstrained system is uninteresting for several reasons: it is equivalent to an HMM with K^M states; it is unlikely to discover any interesting structure in the K state variables, as all variables are allowed to interact arbitrarily; and both the time complexity and sample complexity of the estimation algorithm are exponential in M.

We therefore focus on factorial HMMs in which the underlying state transitions are constrained. A natural structure to consider is one in which each state variable evolves according to its own dynamics, and is *a priori* uncoupled from the other state variables:

$$P(S_t|S_{t-1}) = \prod_{m=1}^{M} P(S_t^{(m)}|S_{t-1}^{(m)}). \tag{40}$$

A Bayesian network representing this model is shown in Figure 5. The transition structure for this model can be parameterised using M distinct $K \times K$ matrices.

As shown in Figure 5, the observation at time step t can depend on all the state variables at that time step in a factorial HMM. For real-valued observations, one simple form for this dependence is linear-Gaussian; that is, the observation Y_t is a Gaussian random vector whose mean is a linear function of the state variables. We represent the state variables as $K \times 1$ vectors, where each of the K discrete values corresponds to a 1 in one position and 0 elsewhere. The resulting probability density for a $D \times 1$ observation vector Y_t is

$$P(Y_t|S_t) = |R|^{-1/2}\,(2\pi)^{-D/2}\,\exp\left\{-\frac{1}{2}\,(Y_t - \mu_t)^\top R^{-1}\,(Y_t - \mu_t)\right\}, \tag{41}$$

where

$$\mu_t = \sum_{m=1}^{M} W^{(m)} S_t^{(m)}. \tag{42}$$

Each $W^{(m)}$ matrix is a $D \times K$ matrix whose columns are the contributions to the means for each of the settings of $S_t^{(m)}$, and R is a $D \times D$ covariance matrix.

One way to understand the observation model in equations (41) and (42) is to consider the marginal distribution for Y_t, obtained by summing over the possible states. There are K settings for each of the M state variables, and thus there are K^M possible mean vectors obtained by forming sums of M columns where one column is chosen from each of the $W^{(m)}$ matrices. The resulting marginal density of Y_t is thus a Gaussian mixture model with K^M mixture components each having a constant covariance matrix R. This static mixture model, without inclusion of the time index and the Markov dynamics, is a factorial parameterization of the standard mixture of Gaussians model that has interest in its own right [59, 25, 18]. The model we have just presented extends the static model by allowing Markov dynamics in the discrete state variables underlying the mixture. A model of that combines features of the factorial HMM and factor analysis has been recently applied to an image tracking problem with impressive results [14]

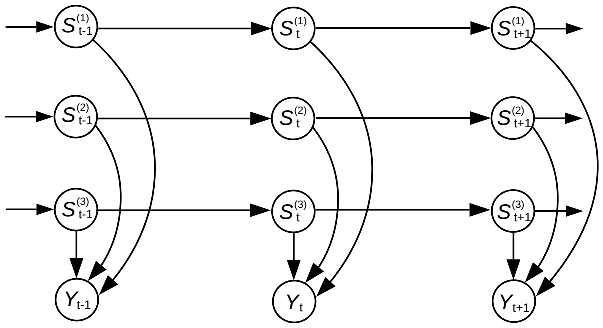

Figure 5: A Bayesian network representing the conditional independence relations in a factorial HMM with $M = 3$ underlying Markov chains. (We only show here a portion of the Bayesian network around time slice t.)

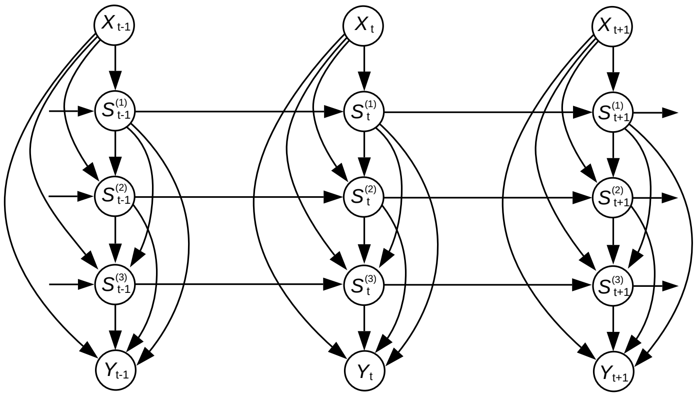

Figure 6: Tree structured hidden Markov models.

5.2 Extension 2: Tree structured HMMs

In factorial HMMs, the state variables at one time step are assumed to be *a priori* independent given the state variables at the previous time step. This assumption can be relaxed in many ways by introducing coupling between the state variables in a single time step [49]. One interesting way to couple the variables is to order them, such that $S_t^{(m)}$ depends on $S_t^{(n)}$ for $1 \leq n < m$. Furthermore, if all the state variables and the output also depend on an observable input variable, X_t, we obtain the Bayesian network shown in Figure 6.

This architecture can be interpreted as a probabilistic decision tree with Markovian dynamics linking the decision variables. Consider how this model would generate data at the first time step, $t = 1$. Given input X_1, the top node $S_1^{(1)}$ can take on K values. This stochastically partitions X-space into K decision regions. The next node down the hierarchy, $S_1^{(2)}$, subdivides each of these regions into K subregions, and so on. The output Y_1 is generated from the input X_1 and the K-way decisions at each of the M hidden nodes. At the next time step, a similar procedure is used to generate data from the model, except that now each decision in the tree is dependent on the decision taken at that node in the previous time step. This model generalises the "hierarchical mixture of experts" [31] and other related decision tree models such as CART [8] and MARS [15] by giving the decisions Markovian dynamics. Tree structured HMMs provide a useful starting point for modeling time series with both temporal and spatial structure at multiple resolutions. We have explored this generalization of factorial HMMs in [30].

