Technical Report No. 2005-06, Department of Computer Science and Engineering, University at Buffalo, SUNY

Tree of Latent Mixtures for Bayesian Modelling and Classification of High Dimensional Data

Hagai T. Attias[*]
htattias@goldenmetallic.com
Golden Metallic, Inc., P.O. Box 475608
San Francisco, CA 94147, USA

Matthew J. Beal[†]
mbeal@cse.buffalo.edu
Dept. of Computer Science and Engineering
State University of New York at Buffalo
Buffalo, NY 14260-2000, USA

March 2004 (updated January, 2005)

Abstract

Many domains of interest to machine learning, such as audio and video, computational biology, climate modelling, and quantitative finance, involve very high dimensional data. Such data are often characterized by statistical structure that includes correlations on multiple scales. Attempts to model those high dimensional data raise problems that are much less significant in low dimensions. This paper presents a novel approach to modelling and classification in high dimensions. The approach is based on a graphical model with a tree structure, which performs density estimation on multiple spatial scales. Exact inference in this model is computationally intractable; two algorithms for approximate inference using variational techniques are derived and analyzed. We discuss learning and classification using these algorithms, and demonstrate their performance on a handwritten digit recognition task.

[*]http://www.goldenmetallic.com
[†]http://www.cse.buffalo.edu/faculty/mbeal

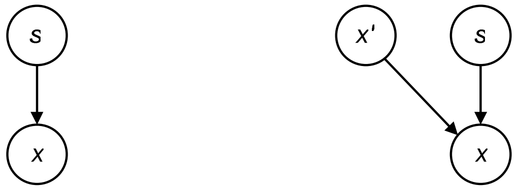

Figure 1: Left: a mixture model. x denotes the data vector and s the component label. Right: a mixture of factor analyzers (MFA) model. x' denotes the hidden factors.

1 Introduction

Many domains of interest to machine learning, such as audio and video, computational biology, climate modelling, and quantitative finance, involve very high dimensional data. Such data are often characterized by a rich statistical structure, including long range spatial and temporal correlations. These correlation make the graphical model builder's job quite challenging. Consider, for instance, fitting a Gaussian mixture model (Fig. 1 left) to K-dimensional data. Using diagonal precision matrices would require just K parameters per matrix, but could lead to highly inaccurate results for strongly correlated data. Using general precision matrices could in principle lead to an accurate model, but at the price of escalating the number of parameters to $K(K+1)/2$ per matrix. As K increases, the $\mathcal{O}(K^2)$ parameters could sharply increase the required computational resources, slow down the learning algorithm, and complicate the likelihood surface, increasing the number of local maxima and the algorithm's chance of getting stuck in one of them.

Many existing techniques attempt to tackle this problem [13, 11]. One group of techniques focus on controlling the number of model parameters. This may be achieved by constraining the precision matrices by e.g., tying parameters across components or applying symmetry considerations. A more sophisticated technique uses a factor analysis model with an appropriately chosen number of factors in each component (Fig. 1 right). Using L factors reduces the number of parameters to $K(L+1)$ per component. However, such techniques do not take into account the actual correlation structure in the data, which may lead to inaccurate modelling and unsatisfying performance.

Another group of techniques focus on controlling the number of data dimensions. This may be done by a variety of methods for feature extraction, projection, and dimensionality reduction. This pre-processing results in lower dimensional data, on which modelling is performed. However, the success of such methods strongly relies on accurate prior knowledge about which features are relevant for a given task. Applying them in the absence of such knowledge may again lead to inaccurate modelling.

This paper presents a novel approach to modelling and classification in high dimensions. The approach is based on a graphical model which performs density estimation on multiple scales. The model is termed *tree of latent mixtures* (TLM). All the nodes in the tree but the leafs are hidden, and each node consists of a mixture distribution (see Fig. 2). The TLM has a generative interpretation that is simple and intuitive: the variable at a given node draws its mean from the parent node, which is associated with a coarser scale; and the distribution about the mean is described by a mixture model. That variable, in turn, controls the mean of the children nodes, which are associated with a finer scale. Hence, TLM models correlations on multiple scales, where short range correlations are modelled by nodes lower in the tree, and progressively longer range correlations are modelled by higher nodes.

The TLM model belongs to the class of hybrid graphical models, which contain both discrete and continuous variables. Like many models in that class [10], exact inference in the TLM is computationally intractable. We present two algorithms for approximate inference, using variational techniques [6], and discuss learning and classification using these algorithms. Performance is demonstrated on a handwritten digit recognition task.

2 Model

Here we define the TLM model and discuss its different interpretations.

2.1 Mathematical Definition

The TLM model is defined as follows. Let $\mathcal{M}$ denote a tree structure with M nodes. Let $m = 1 : M$ index the nodes, where the root node is labelled by M. For node m, let πm denote its parent node, and let ζ_m denote the set of its child nodes. The number of children of node m is denoted by $C_m =\mid \zeta_m \mid$.

