Dynamic Trees for Image Modelling

Nicholas J. Adams — *nicka@dai.ed.ac.uk*

Christopher K. I. Williams* — *c.k.i.williams@ed.ac.uk*

Institute for Adaptive and Neural Computation

Division of Informatics, University of Edinburgh

5 Forrest Hill, Edinburgh, UK

`http://anc.ed.ac.uk/`

13 November 2002

Abstract

This paper introduces a new class of image model which we call *dynamic trees* or DTs. A *dynamic tree* model specifies a prior over structures of trees, each of which is a forest of one or more tree-structured belief networks (TSBN). In the literature standard tree-structured belief network models have been found to produce "blocky" segmentations when naturally occurring boundaries within an image did not coincide with those of the subtrees in the rigid fixed structure of the network. *Dynamic trees* have a flexible architecture which allows the structure to vary to create configurations where the subtree and image boundaries align, and experimentation with the model has shown significant improvements.

For large models the number of tree configurations quickly becomes intractable to enumerate over, presenting a problem for exact inference. Techniques such as Gibbs sampling over trees and search using simulated annealing have been considered, but a variational approximation based upon mean field was found to work faster while still producing a good approximation to the true model probability distribution. We look briefly at this mean field approximation before deriving an EM-style update based upon mean field inference for learning the parameters of the *dynamic tree* model.

^{0}To whom correspondence should be addressed.

After development of algorithms for learning the *dynamic tree* model is applied to a database of images of outdoor scenes where all of its parameters are learned. DTs are seen to offer significant improvement in performance over the fixed-architecture TSBN and in a coding comparison the DT achieves 0.294 bits per pixel (bpp) compression compared to 0.378 bpp for lossless JPEG on images of 7 colours.

Keywords: Bayesian image modelling, belief networks, dynamic tree, variational inference, mean field, expectation-maximisation.

1 Introduction

Probabilistic modelling of images provides a useful and well grounded framework to conduct inference from images. There has been much interest in this area over recent years, and this has given rise to the development of a rich and varied suite of models and techniques, which are reviewed in Section 2.

We describe an image model called the *dynamic tree* which is developed from earlier work on Tree-Structured Belief Networks (TSBNs). Tree-structured belief networks provide a natural way of modelling images within a probabilistic framework. By this method a balanced tree-structured belief network is constructed with a single root node and the image is presented at the leaves. Inference can then be conducted by an efficient linear-time algorithm [27]. Fixed-structure TSBNs have been used by a number of authors as models of images; see e.g. the work of Bouman and Shapiro [5] (where the node variables are discrete) and Willsky and coauthors [25, 21, 14] (where the nodes contain real-valued Gaussian variables). TSBNs have an attractive multi-scale structure, but suffer from problems due to the fixed tree structure, which can lead to very *blocky* segmentations.

Consider for instance the four level binary tree of Figure 1(a). The image applied in the example is a 1-d image of a black bar on a white background, and initially all the other nodes in the tree are uninstantiated. The structure and size of the TSBN directly determines the

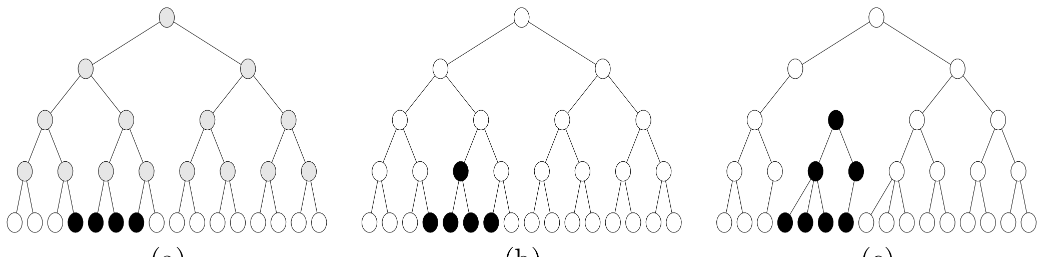


Figure 1: (a) “Balanced” tree with image applied (b) resulting segmentation and (c) an example *dynamic tree*.

images it can deal with. The binary tree shown handles 1-d images, but more commonly quad-trees are used to model 2-d images.

For the segmentation all the nodes in the network are updated to their most probable state given the model parameters and the image. Since the network is probabilistic in nature we can also ascertain measures of certainty about the state of each node and so have a measure of performance. Our example produces the tree of Figure 1(b).

The hierarchical structure of TSBNs naturally leads to coarser-scale representations of the image at successive levels. As well as providing a natural mechanism for all regions in the image to have some influence over each other and thus exert global consistency, there is also potential for using the segmentations given at higher levels in image coding applications.

However, some problems arise when the natural boundaries in the image do not coincide with those of sub-trees in the TSBN. This effect is illustrated by Figure 1(b), where the black bar spans two sub-trees with roots at the third level. The resulting segmented images exhibit an undesirable *blockiness* as a consequence. The aim of our work is to attempt to find models which produce good representations of natural images, overcoming the problems of blockiness found in fixed-structure TSBNs. One such strategy is to break away from the tree structure, and use belief networks with cross connections; see e.g. [5, 10]. However, this means losing the linear-time belief-propagation algorithms that can be used in trees [27] and using approximate inference algorithms.

TSBNs have many attractive properties and we believe that models based upon them, but which have a dynamically adjustable tree structure to enable their boundaries to better reflect those of the image, provide a very promising starting point. We have such models named *dynamic trees* (DTs) and one such tree produced for the toy image data is shown in the Figure 1(c).

Dynamic trees (DTs) are a generalisation of the fixed-architecture tree structured belief networks. Belief networks can be used in image analysis by grouping its nodes into visible units $\mathbf{X}_v$, to which the data is applied, and having hidden units $\mathbf{X}_h$, connected by some suitable configuration, to conduct a consistent inference. DTs set a prior $P(\mathbf{Z})$ over tree structures $\mathbf{Z}$ which allows each node to choose its parent so as to find one which gives a higher posterior probability $P(\mathbf{Z}, \mathbf{X}_h|\mathbf{X}_v)$ for a given image $\mathbf{X}_v$. This effectively produces forests of TSBNs.

Allowing this flexibility to chose their structure is not without cost. For useful *dynamic trees* the number of configurations $\mathbf{Z}$ is very large and typically we cannot tractably enumerate over all of them. However, while it is true that exact inference in DTs is NP-hard, we do retain a *clean* semantics based on the fact that we expect that each pixel should belong to one object.

This paper will explore the *dynamic tree* model from conception to a real world application of the model. After a survey of related work in Section 2, the *dynamic tree* model is introduced in Section 3. The number of configurations of the DT model is large and enumeration over all of them is usually intractable, and therefore we need to rely on other techniques. Sampling and search approaches tend to be computationally intensive and slow to converge. An alternative is to approximate the true posterior probability distribution with a simpler tractable one, and variational methods are proving very popular for this task. In Section 3.2 we derive the mean field approximation for the *dynamic tree* which is the simplest of the variational approximations. Learning algorithms for the DT are developed in Section 3.3.