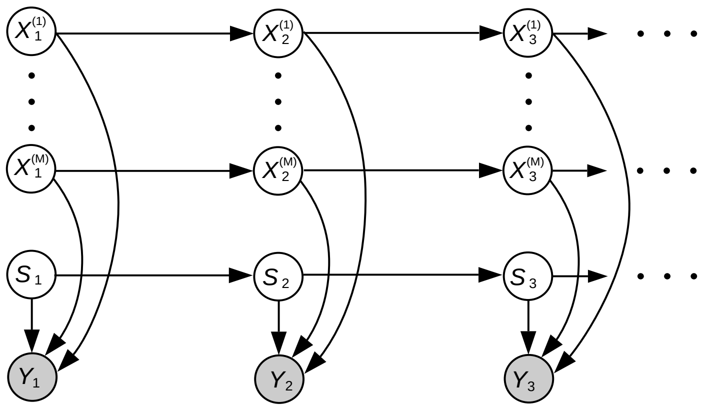

Figure 7: Bayesian network representation for switching state-space models. S_t is the discrete switch variable and $X_t^{(m)}$ are the real-valued state vectors.

5.3 Extension 3: Switching State space models

Both factorial HMMs and tree-structured HMMs use discrete hidden state representations. To model time series with continuous but nonlinear dynamics, it is possible to combine the real-valued hidden state of linear-Gaussian state-space models and the discrete state of HMMs. One natural way to do this is the *switching state-space model* [21].

In switching state-space models, the sequence of observations $Y_{1:T}$ is modeled using a hidden state space comprising M real-valued state vectors, $X_t^{(m)}$, and one discrete state vector S_t. The discrete state, S_t, is a multinomial variable that can take on M values: $S_t \in \{1, \dots, M\}$; for reasons that will become obvious we refer to it as the *switch* variable. The joint probability of observations and hidden states can be factored as

$$\begin{aligned} P(S_{1:T}, X_{1:T}^{(1)}, \dots, X_{1:T}^{(M)}, Y_{1:T}) &= P(S_1) \prod_{t=2}^{T} P(S_t|S_{t-1}) \prod_{m=1}^{M} \left[P(X_1^{(m)}) \prod_{t=2}^{T} P(X_t^{(m)}|X_{t-1}^{(m)}) \right] \\ &\quad \times \prod_{t=1}^{T} P(Y_t|X_t^{(1)}, \dots, X_t^{(M)}, S_t), \end{aligned} \tag{43}$$

which corresponds graphically to the conditional independencies represented by Figure 7. Conditioned on a setting of the switch state, $S_t = m$, the observable is multivariate Gaussian with output equation given by state-space model m. In other words, the probability density of the observation vector Y_t is

$$\begin{aligned} P(Y_t|X_t^{(1)}, \dots, X_t^{(M)}, S_t = m) &= (2\pi)^{-\frac{D}{2}} |R|^{-\frac{1}{2}} \times \\ &\quad \exp\left\{ -\tfrac{1}{2} \left(Y_t - C^{(m)} X_t^{(m)} \right)^\top R^{-1} \left(Y_t - C^{(m)} X_t^{(m)} \right) \right\} \end{aligned} \tag{44}$$

where D is the dimension of the observation vector, R is the observation noise covariance matrix, and $C^{(m)}$ is the output matrix for state-space model m (cf. equation (8) for a single linear-Gaussian state-space model). Each real-valued state vector evolves according to the linear-Gaussian dynamics of a state-space model with differing initial state, transition matrix, and state noise (equation (7)). The switch state itself evolves according to the discrete Markov dynamics specified by initial state probabilities $P(S_1)$ and an $M \times M$ state transition matrix $P(S_t|S_{t-1})$.

This model can be seen as an extension of the "mixture of experts" architecture for modular learning in neural networks [27, 9, 40]. Each state-space model is a linear expert with Gaussian output noise and linear-Gaussian dynamics. The switch state "gates" the outputs of the M state-space models, and therefore plays the role of a gating network with Markovian dynamics [9, 40].

6 Approximate Inference and Intractability

The problem with all the extensions of hidden Markov models and state-space models presented in the previous section is that, given a sequence of observations, most probabilities of interest are intractable to compute.

Consider, for example, computing the likelihood of a factorial HMM—the marginal probability of a sequence of observations given the parameters, $P(Y_{1:T}|\theta)$. This is the sum over all possible hidden state sequences of the joint probability of the sequence and the observations:

$$P(Y_{1:T}|\theta) = \sum_{S_{1:T}} P(S_{1:T}, Y_{1:T}|\theta).$$

There are K^M possible states at each time step, and therefore K^{MT} hidden state sequences of length T, assuming none of the transition probabilities is exactly 0. The brute-force approach of evaluating all such sequences can be avoided by making use of the conditional independencies represented in the Bayesian network. For example, directly applying the forward pass of the forward–backward algorithm outlined in section 4.4, we can compute the likelihood by summing the α's at the last time step

$$\begin{aligned} P(Y_{1:T}|\theta) &= \sum_{S_T} P(S_T, Y_1, \ldots, Y_T|\theta) && (45) \\ &= \sum_{S_T} \alpha_T. && (46) \end{aligned}$$

For the factorial HMM, α_t is a vector of size equal to the full state space at time t, i.e. it has K^M elements. This results in a recursive algorithm that computes the likelihood using $O(TK^{2M})$ operations. This can be further improved upon by using the fact that the state transitions are defined via M matrices of size $K \times K$ rather than a single $K^M \times K^M$ matrix, resulting in a recursive algorithm using $O(TMK^{M+1})$ operations (see [22], appendix B). Unfortunately, this time complexity cannot be improved upon. Given the observation at time t, the K-valued state variables become coupled in an M^{th} order interaction. It is not possible to sum over each variable independently. Like the likelihood, computing the posterior probability of a single state variable given the observation sequence, $P(S_t^{(m)}|Y_1, \ldots, Y_T)$, is also exponential in M. Similar exponential time complexity results hold for the likelihoods and posterior probabilities of tree-structured HMMs and switching state-space models.