Each node in the tree is now expanded into a mixture model that contains two nodes, denoted by s_m and x_m for model m. Node s_m is termed the *component label*, and is a discrete variable with S_m possible values, $s_m = 1 : S_m$. Node x_m is termed the *feature vector*, and is a continuous vector of dimension K_m. (Note that we use the term feature just to refer to the continuous nodes; no actual feature extraction is involved.) We will refer to the nodes of the original tree graph as *models*, and reserve *nodes* for the label and feature variables.

Next we define the probabilistic dependencies of the nodes, shown graphically in Fig. 2. The component label s_m of model m depends on the label $s_{\pi m}$ of the parent model πm, via a probability table $w_{mss'}$,

$$p(s_m = s \mid s_{\pi m} = s') = w_{mss'} \ . \tag{1}$$

Hence, the discrete nodes $s_{1:M}$ themselves form a tree, whose structure is identical to the original tree structure $\mathcal{M}$. The continuous nodes $x_{1:M}$, however, do not form a

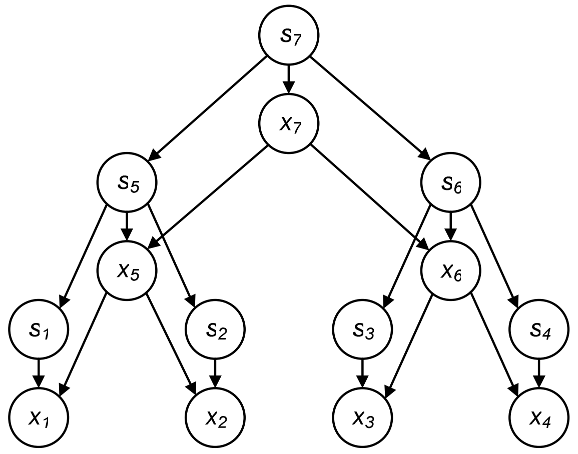

Figure 2: The TLM model. For model m, s_m is the component label and x_m is the feature vector. The models form a tree, which here is symmetric with $C_m = 2$ children for each model. During learning, the leaf nodes $x1, x2, x3, x4$ are visible and all other nodes are hidden.

tree, as each of them has two parents, one discrete and one continuous. The feature vector x_m of model m depends on the component label s_m of that model, and on the feature $x_{\pi m}$ of the parent model. The dependence draws on the mixture of factor analyzers (MFA) model (Fig. 1 right), where x_m is a linear function of $x_{\pi m}$ plus Gaussian noise, with the linear coefficients A_{ms}, a_{ms} and the noise precisions B_{ms} depend on s_m,

$$\begin{aligned} & p(x_m = x \mid s_m = s, x_{\pi m} = x') \\ & = \mathcal{N}(x \mid A_{ms}x' + a_{ms}, B_{ms}) \ . \end{aligned} \tag{2}$$

Like in MFA, the precision matrices B_{ms} are diagonal.[1]

The full joint distribution of the TLM is given by a product over models,

$$p(x_{1:M}, s_{1:M} \mid \theta) = \prod_{m=1}^{M} p(x_m \mid s_m, x_{\pi m}) p(s_m \mid s_{\pi m}) \tag{3}$$

where for the root node we set $s_{\pi M} = 1$ and $x_{\pi M} = 0$, and

$$\theta = \{A_{ms}, a_{ms}, B_{ms}, w_{mss'}\} \tag{4}$$

denotes the model parameters. To help with overfitting protection and model selection, we impose standard (conjugate, see [1]) independent priors over the parameters: Gaussian over A_{ms}, a_{ms}, Wishart over the precisions B_{ms}, and Dirichlet over the label parameters $w_{mss'}$,

$$\begin{aligned} p(A_{ms}) &= \mathcal{N}(A_{ms} \mid 0, \epsilon) \ , \quad & p(a_{ms}) &= \mathcal{N}(a_{ms} \mid 0, \epsilon) \ , \\ p(B_{ms}) &= \mathcal{W}(B_{ms} \mid \nu, \Phi) \ , \quad & p(w_{mss'}) &= \mathcal{D}(w_{mss'} \mid \alpha) \ . \end{aligned}$$

[1]Notation: the Gaussian distribution over a vector x with mean a and precision matrix B is $\mathcal{N}(x \mid a, B) = \mid B/2\pi \mid^{1/2} \exp[-(x-a)^T B(x-a)/2]$.