Section 4 then trains the *dynamic tree* on a database of outdoor scenes. DTs are seen to offer significant improvement in performance over fixed-architecture TSBNs. A discussion in Section 5 concludes this work by summarising the key results obtained and outlining some promising avenues for future research.

2 Related Work

There is of course a vast literature on the subject of image analysis. Here we concentrate particularly on probabilistic formulations of the image analysis problem. A fuller discussion of these issues can be found in Chapter 2 of [1].

Within the probabilistic modelling framework, the most popular models are MRF and TSBN models. In the statistical image modelling community these two types of model are known as non-causal and causal MRF models respectively. They are undirected and directed graphical models [23]. Early work on probabilistic image modelling focussed on non-causal MRFs, see e.g. [17, 7]. These models typically have a "flat", non-hierarchical structure. They define a stationary process (thereby overcoming the problems of blockiness), but in general the inference problem in a MRF is NP-hard.

The alternative causal MRF formulation uses a directed graph, and the most commonly used form of these models is a TSBN, as described in Section 1. In contrast to flat MRFs, TSBNs provide a hierarchical multiscale model for an image and have efficient linear-time inference algorithms. However, as noted above, the fixed tree structure gives rise to a non-stationary image model and leads to problems of blockiness.

We also note that wavelet models are examples of hierarchical multiscale models for images. For example Crouse *et al* [9] have used a multiscale TSBN to model wavelet coefficients, and DeBonet and Viola [11] have used an interesting tree-structured network for image synthesis using non-Gaussian densities.

Our thesis is that the problem with TSBNs is not the tree structure *per se*, but the fact that it is fixed in advance. This suggests that the model should provide a distribution over tree structures, reminiscent of parse trees obtained with context-free grammars (CFGs), see e.g. [6]. Some previous work has looked at CFG models for image analysis, for example [8, 18].

Dynamic tree models are as examples of *dynamic* image analysis architectures, in contrast to *fixed* or static architectures. The distinction between the two is that dynamic models don't merely instantiate hidden variables in a fixed structure conditionally on the supplied data, but they seek also to find relationships between substructures of these variables and are able to modify these relationships on the fly.

von der Malsburg [30, 31] has discussed the Dynamic Link Architecture, whereby the network architecture changes dynamically in response to the input. This parallels the inference process in DT architectures, where posterior tree structures are effectively selected in accordance with how well they fit the image structure. We also note that Montanvert et al [26] have discussed irregular tree-structured networks, where the tree-structure is image dependent. Also, Geman and Geman [17] introduced line processes as an extension of the basic MRF approach. These line processes (which also form an MRF) occupy the boundaries between pixels. The line processes are dynamically combined as edge elements to describe the boundaries between regions in the image.

As well as having a dynamic architecture over a fixed number of units it can also be desirable to allow the number of units to vary. Hinton *et al* (1998) [19] adopt such a strategy with their "hierarchical community of experts", whereby the participation or non-participation of a unit in the model is determined by a gating variable. This model has a general directed acyclic graph (DAG) construction. In later work, Hinton *et al* [20] developed the "credibility network" architecture, where a single parent constraint (similar to the construction of *dynamic trees*) is used. The work on credibility networks is quite closely related to ours, although they do not use hidden state variables $\mathbf{X}_h$ and instead focus more on the presence or absence of nodes.

Although not an image model, the work of Geiger and Heckerman (1996) [16] on multinets is also relevant. As with the *dynamic tree* model, Geiger and Heckerman construct a number of belief networks conditional on a structure variable $\mathbf{Z}$. In their work multinets were used as a way of speeding up inference, as conditional on $\mathbf{Z}$ each network will typically be much simpler than the equivalent network obtained by integrating out $\mathbf{Z}$.

3 Theory

For the theory we start in Section 3.1 by introducing the *dynamic tree* model. Then in Section 3.2 we consider inference in the *dynamic tree*. Williams and Adams (1999) [33] deals with sampling approaches; here the focus is on approximate inference. The mean field algorithm for the *dynamic tree* – the simplest of the variational approximations – will be discussed and this will then be used as the inference engine for the DT learning algorithms DT subsequently developed in Section 3.3.

3.1 The Dynamic Tree Model

There are two essential components that make up a *dynamic tree* network (i) the tree architecture and (ii) the nodes and conditional probability tables (CPTs) in the given tree. We consider the architecture question first.

Consider a number of nodes arranged into layers, as in Figure 2(a). We wish to construct a tree structure so that any child node in a particular layer will be connected to a parent in the layer above. We also allow there to be a null parent for each layer, so that any child connected to it will become a new root. (Technically we are constructing a forest rather than a tree.) An example of a structure generated using this method is shown in Figure 2(c).

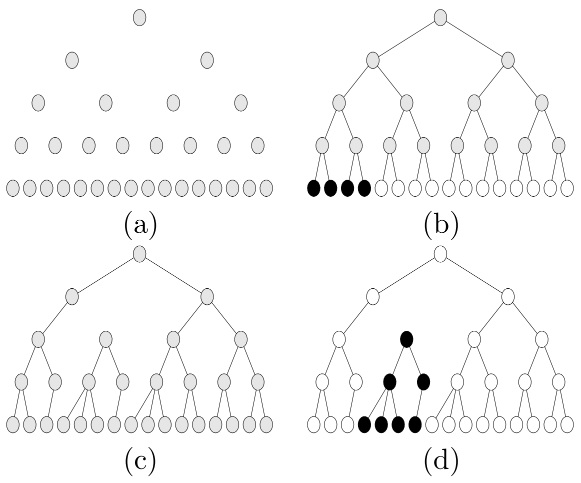


Figure 2: (a) “Naked” nodes, (b) the “balanced” tree architecture, (c) a sample from the prior over $\mathbf{Z}$, (d) data generated from the tree in (c).

There are a number of ways of specifying a prior over trees. If we denote by $\mathbf{z}_i$ the indicator vector which shows to which parent node i belongs, then the tree structure is specified by a matrix $\mathbf{Z}$ whose columns are the individual $\mathbf{z}_i$ vectors (one for each node). The scheme that we have investigated so far is to set $P(\mathbf{Z}) = \prod_i P(\mathbf{z}_i)$.

We specify $P(\mathbf{z}_i)$ as follows: Each child node is considered to have a “natural” parent—its parent in the balanced structure shown in Figure 2(b). Each node in the parent layer is assigned an “affinity” for each child node, and the “natural” parent has the highest affinity. Denote the affinity of that node i has for connecting to node k in the parent layer by a_{ik}. Then we set

$$P(\mathbf{z}_i = \mathbf{e}_j) \overset{\text{def}}{=} \pi_{ij} = \frac{e^{a_{ij}}}{\sum_{j'} e^{a_{ij'}}} \tag{1}$$

where $\mathbf{e}_j$ is the unit vector with a 1 in position j. Note that the “null” parent (regarded as node 0) is included in the sum, and has affinity a_{i0} associated with it, which determines the relative probability of “orphans”.