6.1 Approximation 1: Gibbs sampling

One approach to computing approximate marginal probabilities is to make use of Monte Carlo integration. Since the log likelihood can be expressed as

$$\log P(Y_{1:T}|\theta) = \sum_{S_{1:T}} P(S_{1:T}|Y_{1:T}, \theta) \left[\log P(S_{1:T}, Y_{1:T}, \theta) - \log P(S_{1:T}|Y_{1:T}, \theta)\right],$$

by sampling from the posterior distribution, $P(S_{1:T}|Y_{1:T}, \theta)$, the log likelihood can be approximated using the above expression, which is just the negative of the free energy (16). To learn the parameters of the model, samples from the posterior are used to evaluate the expectations required for EM. Of course, for intractable models sampling directly from the posterior distributions is computationally prohibitive. However, it is often easy to set up a Markov chain that will converge to samples from the posterior. One of the simplest methods to achieve this is Gibbs sampling (for a review of Gibbs sampling and other Markov chain Monte Carlo methods, see [41]).

For a given observation sequence $Y_{1:T}$, Gibbs sampling starts with a random setting of the hidden states $S_{1:T}$. At each step of the sampling process, each state variable is updated stochastically according to its probability distribution conditioned on the setting of all the other state variables. The graphical model is again useful here, as each node is conditionally independent of all other nodes given its *Markov blanket*, defined as the set of children, parents, and parents of the children of a node. For example, to sample from a typical state variable $S_t^{(m)}$ in a factorial HMM we only need to examine the states of a few neighboring nodes:

$$\begin{aligned}
S_t^{(m)} &\sim P(S_t^{(m)}|\{S_\tau^{(n)} : \tau \neq t \ \vee \ n \neq m\}, Y_{1:T}) && (47)\\
&= P(S_t^{(m)}|\{S_t^{(n)} : n \neq m\}, S_{t-1}^{(m)}, S_{t+1}^{(m)}, Y_t) && (48)\\
&\propto P(S_t^{(m)}|S_{t-1}^{(m)})\, P(S_{t+1}^{(m)}|S_t^{(m)})\, P(Y_t|S_t^{(1)}, \ldots, S_t^{(m)}, \ldots, S_t^{(M)}), && (49)
\end{aligned}$$

where $\sim$ denotes "sampled from". Sampling once from each of the TM hidden variables in the model results in a new sample of the hidden state of the model and requires $O(TMK)$ operations. The sequence of states resulting from each pass of Gibbs sampling defines a Markov chain over the state space of the model. This Markov chain is guaranteed to converge to the posterior probabilities of the states given the observations [17] as long as none of the probabilities in the model is exactly zero[9]. Thus, after some suitable time, samples from the Markov chain can be taken as approximate samples from the posterior probabilities. The first and second-order statistics needed to estimate $\langle S_t^{(m)} \rangle$, $\langle S_t^{(m)} S_t^{(n)'} \rangle$ and $\langle S_{t-1}^{(m)} S_t^{(m)'} \rangle$ are collected using the states visited and the probabilities estimated during this sampling process and are used in the approximate E step of EM.[10] Monte Carlo methods for learning in dynamic Bayesian networks have been explored by [11, 34, 10, 22].

6.2 Approximation 2: Variational Methods

Another approach to approximating a probability distribution P is to define a parameterised distribution Q and vary its parameters so as to minimise the distance between Q and P. In the context of the EM algorithm, we have already seen that the log likelihood $\mathcal{L}(\theta)$ is lower bounded by the negative free energy, $\mathcal{F}(Q, \theta)$. The difference between $\mathcal{L}$ and $\mathcal{F}$ is given by the Kullback-Leibler divergence between Q and the posterior distribution of the hidden variables:

$$\begin{aligned}
\mathcal{L}(\theta) - \mathcal{F}(Q, \theta) &= \mathrm{KL}\left(Q(S_{1:T}|\phi) \,\|P(S_{1:T}|Y_{1:T}, \theta)\right) && (50)\\
&= \sum_{S_{1:T}} Q(S_{1:T}|\phi)\ \log\left[\frac{Q(S_{1:T}|\phi)}{P(S_{1:T}|Y_{1:T}, \theta)}\right] && (51)
\end{aligned}$$

where ϕ are the parameters of the distribution Q.

The variational approach uses a tractable Q to approximate the intractable posterior. The tractability of computing expectations with respect to Q depends both on its parametric form and on its conditional independence relations.[11] The art is to chose a family of Qs that have an analytic form and a tractable structure—a Bayesian network that eliminates some of the dependencies in P—but that can approximate P adequately. Given this structure, the parameters of Q are varied so as to obtain the tightest possible bound, which minimises (51). We will refer to the general strategy of using a parameterised approximating distribution as a *variational approximation* and refer to the free parameters of the Q distribution as *variational parameters*.

6.3 Example 1: Mean field for factorial HMMs

We illustrate this approach using the simplest variational approximation to the posterior distribution in factorial HMMs: all state variables in Q are independent (Figure 8 (a)):

$$Q(S_{1:T}|\phi) = \prod_{t=1}^{T} \prod_{m=1}^{M} Q(S_t^{(m)}|\phi_t^{(m)}). \qquad (52)$$

[9]Actually, the weaker assumption of ergodicity is sufficient to ensure convergence

[10]A more Bayesian treatment of the learning problem, in which the parameters are also considered hidden random variables, can be handled by Gibbs sampling by replacing the "M step" with sampling from the conditional distribution of the parameters given the other hidden variables (for example, see [56]).

[11]We will see later how choosing the conditional independence relations of Q sometimes determines the optimal parametric form of Q. This is "true" variational optimisation, since calculus of variations is used to optimise over all distributions Q.

(a) (b)

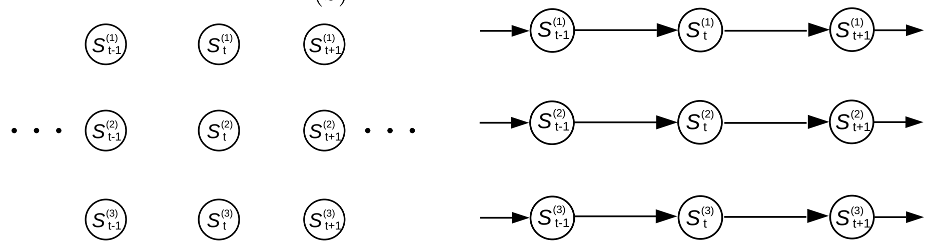

Figure 8: (a) The completely factorised variational approximation assuming that all the state variables are independent (conditional on the observation sequence). (b) A structured variational approximation assuming that the state variables retain their Markov structure within each chain, but are independent across chains.