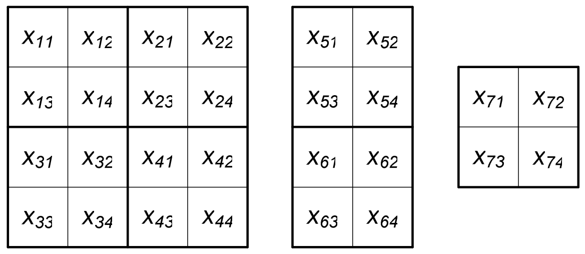
x_{11} x_{12} x_{21} x_{22}
x_{13} x_{14} x_{23} x_{24}
x_{31} x_{32} x_{41} x_{42}
x_{33} x_{34} x_{43} x_{44}
x_{51} x_{52}
x_{53} x_{54}
x_{61} x_{62}
x_{63} x_{64}
x_{71} x_{72}
x_{73} x_{74}

Figure 3: The feature vectors of the TLM of Fig. 2. The data are 16-dimensional and are organized in a spatial 4×4 array. The feature vectors at the leaves x_1, x_2, x_3, x_4 correspond to data variables. The features x_5, x_6, x_7 are hidden. In this example, all features have dimensionality $K_m = 4$.

Let $V \subseteq \{1 : M\}$ denote the models at the leaves, and let $H = \{1 : M\} \setminus V$ denote the rest of the models. Only the feature vectors of the leaf models, denoted by $x_V = \{x_m, m \in V\}$, correspond to observed data variables, and are therefore visible during learning. The rest of the feature vectors, denoted by $x_H = \{x_m, m \in H\}$, as well as all the component labels $s_{1:M}$, are hidden.

To construct a TLM model for a given dataset, one divides the data variables into subsets, and associates a feature vector $x_m \in V$ with each subset. The dimensionality K_m of the feature vectors is the number of variables in the corresponding subset. The subsets may or may not overlap; overlap could help reduce edge effects. Fig. 3 shows an example where 16-dimensional data, organized in a spatial 4×4 array, are modelled by the tree in Fig. 2. The data variables are divided into 4 non-overlapping subsets $V = \{1, 2, 3, 4\}$ with 4 variables each. Hence, the visible feature vectors x_m, $m \in V$ have dimension $K_m = 4$. The hidden features x_m, $m \in H$, where $H = \{5, 6, 7\}$, in this example also have $K_m = 4$. In principle, one may fit to a given dataset a tree of an arbitrary structure $\mathcal{M}$, not necessarily symmetric, and arbitrary parameters K_m and S_m. Estimating them from data is discussed briefly in Section 8.

2.2 Interpretation

Any probabilistic DAG models the process that generates the observed data, and has therefore a generative interpretation. To generate an observed data point from the TLM, start at the root and select a value $s \in \{1 : S_M\}$ for the label s_M with probability $p(s_M = s)$. Then generate the feature via $x_M = u_M$, where u_M is sampled from the Gaussian $p(x_M \mid s_M = s)$ which has mean a_{ms} and precision B_{Ms}. Next, proceed to the children models $m \in \zeta_M$. For each child, select a value s for its label s_m with probabilities $p(s_m = s \mid s_{\pi m})$. Then generate its feature via $x_m = A_{ms} x_{\pi m} + u_m$, where u_m is sampled from the Gaussian $p(x_m \mid s_m, x_{\pi m})$ which has mean a_{ms} and precision B_{ms}. Repeat until reaching the leaves.

The TLM may also be viewed as performing density estimation at multiple spatial scales, where the scale is finest at the leaves and coarsest at the root. Each feature x_m is described by a mixture distribution where the mean of each component, apart from a constant, is proportional to the parent $x_{\pi M}$. Hence, in effect, at a given spatial scale (or level in the tree), the mean is determined by a node at a coarser scale (the parent node, one level up), whereas the nodes at the given scale add the finer statistical details.

There is another attractive interpretation of the TLM, as a bottom-up process of successive clustering and dimensionality reduction. We will discuss it in Section 5 and exploit it for initialization.

3 Inference

Bayesian classification with TLM is performed by training a separate TLM model on data from each class, using an expectation maximization (EM) algorithm derived for the model. A new unlabelled data point is then classified by comparing its likelihood under the models for the different classes, combined with the class prior, as instructed by Bayes' rule.

Both the performing the E-step of EM and computing the data likelihood require inference, i.e., computing the posterior distribution over the hidden nodes x_H and $s_{1:M}$, conditioned on the observed data x_V,

$$p(x_H, s_{1:M} \mid x_V) = \frac{p(x_{1:M}, s_{1:M})}{p(x_V)} \; . \qquad (5)$$

However, this posterior is computationally intractable, since computing the normalization constant $p(x_V)$ requires summing over all possible state configurations $(s_1, ..., s_M)$ of the labels, whose number $\prod_{m=1}^{M} S_m = e^{\alpha M}$ is exponential in the number of models M ($\alpha = \langle \log S_m \rangle$ is the average log-number of states).