Having specified the prior over architectures, we now need to translate this into a TSBN. The units in the tree are taken to be C-class multinomial random variables. Each *layer* of the structure has associated with it a prior probability vector P_l and CPT θ_l. Given a particular $\mathbf{Z}$ matrix which specifies a forest structure, the probability of a particular instantiation of all of the random variables is simply the product of the probabilities of all of the trees, where the appropriate root probabilities and CPTs are picked up from the P_ls and θ_ls. A sample generated from the tree structure in Figure 2(c) is shown in Figure 2(d).

Our intuition as to why DTs may be useful image models is based on the idea that most pixels in an image are derived from a single object. We think of an object as being described by a root of a tree, with the scale of the object being determined by the level in the tree at which the root occurs. In this interpretation the CPTs will have most of their probability mass on the diagonal.

Given some data at the bottom layer of units, we can form a posterior over the tree structures and node instantiations of the layers above. This is rather like obtaining a set of parses for a number of sentences using a context-free grammar[1].

In the DT model as described above different examples are explained by different trees. This is an important difference with the usual priors over belief networks as used, e.g. in Bayesian averaging over model structures. Also, in the usual case of model averaging, there is normally no restriction to TSBN structures, or to tying the parameters (P_ls and θ_ls) between different structures.

3.2 Inference in DTs - The Mean Field Approximation

We now consider the problem of inference in DTs i.e. obtaining the posterior $P(\mathbf{Z}, \mathbf{X}_h|\mathbf{X}_v)$ where $\mathbf{Z}$ denotes the tree-structure, $\mathbf{X}_v$ the visible units (the image clamped on the lowest level) and $\mathbf{X}_h$ the hidden units. In fact, we shall concentrate on obtaining the posterior

[1]CFGs have a $O(n^3)$ algorithm to infer the MAP parse; however, this algorithm depends crucially on the one-dimensional ordering of the inputs. We believe that the possibility of crossed links in the DT architecture means that this kind of algorithm is not applicable to the DT case. Also, the DT model can be applied to 2-d images, where the $O(n^3)$ algorithm is not applicable.

marginal $P(\mathbf{Z}|\mathbf{X}_v)$, as we can obtain samples from $P(\mathbf{X}_h|\mathbf{X}_v, \mathbf{Z})$ using standard techniques for TSBNs.

There are a very large number of possible structures; in fact for a set of nodes created from a balanced tree with branching factor b and depth D (with the top level indexed by 1) there are $\prod_{d=2}^{D}(b^{(d-2)}+1)^{b^{(d-1)}}$ possible forest structures. Our objective will be to obtain the maximum a posteriori (MAP) state from the posterior $P(\mathbf{Z}|\mathbf{X}_v) \propto P(\mathbf{Z})P(\mathbf{X}_v|\mathbf{Z})$. For any $\mathbf{Z}$ it is possible to compute $P(\mathbf{X}_v|\mathbf{Z})$ (using Pearl message passing) and $P(\mathbf{Z})$. However, since the number of configurations of $\mathbf{Z}$ is typically very large it will usually be intractable to enumerate over them all and other approaches need to be adopted[2].

Sampling by Markov Chain Monte Carlo (MCMC) techniques, or search using Simulated Annealing are possibilities which have been considered in [33], but the drawback is that they are slow. An alternative to sampling from the posterior $P(\mathbf{Z}, \mathbf{X}_h|\mathbf{X}_v)$ is to use approximate inference. One possibility is to use a mean-field-type approximation to the posterior of the form $Q_Z(\mathbf{Z})Q_{X_h}(\mathbf{X}_h)$ (Zoubin Ghahramani, personal communication, 1998) and this is what we shall consider below. This material was first published in [2].

The *dynamic tree* can be considered as an ordered set U of nodes $i = 1, 2, \dots, U$, each of which taking on one of C possible states. If $\mathbf{Z}$ is used to denote the set of possible directed tree structures over these nodes then $z_{ij} = 1$ indicates that the node i is connected to parent j. By ordering the nodes such that upper level nodes have lower indices than those in the layer(s) below then it means that $z_{ij} \equiv 0$ for $j \geq i$. Finally defining $\mathbf{X} = \{x_i^k\}$ to be the set of all of the states of the nodes, then analogously with the z indicator variables, $x_i^k = 1$ if node i is in state k, and is zero otherwise.

Given the above notation we can define the prior of Section 3.1 as $P(\mathbf{Z}) = \prod_{ij} \pi_{ij}^{z_{ij}}$. The conditional probability tables define the state transition probabilities when traversing a link

[2]An interesting possibility is to use a structural EM (SEM) algorithm [15] to search for a structure $\mathbf{Z}$ having high posterior probability. Note that in contrast to Friedman's original work on SEM, in our case we would have to use this on each image $\mathbf{X}_v$ individually.

between a node j and its child i, where θ_{ij}^{kl} is the probability of moving from state l to state k during such a transition.

With these definitions the joint prior distribution can be written as follows

$$P(\mathbf{Z},\mathbf{X}|\boldsymbol{\theta},\boldsymbol{\pi}) = \prod_{i=1}^{U}\prod_{j=0}^{U}\pi_{ij}^{z_{ij}}\prod_{k,l=1}^{C}[\theta_{ij}^{kl}]^{x_i^k x_j^l z_{ij}} \tag{2}$$

where the indicator variables z, x pick out the correct probabilities, $\boldsymbol{\theta}$ denotes the set of CPTs in the model and $\boldsymbol{\pi}$ denotes the prior connection probabilities.

The nodes of the *dynamic tree* constitute two distinct sets. The first contains evidential or visible units $\mathbf{X}_v$ which are instantiated with the image data. The second are the hidden units $\mathbf{X}_h$ whose value has to be inferred. Conditioning on the training data (visible units) the posterior distribution of the *dynamic tree* takes the form

$$P(\mathbf{Z},\mathbf{X}_h|\mathbf{X}_v) = \frac{P(\mathbf{Z},\mathbf{X})}{P(\mathbf{X}_v)}. \tag{3}$$

In the mean field variational approach this posterior distribution is approximated by a factorising distribution of the form $Q(\mathbf{Z}|\mathbf{X}_v)Q(\mathbf{X}_h|\mathbf{X}_v)$ where $Q(\mathbf{Z}|\mathbf{X}_v)$ approximates over the $\mathbf{Z}$ distribution and $Q(\mathbf{X}_h|\mathbf{X}_v)$ the hidden units $\mathbf{X}_h$. (A full derivation is given in [1] for the interested reader.) Here the basic approach is described. For notational simplicity the conditioning on the image data $\mathbf{X}_v$ in the variational approximation is dropped, so that we write $Q(\mathbf{Z})Q(\mathbf{X}_h)$ instead of $Q(\mathbf{Z}|\mathbf{X}_v)Q(\mathbf{X}_h|\mathbf{X}_v)$.