The variational parameters, $\phi = \left\{\phi_t^{(m)}\right\}$, are the means of the state variables, where, as before, a state variable $S_t^{(m)}$ is represented as a K-dimensional vector with a 1 in the k^{th} position and 0 elsewhere, if the m^{th} Markov chain is in state k at time t. The elements of the vector $\phi_t^{(m)}$ therefore define the state occupation probabilities of the multinomial variable $S_t^{(m)}$ under the Q distribution:

$$Q(S_t^{(m)}|\phi_t^{(m)}) = \prod_{k=1}^{K} \left(\phi_{t,k}^{(m)}\right)^{S_{t,k}^{(m)}} \qquad \text{where} \quad S_{t,k}^{(m)} \in \{0,1\}; \quad \sum_{k=1}^{K} S_{t,k}^{(m)} = 1. \tag{53}$$

A completely factorised approximation of this kind is often used in statistical physics, where it provides the basis for simple yet powerful *mean field approximations* to statistical mechanical systems [43].

To make the bound as tight as possible we vary ϕ separately for each observation sequence so as to minimise the KL divergence. Taking the derivatives of (51) with respect to $\phi_t^{(m)}$ and setting them to zero, we obtain the set of fixed point equations defined by

$$\phi_t^{(m)\ \text{new}} = \varphi\left\{ {W^{(m)}}^\top R^{-1}\tilde{Y}_t^{(m)} - \frac{1}{2}\Delta^{(m)} + (\log\Phi^{(m)})\ \phi_{t-1}^{(m)} + (\log\Phi^{(m)})^\top\ \phi_{t+1}^{(m)} \right\} \tag{54}$$

where $\tilde{Y}_t^{(m)}$ is the reconstruction error in Y_t given the predictions from all the state variables not including m:

$$\tilde{Y}_t^{(m)} \equiv Y_t - \sum_{\ell \neq m}^{M} W^{(\ell)}\phi_t^{(\ell)}, \tag{55}$$

$\Delta^{(m)}$ is the vector of diagonal elements of ${W^{(m)}}^\top R^{-1} W^{(m)}$, and $\varphi\{\cdot\}$ is the softmax operator, which maps a vector $\mathbf{a}$ into a vector $\mathbf{b}$ of the same size, with elements

$$b_i = \frac{\exp\{a_i\}}{\sum_j \exp\{a_j\}}, \tag{56}$$

and $\log\Phi^{(m)}$ denotes the elementwise logarithm of the transition matrix $\Phi^{(m)}$ (see appendix C in [22] for details of the derivation).

The first term of (54) is the projection of the reconstruction error onto the weights of state vector m—the more a particular setting of a state vector can reduce this error, the larger its associated variational mean. The second term arises from the fact that the second order correlation $\langle S_t^{(m)} S_t^{(m)}\rangle$ evaluated under the variational distribution is a diagonal matrix composed of the elements of $\phi_t^{(m)}$. The last two terms introduce dependencies forward and backward in time.[12] Therefore, although the posterior distribution over the hidden variables is approximated with a completely factorised distribution, the fixed point equations couple

[12]The first term is replaced by $\log\pi^{(m)}$ for $t = 1$ the second term does not appear for $t = T$.

the parameters associated with each node with the parameters of its Markov blanket. In this sense, the fixed point equations propagate information along the same pathways as those defining the exact algorithms for probability propagation.

The following may provide an intuitive interpretation of the approximation made by this distribution. Given a particular observation sequence, the hidden state variables for the M Markov chains at time step t are stochastically coupled. This stochastic coupling is approximated by a system in which the hidden variables are uncorrelated but have coupled means. The variational or "mean-field" equations solve for the deterministic coupling of the means that best approximates the stochastically coupled system.

Each hidden state vector is updated in turn using (54), with a time complexity of $O(TMK^2)$ per iteration. Convergence is determined by monitoring the KL divergence in the variational distribution between successive time steps; in practice convergence is very rapid (about 2 to 10 iterations of (54)). Convergence to a global minimum of the KL divergence is not required, and in general this procedure will converge to a local minimum. Once the fixed point equations have converged, the expectations required for the E step can be obtained as a simple function of the parameters [22].

6.4 Example 2: Structured approximation for factorial HMMs

The approximation presented in the previous section factors the posterior probability into a product of statistically independent distributions over the state variables. Here we present another approximation which is tractable yet preserves many of the probabilistic dependencies in the original system. In this scheme, the posterior distribution of the factorial HMM is approximated by M uncoupled HMMs as shown in Figure 8 (b). Within each HMM, efficient and exact inference is implemented via the forward–backward algorithm. Since this approximation is allowed to have dependencies between the hidden variables it will generally be superior to the completely factorised mean-field approximation presented in the previous section, that is, the lower bound will be higher and the KL-divergence lower. The approach of exploiting such tractable substructures was first suggested in the machine learning literature by Saul and Jordan (1996).