3.1 Variational Inference

In the following, we present two techniques for approximate inference in the TLM model. Both techniques are based on variational methods [6], which we now review briefly. In variational inference, one approximates the exact posterior $p(x_H, s_{1:M} \mid x_V)$ by another distribution $q(x_H, s_{1:M} \mid x_V)$, termed the *variational posterior*. Unlike the exact posterior, q is chosen to be tractable. This is usually achieved by giving it a factorized structure, i.e., grouping hidden variables into separate sets which are mutually independent given the data; in the exact posterior, those sets are correlated. $q(\cdot)$ is typically given in a parametric form, which either has to be specified in advance, or (as is the case here) emerges once the structure has been specified. Its parameters are then optimized to minimize its Kullback-Leibler (KL) distance $KL[q \mid\mid p]$ to the exact posterior p. Interestingly, the optimization requires knowledge of the exact posterior only within normalization. The optimization is performed by an iterative loop nested at the E-step of each EM iteration, and is guaranteed to converge since the KL distance is lower-bounded by zero.

Learning (M-step) in TLM needs the following posterior moments (a.k.a. sufficient statistics). These are the feature conditional means ρ_{ms} and child-parent correlations $\Lambda_{ms,m's'}$, $m' = \pi m$, as well as the label probability γ_{ms} and joint child-parent probabilities $\eta_{ms,m's'}$. They are defined via marginals of the variational posterior $q(x_H, s_{1:M} \mid x_V)$ by

$$\begin{aligned}
\rho_{ms} &= \int dx_m \; q(x_m \mid s_m = s) \; x_m \; , \\
\Lambda_{ms,m's'} &= \int dx_m dx_{m'} \\
&\quad q(x_m, x_{m'} \mid s_m = s, s_{m'} = s') \; x_m x_{m'}^T \; , \\
\gamma_{ms} &= q(s_m = s \mid x_V) \; , \\
\eta_{ms,m's'} &= q(s_m = s, s_{m'} = s' \mid x_V) \; . \qquad (6)
\end{aligned}$$

Those posterior moments are computed below by each inference technique separately.

At the M-step of each EM iteration, one considers the averaged complete data likelihood $E_q \log p(x_{1:M}, s_{1:M} \mid \theta)$, where E_q averages w.r.t. the variational posterior computed at the E-step. The learning rule is derived, as is standard, by maximizing this quantities w.r.t. the mode parameters θ. Once a model has been learned, the likelihood $\mathcal{L} = \log p(x_V)$ of a new data point x_V is approximated by the variational likelihood $\mathcal{F}$,

$$\mathcal{F} = E_q \left[\log p(x_{1:M}, s_{1:M}) - \log q(x_H, s_{1:M} \mid x_V)\right] \; . \qquad (7)$$

In fact, it can be shown that $\mathcal{F} = \mathcal{L} - KL[q \mid\mid p] \leq \mathcal{L}$ (the argument p refers to the exact posterior), hence the variational E-step that minimizes the KL distance also maximizes the variational likelihood.

3.2 Technique I: Tree of Factorized Features

Generally, the posterior may be written as the posterior over labels and over features conditioned on those labels,

$$q(x_H, s_{1:M} \mid x_V) = q(x_H \mid s_H, x_V) q(s_{1:M} \mid x_V) \; . \qquad (8)$$

We now require the features to be conditionally independent of each other,

$$q(x_H \mid s_H, x_V) = \prod_{m \in H} q_m(x_m \mid s_m) \; . \qquad (9)$$

This is the only restriction we impose on the structure of the posterior. Notice that this structure maintains three important properties of the exact posterior: (1) the labels are correlated, (2) the features depend on the labels, and (3) the features themselves are correlated since $q(x_H \mid$

$x_V) = \sum_{S_H} q(x_H \mid s_H, x_V) q(s_H \mid x_V) \neq \prod_{m \in H} q(x_m \mid x_V)$. However, the details of the different dependencies differ from the exact posterior. Similar ideas have been used in [2, 9] in the context of processing multiple speech/sound signals.

The functional form of the variational posterior falls out of free-form optimization of $\mathcal{F}$ under the structural restriction (9); no further assumptions are required. First, the label posterior has a tree structure, just like the prior (1), given by a product over potentials

$$q(s_{1:M} \mid x_V) = \frac{1}{Z} \prod_{m=1}^{M} \exp\left[\phi_m(s_m \mid s_{\pi m})\right] \qquad (10)$$

where Z is a normalization constant. Since the label posterior is a tree, any required statistics, including Z and the posterior moments γ_{ms} and $\eta_{ms,m's'}$ in (6), can be computed efficiently using any of the standard message passing algorithms [8]. We exploit this useful property for computing the feature posterior below.

The potentials ϕ_m are given by the expression

$$\begin{aligned} &\phi_m(s_m = s \mid s_{\pi m} = s') = \log w_{mss'} \qquad (11)\\ &+ \log p(x_m = \rho_{ms} \mid s_m = s, x_{\pi m} = \rho_{\pi m, s'})\\ &- \frac{1}{2} \mathrm{Tr}\left(\Gamma_{ms}^{-1} + A_{ms} \Gamma_{\pi m, s'}^{-1} A_{ms}^T\right) + \frac{1}{2} \log \mid \Gamma_{ms} \mid , \end{aligned}$$

which basically shows how inferring feature x_m modifies the prior $w_{mss'}$ to yield the posterior. Here, ρ_{ms} and Γ_{ms} denote the mean and precision, respectively, of the feature posterior $q(x_m \mid s_m = s, x_V)$ in (9), to be computed below. For the leaf nodes $m \in V$ we substitute $\rho_{ms} = x_m$, $\Gamma_{ms}^{-1} = \log \mid \Gamma_{ms} \mid = 0$, and for the root node $m = M$, $\rho_{\pi m, s} = \Gamma_{\pi m, s}^{-1} = 0$. In addition, $p(x_m = ...)$ in the second line refers to p in (2) with the appropriate substitutions.