The Kullback-Leibler (KL) divergence provides a convenient measure of the divergence between two probability distributions. Choosing good forms for the Q distribution is achieved by minimising the KL divergence between the approximating $Q(\mathbf{Z})Q(\mathbf{X}_h)$ and the true posterior distribution. The KL divergence is given by

$$\begin{aligned} KL(Q||P) &= \sum_{\mathbf{Z},\mathbf{X}_h} Q(\mathbf{Z})Q(\mathbf{X}_h)\log\left(\frac{Q(\mathbf{Z})Q(\mathbf{X}_h)}{P(\mathbf{Z},\mathbf{X}_h|\mathbf{X}_v)}\right) \\ &= \log P(\mathbf{X}_v) - \sum_{\mathbf{Z},\mathbf{X}_h} Q(\mathbf{Z})Q(\mathbf{X}_h)\left[\log P(\mathbf{Z},\mathbf{X}) - \log Q(\mathbf{Z}) - \log Q(\mathbf{X}_h)\right]. \quad (4) \end{aligned}$$

By assuming a factorising form for the Qs mean field inference makes the structure variables $\mathbf{Z}$ be independent of the node states $\mathbf{X}$ which allows us to treat them separately. In the true posterior distribution they are of course not independent - so an exact fit is unlikely - however this makes the computation tractable. We start by optimising with respect to the $Q(\mathbf{Z})$ distribution.

Calculating Q(Z): To optimise the KL divergence for the $\mathbf{Z}$s, $Q(\mathbf{X}_h)$ is fixed and a minimisation with respect to $Q(\mathbf{Z})$, subject to the constraint $\sum_j Q(z_{ij}) = 1, \forall i$ is performed. The results of the analysis produces the following expression for $Q(\mathbf{Z})$

$$Q(\mathbf{Z}) = \prod_{ij} \frac{\exp(z_{ij}\lambda_{ij})}{\sum_s \exp(\lambda_{is})}, \quad (5)$$

where

$$\lambda_{ij} = \log \pi_{ij} + \sum_{kl} \langle x_i^k x_j^l \rangle_{Q(X)} \log \theta_{ij}^{kl}. \quad (6)$$

So with $Q(\mathbf{X}_h)$ fixed we can explicitly calculate the optimal $Q(\mathbf{Z})$, and interestingly this results in a fully factorised form for $Q(\mathbf{Z})$.

Calculating Q(X): To complete the optimisation we now need to minimise the KL divergence for $Q(\mathbf{X}_h)$ keeping $Q(\mathbf{Z})$ fixed. Substituting for $P(\mathbf{Z},\mathbf{X})$ from Equation (2) into the expression for the KL divergence (Equation (4)) gives

$$\begin{aligned}
KL(Q||P) &= \log P(\mathbf{X}_v) + \sum_{\mathbf{Z}} Q(\mathbf{Z}) \log Q(\mathbf{Z}) + \sum_{\mathbf{X}_h} Q(\mathbf{X}_h) \log Q(\mathbf{X}_h) \\
&\quad - \sum_{\mathbf{Z},\mathbf{X}_h} Q(\mathbf{Z}) Q(\mathbf{X}_h) \log \left[\prod_{i=1}^{u} \prod_{j=0}^{u} \pi_{ij}^{z_{ij}} \prod_{kl} [\theta_{ij}^{kl}]^{x_i^k x_j^l z_{ij}} \right] \\
&= \sum_{ij} \mu_{ij} \left[\log \pi_{ij} + \sum_{kl} \langle x_i^k x_j^l \rangle_{Q(\mathbf{X}_h)} \log \theta_{ij}^{kl} \right] + \sum_{\mathbf{X}_h} Q(\mathbf{X}_h) \log Q(\mathbf{X}_h) + f(\mathbf{Z}) \quad (7)
\end{aligned}$$

where $\mu_{ij} = \langle z_{ij} \rangle_{Q(\mathbf{Z})}$, and $f(\mathbf{Z})$ contains only $Q(\mathbf{Z})$ terms which are constant in the minimisation with respect to $Q(\mathbf{X}_h)$.

Evaluating the $\langle x_i^k x_j^l \rangle_{Q(\mathbf{X}_h)}$ term is intractable and presents a problem. To proceed any further it is necessary to make the assumption of a factorised form for $Q(\mathbf{X}_h)$:

$$Q(\mathbf{X}_h) = \prod_{i \in h} Q(X_i) = \prod_{k,i \in h} (m_i^k)^{x_i^k}. \quad (8)$$

In the above expression m_i^k is the probability under $Q(\mathbf{X}_h)$ that node i is in state k. It is the mean that node i will be in state k independent of the other hidden nodes in the network. This results in the fully factorised distribution for $Q(\mathbf{Z}, \mathbf{X}_h)$ which is the characteristic hallmark of the mean field approximation and is what makes it the simplest of all the variational techniques.

This assumption allows us to make the following substitution

$$\langle x_i^k x_j^l \rangle_{Q(X)} = m_i^k m_j^l. \quad (9)$$

Care must be taken to ensure that $\sum_k m_i^k = 1, \forall i$ and this is achieved simply by the addition of a Lagrange multiplier term $\sum_\alpha \rho_\alpha \left(\sum_\beta m_\alpha^\beta - 1 \right)$. A straightforward application of calculus gives the following expression for the means

$$m_s^r = \frac{\exp(\gamma_s^r)}{\sum_{r'} \exp(\gamma_s^{r'})}, \tag{10}$$

where

$$\gamma_s^r = \sum_{j<i} \sum_l \mu_{sj} m_j^l \log \theta_{sj}^{rl} + \sum_i \sum_k \mu_{is} m_i^k \log \theta_{is}^{kr}. \tag{11}$$

Equations (10) and (11) form a set of coupled mean field equations which can be solved by an iterative update [28]. This update is performed asynchronously on each of the nodes and repeated cyclically until convergence is reached.

The whole procedure: Equations (5) and (10) provide the necessary results to perform an optimisation on the KL divergence. The complete procedure is as follows.

1. Initialise all μ_{ij} and m_i^k. The m_i^k's are initialised to $1/C$ plus a small amount of zero-mean Gaussian noise to break symmetry. The μ_{ij}'s are initialised randomly so that $\sum_j \mu_{ij} = 1$ over those parents that have non-zero probability of connection.
2. Cyclically update Equations (10) to find the local optimum for the means[3] m_i^k.
3. The $Q(\mathbf{Z})$s can then be calculated directly from Equation (5).
4. Repeat steps 2–3 until converged.

Note that each step of the process is guaranteed not to increase the KL divergence (4), and as the KL divergence is bounded from below by 0, convergence is assured.