We write the structured variational approximation as

$$Q(S_{1:T}|\phi) = \frac{1}{Z_Q} \prod_{m=1}^{M} Q(S_1^{(m)}|\phi) \prod_{t=2}^{T} Q(S_t^{(m)}|S_{t-1}^{(m)}, \phi), \tag{57}$$

where Z_Q is a normalization constant ensuring that Q sums to one. The parameters of Q are $\phi = \{\pi^{(m)}, \Phi^{(m)}, h_t^{(m)}\}$—the original priors and state transition matrices of the factorial HMM and a time-varying bias for each state variable. The prior and transition probabilities for Q are

$$Q(S_1^{(m)}|\phi) = \prod_{k=1}^{K} \left(h_{1,k}^{(m)} \pi_k^{(m)} \right)^{S_{1,k}^{(m)}} \tag{58}$$

$$\begin{aligned} Q(S_t^{(m)}|S_{t-1}^{(m)}, \phi) &= \prod_{k=1}^{K} \left(h_{t,k}^{(m)} \sum_{j=1}^{K} \Phi_{k,j}^{(m)} S_{t-1,j}^{(m)} \right)^{S_{t,k}^{(m)}} \\ &= \prod_{k=1}^{K} \left(h_{t,k}^{(m)} \prod_{j=1}^{K} \left(\Phi_{k,j}^{(m)} \right)^{S_{t-1,j}^{(m)}} \right)^{S_{t,k}^{(m)}}, \end{aligned} \tag{59}$$

where the last equality follows from the fact that $S_{t-1}^{(m)}$ is a vector with a 1 in one position and 0 elsewhere. Comparing equations (57)–(59) to equation (1), we can see that the $K \times 1$ vector $h_t^{(m)}$ plays the role of the probability of an observation ($P(Y_t|S_t)$ in (1)) for each of the K settings of $S_t^{(m)}$. For example, $Q(S_{1,j}^{(m)} = 1|\phi) = h_{1,j}^{(m)} \; P(S_{1,j}^{(m)} = 1|\phi)$ is equivalent to having an observation at time $t = 1$ that under state $S_{1,j}^{(m)} = 1$ has probability $h_{1,j}^{(m)}$.

Intuitively, this approximation uncouples the M Markov chains and attaches to each state variable a distinct fictitious observation. The probability of this fictitious observation can be varied so as to minimise the KL divergence between Q and P.

Applying the same arguments as before, we obtain a set of fixed point equations for $h_t^{(m)}$ that minimise $\mathrm{KL}(Q\|P)$:

$$h_t^{(m)\,\mathrm{new}} = \exp\left\{ W^{(m)^\top} R^{-1} \tilde{Y}_t^{(m)} - \frac{1}{2}\Delta^{(m)} \right\}, \tag{60}$$

where $\Delta^{(m)}$ is defined as before, and where we redefine the residual error to be

$$\tilde{Y}_t^{(m)} \equiv Y_t - \sum_{\ell \neq m}^{M} W^{(\ell)} \langle S_t^{(\ell)} \rangle. \tag{61}$$

The parameter $h_t^{(m)}$ obtained from these fixed point equations is the observation probability associated with state variable $S_t^{(m)}$ in hidden Markov model m. Using these probabilities, the forward–backward algorithm is used to compute a new set of expectations for $\langle S_t^{(m)} \rangle$, which are fed back into (60) and (61). The forward–backward algorithm is therefore used as a subroutine in the minimisation of the KL divergence.

Notice the similarity between equations (60)–(61) and equations (54)–(55) for the completely factorised approximation. In the completely factorised approximation, since $\langle S_t^{(m)} \rangle = \phi_t^{(m)}$, the fixed point equations can be written explicitly in terms of the variational parameters. In the structured approximation, the dependence of $\langle S_t^{(m)} \rangle$ on $h_t^{(m)}$ is computed via the forward–backward algorithm. Also, the fixed point equations (60) do not contain terms involving the prior, $\pi^{(m)}$, or transition matrix, $\Phi^{(m)}$. These terms are handled exactly by our choice of approximation.

The other intractable dynamic Bayesian networks we have presented are also amenable to structured variational approximations. In the case of tree-structured HMMs there are two natural choices for the substructures to retain in the approximation. One choice is to remove the arcs within a time step and retain the temporal dependencies, resulting in the Bayesian network shown in Figure 8 (b). The other choice is to retain the arcs *within* a time step and eliminate the arcs between consecutive time steps. Both of these approximations, along with an approximation based on the Viterbi, algorithm are pursued in [30].

For switching state-space models, the natural approximation is to make the M state-space models (SSMs) and the discrete Markov process controlling the switch variable stochastically independent. Again, the variational approximation couples all the SSMs and the switch variable deterministically, but this coupling can be computed tractably using Kalman smoothing on each state-space model separately and the forward–backward algorithm on the Markov switching process. The variational parameters are the real-valued "responsibilities" of each state-space model for each observation in the sequence. To determine the best variational parameters we start from some responsibilities and compute the posterior probability of the state in each SSM using Kalman smoothing, with the data *weighted by the responsibilities*. A weighting of 1 corresponds to applying the normal Kalman smoothing equations, whereas a weighting of 0 corresponds to assuming that the data was not observed at all; intermediate weighting are implemented by dividing the R matrix in (38) by the responsibility. We then recompute responsibilities by running the forward–backward algorithm on the switch process using the prediction error of each SSM. Iterating this procedure until the responsibilities converge decreases the KL-divergence. Details of this structured variational approximation for switching state-space models and experimental results are presented in [21].

7 Bayesian HMMs

We now turn to two very important issues in the learning of HMMs and other graphical models: overfitting and model selection. Overfitting refers to the scenario where the model fits the training set very well but generalises poorly to a test set chosen from the same data distribution. Overfitting is most prevalent when the training set is small relative to the complexity (i.e. number of free parameters) of the model, and there is nothing in the maximum likelihood fitting procedure itself to avoid it. Model selection, or learning model structure, is the closely related problem of picking a particular structure amongst several alternatives (or learning a distribution over these alternatives). In the case of HMMs the 'model structure' would include everything from the number of hidden states to the form of the state transition matrix and output probabilities. Model selection would also include whether to opt for a regular HMM or the more complex factorial HMM, for example. To learn model structure it is necessary to compare models of different complexity and again, there is nothing in ML parameter fitting that does this automatically.

There are three main ways to deal with the overfitting and model selection problems: cross-validation, regularisation, and Bayesian integration. *Cross-validation* repeatedly splits the training data into two sets: a new training set and a validation set, and compares how different model classes which have been trained on the new training set generalise to the validation set. This provides an estimate of the true generalisation error but can become computationally prohibitive if more than a few model structure parameters have to be determined.