Next, the feature posteriors turn out to be Gaussian,

$$q(x_m \mid s_m = s) = \mathcal{N}(x_m \mid \rho_{ms}, \Gamma_{ms}) \qquad (12)$$

whose the precision matrix is given by

$$\Gamma_{ms} = B_{ms} + \sum_{m' \in \zeta_m} E_{m's'|ms} A_{m's'}^T B_{m's'} A_{m's'} \ . \qquad (13)$$

$E_{m's'|ms}$ denotes posterior averaging over $s_{m'}$ conditioned on $s_m = s$, e.g.,

$$E_{m's'|ms} A_{m's'} = \sum_{s'} q(s_{m'} = s' \mid s_m = s, x_V) A_{m's'} \ ,$$

where $q(s_{m'} = s' \mid s_m = s, x_V) = \eta_{m's',ms}/\gamma_{ms}$.

The means of (12) are given by a linear equation that expresses ρ_{ms} in terms of the parent and children of node m,

$$\begin{aligned} \Gamma_{ms} \rho_{ms} &= B_{ms}\left(A_{ms} E_{\pi m, s'|ms} \rho_{\pi m, s'} + a_{ms}\right) \qquad (14)\\ &+ \sum_{m' \in \zeta_m} E_{m's'|ms} A_{m's'}^T B_{m's'} \left(\rho_{m's'} - a_{m's'}\right) \ . \end{aligned}$$

Direct solution of this linear system may be inefficient, since its dimension $\sum_m K_m S_m$ may be quite large. However, most of the coefficients are zero since node m is coupled only to its parent and children, hence sparse matrix techniques can solve it efficiently. Instead, we implemented an iterative solution which, starting with the initialization described in Section 5, makes repeated passes through the tree and updates the means using (14). For the experiments described below, convergence was almost always achieved in $\mathcal{O}(10)$ passes.

The conditional feature correlations $\Lambda_{ms,m's'}$ in (6) can now be computed via

$$\Lambda_{ms,m's'} = \rho_{ms} \rho_{m's'}^T + \delta_{mm'} \Gamma_{ms}^{-1} \qquad (15)$$

Finally, as usual with variational techniques, the equations for $q(s_{1:} \mid x_V)$ and $q(x_H \mid s_H, x_V)$ are mutually dependent and must be solved iteratively.

3.3 Technique II: Factorized Trees

Here, we require the features to be independent of the labels,

$$q(x_H, s_{1:M} \mid x_V) = q(x_H \mid x_V) q(s_{1:M} \mid x_V) \ . \qquad (16)$$

This is a stronger restriction than the one proposed in the previous section (9), and has the disadvantage that the posterior over the features is unimodal (above it was a mixture distribution, $q(x_H) = \sum_{s_H} q(x_H \mid s_H) q(s_H)$). In fact, it can be shown to be a single Gaussian, which is a fairly simplistic approximation to the exact posterior. However, it has the advantage of preserving direct correlations among the features, and it is also simpler to compute.

Just like above, the functional form of the variational posterior falls out of a free-form optimization of $\mathcal{F}$ under the restriction (9); no further assumptions are required. First, the label posterior again has a tree structure, given by a product over potentials

$$q(s_{1:M} \mid x_V) = \frac{1}{Z} \prod_{m=1}^{M} \exp\left[\psi_m(s_m \mid s_{\pi m})\right] \qquad (17)$$

where Z is a normalization constant. Any required statistics, including Z and the posterior moments γ_{ms} and $\eta_{ms,m's'}$ in (6), can be computed efficiently via message passing [8].

The potentials ψ_m are obtained via

$$\begin{aligned} &\psi(s_m = s \mid s_{\pi m} = s') = \log w_{mss'} \qquad (18)\\ &+ \log p(x_m = \rho_m \mid s_m = s, x_{\pi m} = \rho_{\pi m})\\ &- \frac{1}{2} \mathrm{Tr}\left(\Lambda_{m,m} + A_{ms} \Lambda_{\pi m, \pi m} A_{ms}^T - 2 \Lambda_{m, \pi m} A_{ms}^T\right) \ , \end{aligned}$$

where ρ_m and $\Lambda_{mm'}$ (6) are discussed below. For the leaf nodes $m \in V$ we substitute $\rho_m = x_m$, $\Lambda_{mm'} = 0$, and for the root node $m = M$, $\rho_{\pi m} = \Gamma_{\pi m, m'} = 0$. In addition,

$p(x_m = ...)$ in the second line refers to p in (2) with the appropriate substitutions.