Mean field inference is therefore an attractive approach as it replaces the intractable dependencies between **X** and **Z** in the true posterior with simpler computations which can be solved iteratively. This allows us to fully enumerate over tree structures **Z**, but at the

[3]Note that $\langle x_i^k x_j^l \rangle_{Q(X)}$ needs only to be computed for $j < i$ as $z_{ij} \equiv 0$ for $j \geq i$.

cost of now only having an approximation to the true posterior distribution. In the next Section we will develop learning algorithms for the *dynamic tree* based upon this mean field approximation.

3.3 Learning in Dynamic Trees

3.3.1 An EM update for learning the CPTs

Given a training set of $p = 1 \ldots P$ patterns, the log likelihood of the data under the *dynamic tree* model is given by

$$\sum_p \log P(\mathbf{X}_v^p) = \sum_{p=1}^{P} \log \sum_{\mathbf{Z}^p, \mathbf{X}_h^p} P(\mathbf{X}_v^p, \mathbf{X}_h^p | \mathbf{Z}^p, \boldsymbol{\theta}) P(\mathbf{Z}^p | \boldsymbol{\pi}). \tag{12}$$

Note that the $\mathbf{Z}$s are summed over T tree configurations, and for each there will be a different $\mathbf{X}_h$. Notation for this is omitted for clarity.

To assign each parent-child combination its own unique CPT would lead to massive over-parameterisation for the limited training data usually available, so it was deemed sensible to share the CPTs among nodes on the same level (scale). θ_I is used to denote the shared CPT for the set of nodes $\mathbf{X}_I$.

Standard calculus and the use of Lagrange multipliers to ensure that the CPTs are valid probabilities, produces the following EM update for the CPT element θ_{Ij}^{kl} representing the transition probability $P(x_i^k | x_j^l)$

$$\hat{\theta}_{Ij}^{kl} = \frac{\sum_{p,\mathbf{Z}^p} \sum_{x_i \in \mathbf{X}_I} P(x_i^{k(p)}, x_j^{l(p)} | \mathbf{X}_v^p, \mathbf{Z}^p, \boldsymbol{\theta}) P(\mathbf{Z}^p | \mathbf{X}_v^p, \boldsymbol{\phi})}{\sum_{p,\mathbf{Z}^p} \sum_{x_i \in \mathbf{X}_I} \sum_{k'} P(x_i^{k'(p)}, x_j^{l(p)} | \mathbf{X}_v^p, \mathbf{Z}^p, \boldsymbol{\theta}) P(\mathbf{Z}^p | \mathbf{X}_v^p, \boldsymbol{\phi})}, \tag{13}$$

where

$$P(x_i^k, x_j^l | \mathbf{X}_v^p, \mathbf{Z}^p, \boldsymbol{\theta}) = \frac{1}{\sum_{l'} \pi(x_j^{l'}) \lambda(x_j^{l'})} \lambda(x_i^k | \boldsymbol{\theta}) \theta_{Ij}^{kl} \pi(x_j^l | \boldsymbol{\theta}) \prod_{y \in s(x_i)} \lambda_y(x_j^l | \boldsymbol{\theta}). \tag{14}$$

The λs and πs are the Pearl messages used to pass information to a node about the states of its children and parents respectively [27], and $s(x_i)$ is the set of siblings of node i. This derivation is an extension of that given in [12] for fixed architecture TSBNs; full details of which given in [1].

3.3.2 Mean Field EM in Dynamic Trees

In the mean field DT the true posterior distribution $P(\mathbf{X}_h, \mathbf{Z} | \mathbf{X}_v, \boldsymbol{\theta}, \boldsymbol{\pi})$ is approximated by a factorising distribution, $Q(\mathbf{X}_h, \mathbf{Z} | \mathbf{X}_v) = Q(\mathbf{X}_h) Q(\mathbf{Z})$. This can be used to find a lower bound on the log-likelihood of the data

$$\sum_p \log P(\mathbf{X}_v^p) \geq \sum_{p, \mathbf{X}_h^p, \mathbf{Z}^p} Q(\mathbf{X}_h^p, \mathbf{Z}^p | \mathbf{X}_v^p) \log \frac{P(\mathbf{X}_v^p, \mathbf{X}_h^p, \mathbf{Z}^p | \boldsymbol{\theta}, \boldsymbol{\phi})}{Q(\mathbf{X}_h^p, \mathbf{Z}^p | \mathbf{X}_v^p)} \overset{def}{=} \mathcal{L}(Q, \boldsymbol{\theta}, \boldsymbol{\pi}). \tag{15}$$

We call $\mathcal{L}(Q, \boldsymbol{\theta}, \boldsymbol{\pi})$ the variational log likelihood. The lower bound can be shown to be tightest when the KL-divergence between the approximating distribution and the true posterior is minimised, and suggests an iterative EM-style algorithm for variational methods, as proposed in [22]. In the E-step the bound (variational log-likelihood) is maximised wrt Q holding $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ fixed (by minimising the KL-divergence). Then in the M-step Q is fixed and the bound is maximised wrt to the model parameters. The CPTs $\boldsymbol{\theta}$ and the connection probabilities $\boldsymbol{\pi}$ constitute two very distinct parameter types and as such require different treatment.

Learning the CPTs: The derivation uses a similar methodology to that of exact EM (see Equation (13) above). Performing this optimisation on the DT gives rise to the following update rule for the CPTs

$$\hat{\theta}_{Ij}^{kl} = \frac{\sum_{p,\mathbf{Z}^{(p)}} \sum_{x_i \in \mathbf{X}_I} Q(X_i^{k(p)}) Q(X_j^{l(p)}) Q(\mathbf{Z}^{(p)}) z_{ij}^{(p)}}{\sum_{k'} \sum_{p,\mathbf{Z}^{(p)}} \sum_{x_i \in \mathbf{X}_I} Q(X_i^{k'(p)}) Q(X_j^{l(p)}) Q(\mathbf{Z}^{(p)}) z_{ij}^{(p)}}. \tag{16}$$

Learning the Affinities: The affinities set a prior over tree structures. As for the CPTs affinities also can be shared between nodes to reduce parameterisation. a_{ij} is the individual *affinity* of node i for parent j, as defined in Section 3.1. Let $\boldsymbol{\phi}$ denote such sets of shared affinities. To obtain an update we maximise the bound on the log likelihood (Equation (15)) with respect to $\boldsymbol{\phi}$.

Differentiating Equation (15) with respect to a shared affinity parameter ϕ gives

$$\frac{\partial \mathcal{L}(Q, \boldsymbol{\theta}, \boldsymbol{\phi})}{\partial \phi} = \sum_{p} \sum_{i,j:a_{i,j} \in \phi} \left[\mu_{ij} - \pi_{ij} \right]. \tag{17}$$

(See [1] for the complete derivation.) Because of the softmax relationship between affinities and π's as given in equation (1), we cannot obtain an EM-style update for $\boldsymbol{\phi}$. However, $\mathcal{L}$ can be optimised wrt $\boldsymbol{\phi}$ using standard gradient optimisation techniques such as conjugate gradients.