Regularisation augments the likelihood objective function with a penalty term that favours simpler models over more complex models. For function approximation problems this regulariser is often of the form of a smoothness penalty on the function classes. In the case of neural network models the regulariser is usually expressed as some sort of weight decay term. While there are many ad hoc ways of picking regularisers, in the context of probabilistic modelling it is often illuminating to view regularisers are expressing a prior over the parameters, and the regularised ML fitting procedure as finding maximum a posteriori (MAP) parameters under such a prior. Thus the choice of regulariser can be assessed subjectively by asking whether the implicit prior over parameters "makes sense" to the modeller.

For HMMs with discrete outputs, a natural choice of prior over the parameters is given by the Dirichlet distribution. There are two main reasons for this[13]. First, the Dirichlet distribution has the mathematically convenient property of being conjugate to the multinomial distribution. A family of priors is said to be *conjugate* to a family of likelihoods if the posterior obtained by multiplying the prior by the likelihood is in the same family as the prior. For example, since the likelihood of the initial state given the parameter vector $\boldsymbol{\pi}$ is multinomial (cf equation (23)):

$$P(S_1|\boldsymbol{\pi}) = \prod_{i=1}^{K} \boldsymbol{\pi}_i^{S_{1i}},$$

if the prior probability of $\boldsymbol{\pi}$ is Dirichlet,

$$P(\boldsymbol{\pi}) = \frac{1}{Z} \prod_{i=1}^{K} \boldsymbol{\pi}_i^{u_i - 1}$$

with hyperparameter vector $\mathbf{u} = [u_1 \dots u_K]$ and normalisation constant Z, then the posterior is also Dirichlet:

$$P(\boldsymbol{\pi}|S_1) = \frac{1}{Z'} \prod_{i=1}^{K} \boldsymbol{\pi}_i^{u_i + S_{1i} - 1}.$$

Similar Dirichlet priors can be set up for columns of the transition matrix Φ and emission matrix E.

Second the Dirichlet distribution has the desirable property that its hyperparameters can be interpreted as hypothetical counts of observations. In the above example, if $u_i = 2$ and $u_j = 1$ for $j \neq i$, the MAP estimate of $\boldsymbol{\pi}$ is identical to an ML estimate of $\boldsymbol{\pi}$ with a training set augmented with one additional observation of the initial state being in state i. This makes it possible to implement MAP estimation with Dirichlet priors as a minor variant of the Baum-Welch procedure. It also gives some theoretical justification for the seemingly ad hoc but very common regularisation method for HMMs which just adds a small positive number to all elements of the parameter vector.

As outlined at the beginning of section 4, a Bayesian approach to learning treats all unknown quantities as random variables, assigns priors to these quantities, and infers posterior probabilities having observed the data. In the case of HMMs, these unknown quantities comprise the structure of the HMM (e.g. number of states), the parameters, and the hidden states. Unlike ML and MAP, which find point estimates of the parameters, we can now compare between model structures, but we need to integrate over both the parameters and the hidden states. We call this approach *Bayesian integration*.

There are several methods for approximating the required integrals, which for HMMs and their extensions are intractable. We briefly mention four of these methods. First let us make clear which integral we are referring to. To compare models it is necessary to compute the posterior probability of a model, which is

[13]Other, more theoretically motivated reasons are provided in [16].

proportional to the product of the prior and the marginal likelihood, also known as the *evidence* (cf equation (10)):

$$P(\mathcal{M}|\mathcal{D}) \propto P(\mathcal{M}) \ \left[\int P(\mathcal{D}|\theta,\mathcal{M})P(\theta|\mathcal{M})\ d\theta\right].$$

The evidence, bracketed here, is a high dimensional, often multimodal, intractable integral.

Monte Carlo methods approximate the integral by taking samples from regions of high probability. This can itself be hard, but can be made easier by setting up a Markov chain to converge to the correct equilibrium distribution [47].

Laplace approximations invoke the central limit theorem, which for well-behaved priors and data asserts that the parameter posterior will converge in the limit of large number of training samples to a Gaussian around the MAP estimate of the parameters.[14] To estimate the evidence using the Laplace approximation MAP parameters are found in the usual optimisation routines and then the curvature (Hessian) of the log likelihood is computed at the MAP estimate. The evidence is approximated by evaluating the ratio $P(\theta,\mathcal{D})/P(\theta|\mathcal{D})$ at the MAP estimate of θ, using the Gaussian approximation in the denominator. The Laplace approximation suffers from several disadvantages; here we mention two. First, computing the Hessian matrix for the parameters is usually very computationally costly. Second, the Gaussian approximation is not very good for models with parameters which are positive and sum to one, especially when there are many parameters relative to the size of the data set. To our knowledge the Laplace approximation has not been used for HMMs.

Stolke and Omohundro [55] present an ingenious method for approximating the Bayesian integrals for HMMs. If the states of the HMM were observed rather than hidden and if the parameter priors are Dirichlet, the parameter posteriors also become Dirichlet and the evidence integral factors into a product of easy Dirichlet integrals. So the intractability of the evidence integral for HMMs stems from the fact that both the states and parameters are hidden. Stolcke and Omohundro used a Viterbi-like algorithm to determine a single most likely sequence of hidden states, and treated this sequence as if it had been observed. They could then do the evidence integrals easily. By iterating between these two steps they incrementally searched over model structures, merging or splitting states based on comparisons of this (approximate) evidence. Their approach, which trades off integrating over hidden variables for integrating over parameters, attained impressive results recovering some simple finite state grammars.