Next, the feature posterior is shown to be a Gaussian tree, given by a product over potentials

$$q(x_H \mid x_V) = \frac{1}{Z'} \prod_{m \in H} \exp\left[\Psi_m(x_m \mid x_{\pi m})\right] \ , \tag{19}$$

where Z' is a normalization constant. The potentials Ψ_m are given by the quadratic expressions

$$\begin{aligned} &\Psi_m(x_m = x \mid x_{\pi m} = x') = x^T(E_{ms}B_{ms})x \\ &+x'^T(E_{ms}A_{ms}^T B_{ms} A_{ms})x' - 2x'^T(E_{ms}A_{ms}^T B_{ms})x \\ &-2(E_{ms}a_{ms}^T B_{ms})x + 2(E_{ms}a_{ms}^T B_{ms} A_{ms})x' \ , \end{aligned} \tag{20}$$

where E_{ms} denotes posterior averaging over s_m, e.g., $E_{ms}A_{ms} = \sum_s \gamma_{ms} A_{ms}$.

Finally, the conditional feature means and correlations in (6) in this case are not conditioned on the label, due to the variational factorization (16). Hence

$$\begin{aligned} \rho_{ms} &= \rho_m = \int dx_m \, q(x_m) \, x_m \ , \\ \Lambda_{ms,m's'} &= \Lambda_{mm'} = \int dx_m dx_{m'} q(x_m, x_{m'}) \, x_m x_{m'} \ , \end{aligned} \tag{21}$$

and both $\rho_m, \Lambda_{mm'}$ are computed via message passing, due to the tree structure of the feature posterior.

4 Learning

Given the posterior moments (6), the update rules (M-step) for the TLM model parameters θ (4) are straightforward to derive. We define, as is standard, the extended feature vector $\bar{x}_m^T = (x_m^T, 1)$ by appending 1, and the extended matrix $\bar{A}_{ms} = (A_{ms}, a_{ms})$ by appending the column a_{ms} to the right. We extend $\Lambda_{ms,m's'}$ of (6) such that $\bar{\Lambda}_{ms,m's'}$ is the correlation of $\bar{x}_m, \bar{x}_{m'}$, and $\bar{\Lambda}'_{ms,m's'}$ is the correlation of $x_m, \bar{x}_{m'}$. We also define the matrices

$$\begin{aligned} F_{ms} &= E_{\pi m, s' \mid ms} \bar{\Lambda}_{\pi m, s', \pi m, s'} \ , \\ F'_{ms} &= E_{\pi m, s' \mid ms} \bar{\Lambda}'_{ms, \pi m, s'} \ , \end{aligned} \tag{22}$$

where the average $E_{m's' \mid ms}$ is defined in (14).

The learning algorithm presented here runs in batch mode on an N-point dataset. Averaging over the data is denoted by $\langle \cdot \rangle$.

For $\bar{A}_{ms}$ we obtain

$$\bar{A}_{ms} = \langle F'_{ms} \rangle \left(\langle F_{ms} \rangle + \epsilon / N \right)^{-1} \ . \tag{23}$$

For B_{ms} we obtain

$$\begin{aligned} B_{ms}^{-1} &= (1 + \nu/N)^{-1} \big(\Lambda_{ms,ms} + \bar{A}_{ms} \langle F_{ms} \rangle \bar{A}_{ms}^T \\ &- 2\langle F'_{ms} \rangle A_{ms}^T + \Phi/N \big) \ . \end{aligned} \tag{24}$$

For $w_{mss'}$ we obtain

$$w_{mss'} = \frac{\langle \eta_{ms,\pi m,s'} \rangle + \lambda/N}{\langle \gamma_{\pi m,s'} \rangle + \lambda S_m / N} \ . \tag{25}$$

Note how the parameters of the priors ϵ, ν, Φ, and λ regularize the update rules by preventing ill-conditioning.

5 Initialization

As EM is not guaranteed to converge to the global maximum of the likelihood, using an effective initialization procedure is important. Here we discuss a heuristic that emerges from an interesting, though not rigorous, interpretation of what the TLM actually does. Taking a bottom-up view, the leaf models $m \in V$ perform clustering on their data x_m. The parent feature x_{π_m} then performs linear dimensionality reduction of the clustering errors from all its children, i.e., of the difference between each data point and the center of its assigned cluster. At the next level up, the models πm cluster the features $x_{\pi m}$, and their parent features reduce dimensionality of the errors of that clustering. The process continues until the root is reached.