The Complete Learning Algorithm therefore consists of fixing the model parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, and running mean field until the Qs reach convergence. Then we fix the Qs and calculate the update for the CPT using Equation (16). A gradient based optimiser can then be used on Equation (17) to traverse one or more steps along the affinity gradient. The $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ parameters are then updated, and the process repeated.

3.4 Handling Missing Data

A reality of dealing with the real world is that frequently we are given examples which are incomplete, broaching the issue of how do we deal with missing data? If we simply set the missing pixel to a random value we may well end up learning noise, and even a uniform instantiation with all states equi-probable could still be saying something which is untrue.

Assuming that the data is missing at random (see [24]) the correct solution is to marginalize out the uninstantiated variables. For exact inference using Pearl message passing in belief networks this is achieved automatically by the algorithm when the λ value of the uninstantiated node is set to $(1, 1, \ldots, 1)$. With the mean field approximation a little care is required as it is unclear as to whether an uninstantiated leaf node may exert any unintended influence on the resultant equilibrium distribution. The solution we have used is to temporarily modify the connection probability table so that missing data leaf nodes have a probability of disconnection of 1.0. They then do not contribute anything towards the mean field equilibrium distribution. The implications for the learning rule update equations is that such nodes should be ignored for the given example.

An alternative to marginalising out unlabelled pixels would be to model them as a class in their own right. This is entirely possible within the framework of the current algorithm and can be justified within a principled approach. However, as unlabelled pixels are only an artifact of the labelling scheme and are not a property of the distribution that underlies the images which we are trying to model we prefer the solution given above.

4 Experiments

To illustrate the operation of the *dynamic tree* model we consider firstly inference, and in Section 4.1 briefly examine mean field DT inference on artificially generated 1-d images. Section 4.2 then employs the learning algorithms developed earlier to learn the *dynamic* *tree* model parameters for a database of images of outdoor scenes.

4.1 Inference in the Dynamic Tree

In this Section we illustrate the *dynamic tree* in operation. We consider a 6 layer binary tree which models 32 pixel 1-d images. Binary variables were used for each node. The image data was generated by sampling from a *dynamic tree* having the same node configuration. The CPTs had values of 0.99 on the diagonal. The affinities were set with $a_{nat} = 0$, $a_{null} = -0.625$ and the affinity for the Nth nearest neighbours of the natural parent was $-1.25N$.

In [33] simulated annealing was used to find maximum a posteriori (MAP) DT configurations. Figure 3(a)–(c) shows the MAP configurations found for 3 examples, and we see that the *dynamic tree* very nicely constructs sub-trees for each of the different regions in the images, and the height of these sub-trees reflects the size of the region it models. Thus *dynamic trees* capture regions in an image as distinct "objects" and the size of the tree reflects the number of pixels an "object" spans which is a very desirable property.

Mean field inference (unlike sampling) yields a full distribution for the posterior. However, by selecting the highest link probability $Q(\mathbf{z}_i)$ for each node i we can construct the highest probability structure. We call this the highest probability mean field (HPMF) configuration, and the corresponding HPMF trees found for the same image data as used in the annealing runs are shown in Figure 3(d)–(f). We see that mean field inference also captures the "objects" in the images, though they tend to be squashed into smaller trees. This is primarily due to the independence assumption that mean field makes between each node which weakens the influence of the instantiated image data over nodes further away. However, despite this mean field still produces very favourable interpretations and a considerably faster speed, which makes it very attractive for use in learning algorithms for the *dynamic tree's* parameters. This is addressed in the next Section.

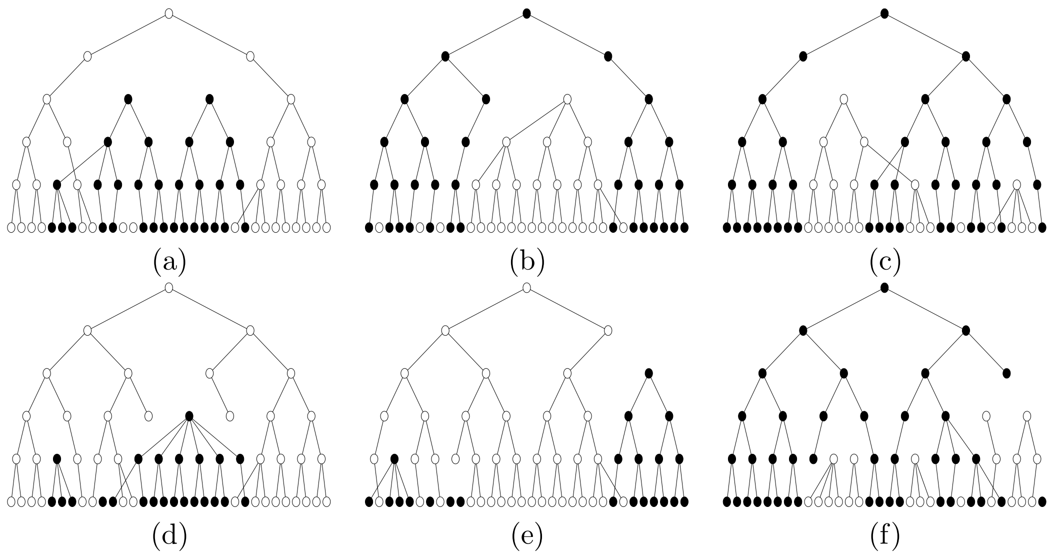


Figure 3: (a)-(c) The MAP trees found by annealing for 3 different images, and (d)-(f) the corresponding HPMF structures.

4.2 Learning on Real Images

4.2.1 The Image Data

We use a set of colour images of out-door scenes from the Sowerby Image database[4] for our experiments. These feature both rural and urban examples and contain many of the typical objects you would expect to find – such as the roads, cars and buildings of urban environments to the country lanes and fields of the rural. The original scenes were photographed using small-grain 35mm transparency film under carefully controlled conditions at a number of locations in and around Bristol, UK. The analogue transparencies were then digitised using a calibrated scanner to produce high quality 24-bit colour representations.

In addition to the raw images the database also contains corresponding labelled examples created by over-segmenting the images and then hand labelling each region produced. This

[4]This database can be made available to other researchers. Please contact Dr Andy Wright or Dr Gareth Rees, Advanced Information Processing Department, Advanced Technology Centre - Sowerby, BAE SYSTEMS Ltd, PO Box 5 Filton, Bristol, BS34 7QW, UK, email: `gareth-s.rees@baesystems.com` for details.

gave rise to 92 labels, organised hierarchically. A fixed TSBN model has already been applied to this database [13] in which the 92 class labels were merged down to 7 super classes. Since the fixed TSBN can be viewed as a special case of the *dynamic tree* where the architecture is not allowed to change this provides an excellent model for comparison and consequently we chose to adopt the same class labels.