The fourth approach to approximate Bayesian integration is known both as *ensemble learning* and the *variational Bayesian* method. The basic idea is to simultaneously approximate the distribution over both hidden states and parameters with a simpler distribution, usually by assuming the hidden states and parameters are independent. More specifically, the evidence can be lower bounded by applying Jensen's inequality twice:

$$\begin{aligned}
\log P(\mathcal{D}|\mathcal{M}) &= \log\int d\theta\ P(\mathcal{D},\theta|\mathcal{M}) && (62)\\
&\geq \int d\theta\ Q(\theta)\log\frac{P(\mathcal{D},\theta|\mathcal{M})}{Q(\theta)} && (63)\\
&= \int d\theta\ Q(\theta)\left[\log P(\mathcal{D}|\ \theta,\mathcal{M}) + \log\frac{P(\theta|\mathcal{M})}{Q(\theta)}\right] && (64)\\
&\geq \int d\theta\ Q(\theta)\left[\sum_S\ Q(S)\log\frac{P(S,\mathcal{D}|\ \theta,\mathcal{M})}{Q(S)} + \log\frac{P(\theta|\mathcal{M})}{Q(\theta)}\right] && (65)\\
&\equiv \mathcal{F}(Q(\theta),Q(S)) && (66)
\end{aligned}$$

The variational Bayesian approach iteratively maximises $\mathcal{F}$ as a functional of the two free distributions, $Q(S)$ and $Q(\theta)$. From (65) we can see that this maximisation is equivalent to minimising the KL divergence between $Q(S)\ Q(\theta)$ and the joint posterior over hidden states and parameters $P(S,\theta|\mathcal{D},\mathcal{M})$.

This approach was first proposed for one-hidden layer neural networks (which have no hidden state) by Hinton and van Camp (1993) using the restriction that $Q(\theta)$ is Gaussian [26]. It has since been applied to

[14] In fact, this is never the case for unconstrained HMMs which suffer from an identifiability problem, i.e. the identity of the states can be permuted with no effect on the likelihood. Because of this, the posterior for HMMs converges to a mixture of Gaussians.

various other models with hidden states and no restrictions on $Q(\theta)$ and $Q(S)$ other than the assumption that they factorise in some way [58, 7, 2, 19]. With only these factorisation assumptions, free-form optimisation with respect to the distributions $Q(\theta)$ and $Q(S)$ is done using calculus of variations, and often results in a modified EM-like algorithm.

MacKay (1997) first presented a variational Bayesian approach to learning in HMMs [39]. By assuming that the parameter prior was Dirichlet and approximating the posterior to have independent parameters and hidden states, he showed that the optimal $Q(\theta)$ was a Dirichlet distribution. Furthermore, he showed that the optimal $Q(S)$ could be obtained by applying the forward–backward algorithm to an HMM with pseudoparameters given by $\theta^* = \exp\{\int d\theta Q(\theta) \log \theta\}$, which can be evaluated for Dirichlet distributions. Thus the whole variational Bayesian algorithm can be implemented as a simple modification of the Baum-Welch algorithm. The variational Bayesian method contains as special cases both the MAP approach and a simple form of the Stolke and Omohundro approach. While very promising, especially given that it has been used successfully for non-trivial model structure learning in other models [19], its potential has not been fully explored for HMMs and their extensions.

8 Conclusion

In this paper we have reviewed hidden Markov models in the context of recent advances in the understanding of Bayesian networks. We have shown how HMMs are a kind of Bayesian network, and as such, the algorithms for learning and inference in HMMs can be derived from more general algorithms for Bayesian networks. It is possible to invent many generalisations of the HMM—such as factorial HMMs, tree-structured HMMs, and switching state-space models—which, by using richer hidden representations, can model more interesting temporal relationships than HMMs. However having richer hidden state representations invariably leads to computational intractability in the algorithms for inferring the hidden state from observations. Monte Carlo methods, such as Gibbs sampling, and variational methods are two ways of handling this intractability.

Finally, we discussed avoiding overfitting and learning the model structure. We presented several approaches, including one which makes a variational approximation to full Bayesian integration.

Acknowledgements

The author would like to thank Geoffrey E. Hinton, Michael I. Jordan, David J.C. MacKay, and Lawrence K. Saul who were collaborators on some of the work reviewed in this paper.

References

[1] B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Prentice-Hall, Englewood Cliffs, NJ, 1979.

[2] H. Attias. Inferring parameters and structure of latent variable models by variational Bayes. In *Proc. 15th Conf. on Uncertainty in Artificial Intelligence*, 1999.

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

[4] L.E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41:164–171, 1970.

[5] 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.

[6] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *J. Royal Stat. Soc. B*, 36:192–326, 1974.

[7] C.M. Bishop. Variational PCA. In *Proc. Ninth Int. Conf. on Artificial Neural Networks. ICANN*, 1999.

[8] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth International Group, Belmont, CA, 1984.

[9] T. W. Cacciatore and S. J. Nowlan. Mixtures of controllers for jump linear and non-linear plants. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 719–726. Morgan Kaufmann Publishers, San Francisco, CA, 1994.

[10] C. K. Carter and R. Kohn. Markov chain Monte Carlo in conditionally Gaussian state space models. *Australian Graduate School of Management, University of New South Wales*, 1996.

[11] T. Dean and K. Kanazawa. A model for reasoning about persitence and causation. *Computational Intelligence*, 5(3):142–150, 1989.

[12] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statistical Society Series B*, 39:1–38, 1977.

[13] V. Digalakis, J. R. Rohlicek, and M. Ostendorf. ML estimation of a Stochastic Linear System with the EM Algorithm and its Application to Speech Recognition. *IEEE Transactions on Speech and Audio Processing*, 1(4):431–442, 1993.

[14] B. J. Frey and N. Jojic. Transformed component analysis: Joint estimation of spatial transformations and image components. In *Proceedings of the IEEE International Conference on Computer Vision 1999, Kerkyra, Greece*, Los Alamitos, CA, 1999. IEEE Computer Society Press.

[15] J. H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19:1–141, 1991.

[16] D. Geiger and D. Heckerman. A characterization of the Dirichlet distribution through global and local independence. Technical Report MSR-TR-94-16, Microsoft Research, 1994 (revised 1996).

[17] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[18] Z. Ghahramani. Factorial learning and the *EM* algorithm. In G. Tesauro, D.S. Touretzky, and T.K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 617–624. MIT Press, Cambridge, MA, 1995.

[19] Z. Ghahramani and M. J. Beal. Variational inference for Bayesian mixtures of factor analysers. In *Adv. Neur. Inf. Proc. Sys. 12.* MIT Press, 2000.