Based in this view, we define the following procedure. Consider each leaf $m \in V$ in isolation, setting $A_{sm} = 0$, and run EM-GM (EM for a Gaussian mixture) to determining the cluster means and precision a_{ms}, B_{ms}. Set the weight $w_{mss'}$ to the responsibility (posterior probability) of cluster s for data x_m. Vector quantization (VQ) may be used to initialize EM-GM. Next, for each model $m \in V$ and data point x_m, define the clustering error $u_m = x_m - a_{m\hat{s}(m)}$, where $\hat{s}(m) = \arg\min_s \mid x_m - a_{ms} \mid$. Now, consider each parent m' of a leaf, and run EM-FA (EM for factor analysis) using $x_{m'}$ as factors and the error from all its children u_m, $\pi_m = m'$ as data. This will determine the A_{ms} (notice that $\hat{s}(m)$ must be used to label the u_m). Principal component analysis (PCA) may be used to initialize EM-FA.

Next, set the features $x_{m'}$ to their MAP values obtained by EM-FA. Regard those values as observed data, and perform the above procedure of EM-GM followed by EM-FA. Repeat until the root is reached.

This completes the initialization of all model parameters θ. However, for variational EM one must also initialize either the moments of the label posterior or the moments of the feature posterior (6). We did the latter, by setting γ_{ms} to the component responsibility in each model m, computed from EM-GM for that model, and approximating $\eta_{ms,m's'} = \gamma_{ms}\gamma_{m's'}$.

We point out that several variants of our initialization procedure exist. In one variant the tree is built and initialized section by section. One stops the procedure after only part of the tree has been initialized, and runs variational EM only on that part. After convergence, one proceeds upward with initialization, and possibly stops again to run variational EM before reaching the root.

6 Bayesian Classification

Here we consider the TLM in the context of supervised classification. In that task, a training set consisting of N pairs (x_V, c) of data and class label is given. These data are used to train a different TLM model for each class. Let θ_c denote the parameters learned for class c, and let

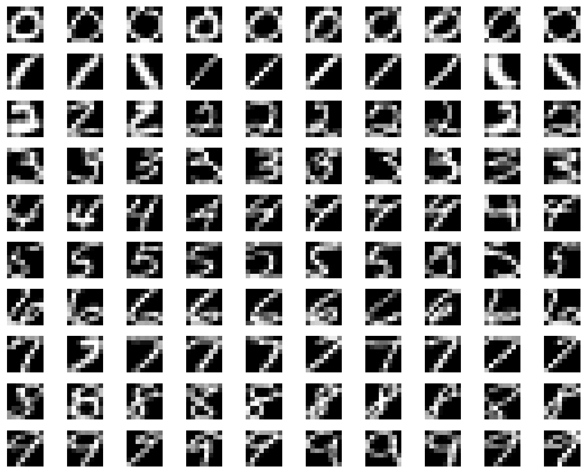

Figure 4: Handwritten digits from the Buffalo dataset.

$p(x_{1:M}, s_{1:M} \mid \theta_c)$ denote the TLM joint distribution (3) for that class. Let $p(c)$ denote the prior over classes. The exact Bayes classifier for a new data point x_V is given by $\hat{c}(x_V) = \arg\max_c p(x_V \mid \theta_c)p(c)$.

However, the marginal $p(x_V \mid \theta_c)$ is computationally intractable, and in the variational framework it is approximated via the variational likelihood (7). Let $\mathcal{F}_c$ denote the variational likelihood for the TLM of class c. Then $\mathcal{F}_c(x_V) \leq \log p(x_V \mid \theta_c)$, and the approximate Bayes classifier is given by

$$\hat{c}(x_V) = \arg\max_c \left[\mathcal{F}_c(x_V) + \log p(c)\right] \ . \tag{26}$$

Computing $\mathcal{F}_c$ can be shown to be remarkably simple in either of the variational techniques discussed above. Recall that the label posterior in either case was a tree, given as a product over potential normalized by Z (10,17), which in turn is computed by message passing. Denoting it by $Z_c(x_V)$ for the TLM of class c, we have

$$\mathcal{F}_c(x_V) = \log Z_c(x_V) \tag{27}$$

within a constant independent of x_V.

7 Experiments

We tested the TLM on the Buffalo post office dataset, which contains 1100 examples for each digits 0-9. Each digit is a gray-level 8×8 pixel array (see examples in Fig. 4). We used 5 random 800-digit batches for training, and a separate 200-digit batch for testing.

We considered 3 different TLM structures. The structures differed by the number of models M, the parameter values K_m, S_m, C_m, and the degree of overlap between the pixels associated with the data variables x_m, $m \in V$. In all cases we tied the parameters of A_{ms} across components, such that $A_{ms} = A_m$, since the available number of training examples may be too small to estimate all A_{ms} reliably (the NIST digits dataset should alleviate this problem, see below). Structure (1) included 2 levels with $M = 5$ models. Each of the 4 leaf nodes covered a 4×4 pixel array with no overlap. The leaf models created a 2×2 model array, and all had the root as parent, hence $C_m = 4$. We used $K_m = 16$ and $S_m = 6$ for all models. Structure (2) included 3 levels with $M = 14$ models. Each of the 9 leaf nodes again covered a 4×4 pixel array, but it overlapped its horizontal neighbors by a 4×2 and its vertical neighbors by 2×4 pixel array. The leaf models created a 3×3 model array. Next, the middle level had 4 models, each parenting a 2×2 model array of the leaf level, with an overlap as before. The middle level models created a 2×2 model array, and all had the root as parent. Hence, this structure also had $C_m = 4$. We used $K_m = 16$ for all models, and set $S_m = 2$ to keep the total number of parameters approximately equal to structure (1). Structure (3) included 3 levels with $M = 21$ models. Each of the 16 leaf models covered a 2×2 pixel array with no overlap. The leaf models created a 4×4 model array, and the tree was constructed upward with $C_m = 4$ similarly to structure (1). We used $K_m = 4$ for all models with $S_m = 4$, leading to about half the number of parameters of structures (1),(2).