The labels distinguish all of the key regions of interest in the image, representing "sky", "vegetation", "road markings", "road surface", "building", "street furniture" and "mobile object." Such is the nature of gathering real data that circumstances inevitably arise where there is missing data, so a further dummy "unlabelled" class is added to accommodate this. Unlike the others the unlabelled class is not learned since it is an artifact of the labelling strategy adopted and is dealt with as described in Section 3.4. Figure 4(a) shows one urban and one rural scene from the database.

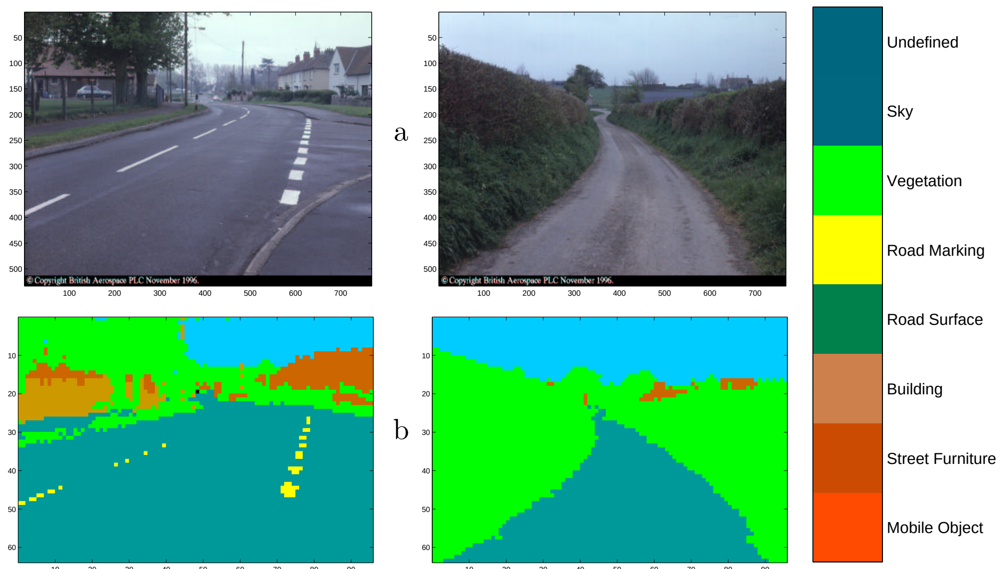


Figure 4: (a) An urban and a rural scene from the Sowerby database and (b) their corresponding labelled images. To the right is a key defining the labels used.

The *dynamic tree* as formulated operates with discrete classes hence it is necessary to use the labelled images instead of the real valued data. However there are a wealth of established techniques which can be used to produce a mapping from real valued pixels to a discrete number of classes. For example in [12] a multi-layer perceptron (MLP) is trained to do this. The reverse mapping from labels to real valued pixels is equally straightforward, e.g. using Gaussian mixture models as used in [5].

4.2.2 Experimental Procedure

The Sowerby database contains 104 images which were randomly divided by Feng *et al* [12] into a training set of 61 images and the rest allocated as a test set. For comparative purposes we use the same training and test sets to learn the *dynamic tree* model parameters.

The full-size images are 768×512 pixels, which Feng *et al* [13] reduced by sub-dividing them into regions and adopting a majority voting strategy to choose the winning class label. In cases where there was a tie a label was chosen probabilistically from the competing classes based on the prior probabilities of the given labels being seen in the images. The class label prior probabilities for the training and test sets are given in Table 1.

Class	**P(Label)**	
	Training Set	Test Set
Unlabelled	0.0036	0.0418
Atmospheric phenomena	0.1443	0.1140
Vegetation	0.3703	0.3899
Road surface markings	0.0012	0.0008
Road surface	0.4210	0.3804
Building	0.0473	0.0569
Street furniture	0.0056	0.0112
Mobile object	0.0067	0.0050

Table 1: Image pixel class priors for the training and test sets.

We adopt the same procedure and downsample to an image size of 96×64 pixels. An urban and rural examples at this resolution are shown in Figure 4(b) and it can be clearly seen that most of the detail is still present.

For the experiments we use a 7-level model. The node arrangement is quad-tree for all bar the second level where there are 6 instead of 4 nodes. The latter is to accommodate the $3 : 2$ aspect ratio of the training images and produces the desired image size of 96×64 pixels.

Setting the initial model parameters is an open question and probably a paper in itself. This issue has been explored in previous work [4, 13] and in keeping with the philosophy that a child node would favour being in the same state as its parent we initialise the CPTs to be strongly diagonal with probability 0.9 and the rest of the probability mass is shared equally among the 6 other states. The prior for root nodes being in a particular state was set to be uniform. All models were initialised with the above CPTs and state prior.

The affinities are given relative to the *natural parent* affinity. The natural parent of a node is the parent it would have if it were part of a balanced tree. This affinity is set to 0 for reference purposes. Important other connections a node may wish to make are to the nearest neighbour(s) of its natural parent, or to disconnect and become a root. Equation (1) is used to map the affinity values into probabilities. In [1] investigation showed a useful working range for the *dynamic tree* was for affinity values of 0 and -3 for the nearest neighbour and null connections respectively. Other connections are given a probability of zero. The interpretation for this is that nearest neighbour connections are equi-probable with the natural parent and disconnections are possible but with a far lower probability.

Specifying each of the model parameters individually is unwise considering the limited amount of training data usually available. Here we chose to share the CPTs on a level-by-level basis and learn the affinities a_{nn} and a_{null} as single parameters over the whole *dynamic tree*.

In the experiments on the DT the mean field EM learning algorithm described in Section 3.3.2 was used. Firstly mean field inference was run over all of the training examples to produce an approximation of the joint distribution for the E-step of the algorithm. This involved updating the means cyclically for a number of iterations until equilibrium was reached, then updating the structure $Q(\mathbf{Z})$ in single step and repeating until convergence. In practice the 5 complete iterations were sufficient but the algorithm was allowed to terminate early if the variational log-likelihood altered by less than 0.05 between cycles.

The M-step of the algorithm is a single step update for the CPTs, but for the affinities a gradient method is necessary. Conjugate gradients was used with the optimiser being allowed to take up to 3 steps. Three steps were necessary in order to take advantage of the conjugate gradients—performing only one would simply be gradient descent. After calculating new estimates for the CPTs and affinities all of the model parameters were updated and the process repeated.

A comparison was made between three types of model, the fixed quad-tree (fixed architecture), a *dynamic tree* where only the CPTs were learned (CPT-only DT) and the *dynamic tree* model where all parameters were learned (full DT). All used the mean field EM algorithm as summarised in Section 3.3.2 (described fully in [1]), and additionally exact EM learning was performed on a fixed architecture quad-tree model using the exact EM learning update given in Section 3.3.1. (For a further comparison of mean field EM against the exact method in fixed trees see [4].)