[20] Z. Ghahramani and G. E. Hinton. Parameter estimation for linear dynamical systems. Technical Report CRG-TR-96-2 [http://www.gatsby.ucl.ac.uk/∼zoubin/papers/tr-96-2.ps.gz], Department of Computer Science, University of Toronto, 1996.

[21] Z. Ghahramani and G. E. Hinton. Variational learning for switching state space models. *Neural Computation*, 12:963–996, 2000.

[22] Z. Ghahramani and M. I. Jordan. Factorial hidden Markov models. *Machine Learning*, 29:245–273, 1997.

[23] G.C. Goodwin and K.S. Sin. *Adaptive filtering prediction and control.* Prentice-Hall, 1984.

[24] D. Heckerman. A tutorial on learning with Bayesian networks. Technical Report MSR-TR-95-06 [ftp://ftp.research.microsoft.com/pub/tr/TR-95-06.PS] , Microsoft Research, 1996.

[25] G. E. Hinton and R. S. Zemel. Autoencoders, minimum description length, and Helmholtz free energy. In J.D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6.* Morgan Kaufmann Publishers, San Francisco, CA, 1994.

[26] G.E. Hinton and D. van Camp. Keeping neural networks simple by minimizing the description length of the weights. In *Sixth ACM Conference on Computational Learning Theory, Santa Cruz*, 1993.

[27] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixture of local experts. *Neural Computation*, 3:79–87, 1991.

[28] F. V. Jensen. *Introduction to Bayesian Networks.* Springer-Verlag, New York, 1996.

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

[30] M. I. Jordan, Z. Ghahramani, and L. K. Saul. Hidden Markov decision trees. In M.C. Mozer, M.I Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9.* MIT Press, Cambridge, MA, 1997.

[31] M. I. Jordan and R.A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214, 1994.

[32] B. H. Juang and L. R. Rabiner. Hidden Markov models for speech recognition. *Technometrics*, 33:251–272, 1991.

[33] R. E. Kalman and R. S. Bucy. New results in linear filtering and prediction. *Journal of Basic Engineering (ASME)*, 83D:95–108, 1961.

[34] K. Kanazawa, D. Koller, and S. J. Russell. Stochastic simulation algorithms for dynamic probabilistic networks. In P. Besnard and S. Hanks, editors, *Uncertainty in Artificial Intelligence. Proceedings of the Eleventh Conference.*, pages 346–351. Morgan Kaufmann Publishers, San Francisco, CA, 1995.

[35] J. H. Kim and J. Peal. A computational model for causal and diagnostic reasoning in inference systems. In *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, pages 190–193. 1983.

[36] A. Krogh, M. Brown, I. S. Mian, K. Sjölander, and D. Haussler. Hidden Markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235:1501–1531, 1994.

[37] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *J. Royal Statistical Society B*, pages 157–224, 1988.

[38] L. Ljung and T. Söderström. *Theory and Practice of Recursive Identification.* MIT Press, Cambridge, MA, 1983.

[39] D.J.C. MacKay. Ensemble learning for hidden Markov models. Technical report, Cavendish Laboratory, University of Cambridge, 1997.

[40] M. Meila and M. I. Jordan. Learning fine motion by Markov mixtures of experts. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8.* MIT Press, 1996.

[41] R. M. Neal. Probabilistic inference using Markov chain monte carlo methods. Technical Report CRG-TR-93-1, 1993.

[42] R. M. Neal and G. E. Hinton. A new view of the EM algorithm that justifies incremental and other variants. Technical report, Department of Computer Science, University of Toronto, 1993.

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

[44] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Mateo, CA, 1988.

[45] L. R. Rabiner and B. H. Juang. An Introduction to hidden Markov models. *IEEE Acoustics, Speech & Signal Processing Magazine*, 3:4–16, 1986.

[46] H. E. Rauch. Solutions to the linear smoothing problem. *IEEE Transactions on Automatic Control*, 8:371–372, 1963.

[47] C. P. Robert, G. Celeux, and J. Diebolt. Bayesian estimation of hidden Markov chains: a stochastic implementation. *Statist. Prob. Letters*, 16:77–83, 1993.

[48] S. T. Roweis and Z. Ghahramani. A unifying review of linear Gaussian models. *Neural Computation*, 11(2):305–345, 1999.

[49] L. K. Saul and M. I. Jordan. Mixed memory Markov models. In D. Madigan and P. Smyth, editors, *Proceedings of the 1997 Conference on Artificial Intelligence and Statistics*. Ft. Lauderdale, FL, 1997.

[50] L. K. Saul and M. Rahim. Modeling acoustic correlations by factor analysis. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10, pages 749–755. The MIT Press, 1998.

[51] L.K. Saul and M. I. Jordan. Exploiting tractable substructures in Intractable networks. In D.S. Touretzky, M.C. Mozer, and M.E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*. MIT Press, 1996.

[52] R. H. Shumway and D. S. Stoffer. An approach to time series smoothing and forecasting using the EM algorithm. *J. Time Series Analysis*, 3(4):253–264, 1982.

[53] P. Smyth. Hidden Markov models for fault detection in dynamic systems. *Pattern Recognition*, 27(1):149–164, 1994.

[54] P. Smyth, D. Heckerman, and M. I. Jordan. Probabilistic independence networks for hidden Markov probability models. *Neural Computation*, 9:227–269, 1997.

[55] A. Stolcke and S. Omohundro. Hidden Markov model induction by Bayesian model merging. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems 5*, pages 11–18. Morgan Kaufmann, San Francisco, CA, 1993.

[56] M. A. Tanner and W. H. Wong. The calculation of posterior distributions by data augmentation (with discussion). *Journal of the American Statistical Association*, 82:528–550, 1987.

[57] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Trans. Informat. Theory*, IT-13:260–269, 1967.

[58] S. Waterhouse, D.J.C. Mackay, and T. Robinson. Bayesian methods for mixtures of experts. In *Adv. Neur. Inf. Proc. Sys. 7*. MIT Press, 1995.

[59] R. S. Zemel. *A minimum description length framework for unsupervised learning*. Ph.D. Thesis, Dept. of Computer Science, University of Toronto, Toronto, Canada, 1993.