Table 1: Misclassification rate for three different TLM structures using two variational inference techniques. MOG denotes a Gaussian mixture benchmark.

STRUCTURE	TECHNIQUE I	TECHNIQUE II
(1)	1.8%	2.2%
(2)	1.6%	1.9%
(3)	2.2%	2.7%
MOG	2.5%	2.5%

For each structure we ran variational EM with both inference techniques. The results are shown in Table 1. As benchmark, denoted MOG, we used standard Gaussian mixture model with 30 components. All TLM structures, except (3) with inference technique II, outperformed the benchmark. Structure (2) with technique I outperformed the rest.

While our experiments are not yet exhaustive, they seem to indicate that (1) overlap at the leaf level may enhance performance, possibly by reducing edge effects; and (2) technique I may be superior, possibly due to the multimodality of the feature posterior. We are currently performing experiments with additional TLM structures, and will report the results in the final version of this paper. We are also working on a second set of experiments using the NIST digits dataset, which contains $60,000$ training examples on a 20×20 pixel array.

8 Extensions

The work presented here may be extended in several interesting directions. We are currently pursuing new techniques for approximate inference in TLM. A

Rao-Blackwellized Monte Carlo approach [4, 7], which combines sampling the labels with exact inference on the features given the label samples, has given promising preliminary results. Suitable versions of other methods, including loopy belief propagation [14], expectation propagation [12], and the algorithm of [3] (originally designed for time series), may also prove effective.

An important direction we plan to pursue is learning the structure $\mathcal{M}$ of TLM, including the parameters C_m, K_m, S_m, from data. In classification, for instance, different classes may require models with different structures. Structure search, accelerated using our initialization method, and combined with scoring using a variational Bayesian technique [1, 5], could produce a powerful extension to the current TLM algorithms.

Another direction involves extending TLM into a dynamic graphical model, by constructing a Markov chain of TLMs. This graph would model complex time series on multiple spatiotemporal scales, and could produce a novel and effective forecasting tool.

References

[1] H. Attias. A variational Bayesian framework for graphical models. In *Advances in Neural Information Processing Systems 12*, pages 209–215. MIT Press, 2000.

[2] H. Attias. Source separation with a sensor array using graphical models and subband filtering. In *Advances in Neural Information Processing Systems 15*. MIT Press, 2003.

[3] X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence*, pages 33–42. Morgan Kaufmann, 1998.

[4] A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer, 2001.

[5] Z. Ghahramani and M. J. Beal. Propagation algorithms for variational Bayesian learning. In *Advances in Neural Information Processing Systems 13*, pages 507–513. MIT Press, 2001.

[6] M. I. Jordan, Z. Ghahramani, and T. S. Jaakkola. An introduction to variational methods for graphical models. In M.I. Jordan, editor, *Learning in Graphical Models*. MIT Press, 1998.

[7] T. Kristjansson, H. Attias, and J. R. Hershey. Stereo based 3d tracking and learning using EM and particle filtering. In *Proceedings of the 18th European Conference on Computer Vision*, page in press, 2004.

[8] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47:498–519, 2001.

[9] L. J. Lee, H. Attias, and L. Deng. A multimodal variational approach to learning and inference in switching state space models. In *Proceedings of the 2004 International Conference on Acoustics, Speech, and Signal Processing*, page in press, 2004.

[10] U. Lerner and R. Parr. Inference in hybrid networks: theoretical limits and practical algorithms. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence*, pages 310–318. Morgan Kaufmann, 2001.

[11] M. Meila and M. I. Jordan. Learning with mixtures of trees. *Journal of Machine Learning Research*, 1:1–48, 2000.

[12] T. Minka and J. Lafferty. Expectation-propagation for the generative aspect model. In *Proceedings of the 18th Annual Conference on Uncertainty in Artificial Intelligence*, pages 352–359. Morgan Kaufmann, 2002.

[13] A. Moore. Very fast EM-based mixture model clustering using multiresolution kd-trees. In *Advances in Neural Information Processing Systems 11*, pages 543–549. MIT Press, 1999.

[14] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Generalized belief propagation. In *Advances in Neural Information Processing Systems 13*, pages 689–695. MIT Press, 2001.