4.3 Experimental results

The learning curves on the training set of 61 patterns are given in Figure 5, showing the variational log-likelihoods obtained by the three DT models learned using the mean field EM approach, and the log-likelihood of the fixed architecture quad-tree model learned by exact EM.

Considering firstly mean field EM learning, it can be seen from the Figure that while the fixed architecture mean field EM model shows good improvement in variational log-likelihood during training it can only go so far. Learning only the CPTs of a *dynamic*

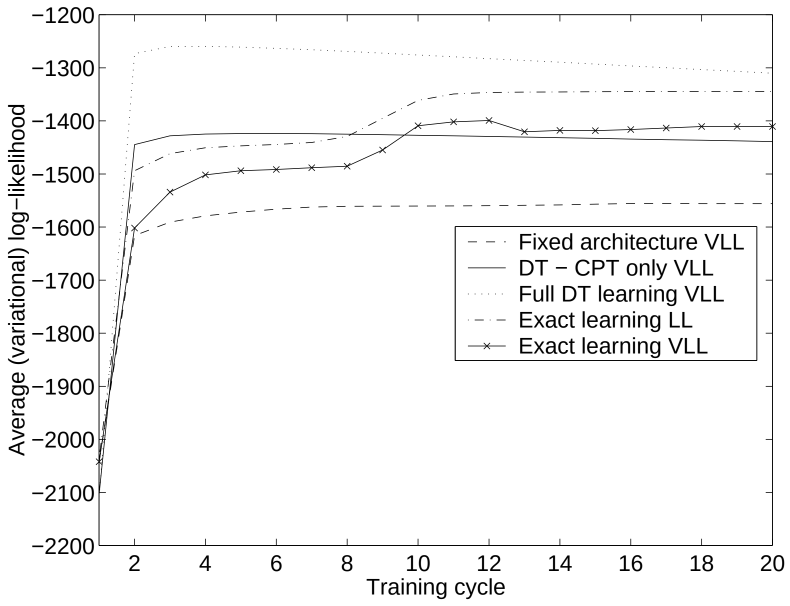


Figure 5: Learning curves on the 61 image training set for the fixed tree, CPT-only learned *dynamic tree*, fully learned *dynamic tree* models, and exact EM learning on the fixed tree.

tree model whose affinities were set from observed optimal parameters in toy data (see [4]) appears to offer an advantage over the fixed architecture model with increased variational log-likelihood over the training set. The full DT model does better still, indicating that having variable architecture offers an advantage over the fixed architecture model. Although it is not surprising that the CPT-only DT and full DT models obtain higher variational log likelihoods (because they have more free parameters), the results show that the search procedure for learning these parameters is working sufficiently well that these gains are realized.

Exact EM on the fixed architecture model can be seen to obtain a higher log-likelihood than variational log-likelihood bound given by both the fixed and CPT-only mean field learning models. This is not surprising as the exact approach using the true probability distribution is able give the actual log-likelihood whereas mean field gives only a lower bound on the likelihood of the data $P(\mathbf{X}_v)$, and the comparisons made in [4] suggest that it may not be that tight. To confirm this on the real data the mean field variational log-likelihood was calculated during training at each iteration of the model learned under exact EM. This is shown as the crossed line in the plot of Figure 5 and we see clearly that the variational log likelihood does indeed lie significantly below the exact log likelihood. However, we observe that the variational log likelihood obtained by mean field learning on the full *dynamic tree* beats the exact log likelihood for EM training of the fixed architecture model, so clearly even though we can't perform exact learning on the *dynamic tree* model – which we would like to do – it still out-performs the fixed architecture model learned by exact approaches.

Though apparently fairly stable, it can be seen from the plot that the average variational log-likelihood for both variants of the DT appears to decline after 3–4 training iterations. A known weakness of the mean field learning algorithm [4] is that it tends to settle at unstable equilibria where even a slight perturbation of the parameters can cause "spontaneous symmetry breaking." In an experiment on toy data [1] it was observed that in subsequent iterations the model tries to correct this by hardening the CPTs resulting in a drop in performance, and this is probably what is happening here.

The DT and fixed-architecture TSBN are generative, probabilistic models of label images. We can evaluate the quality of the learned models by calculating the (variational) log-likelihood on the test set of 43 images. This can then be used to obtain the average coding cost in bits/pixel (bpp), where the coding cost of an image $\mathbf{X}_v^p$ is given by

$$\text{Coding cost} = -\frac{\log_2 P(\mathbf{X}_v^p)}{\#\text{ labelled pixels in image}}. \tag{18}$$

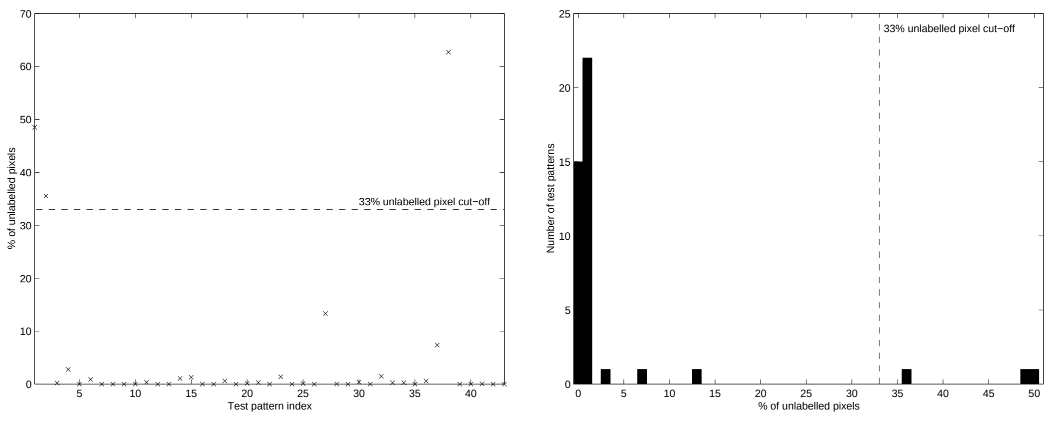
An important point to remember is that with variational methods we are calculating a lower bound on the likelihood (an upper bound on coding cost), and the likelihood can only be at least as good or better.

Table 2 gives these coding costs for the various models and is compared with the lossless JPEG-LS codec[5] which is available from `http://www.hpl.hp.com/loco/`. The second col-

[5]In the case of JPEG-LS the coding cost was obtained by compressing the images and measuring the size

Model		**After Training Cycle**	**Bound on Coding Cost** (bpp)	
			Full	Less than 33% missing
Mean field	Fixed architecture	15	0.8588	0.3918
	DT - CPT only	2	0.4089	0.3228
	Full *dynamic tree*	2	0.3805	0.2942
Exact EM	Fixed architecture	10	0.3421	0.3253
JPEG-LS		–	0.3810	0.3782

Table 2: Performance on the test set of 43 images.



(a) (b)

Figure 6: Percentage of unlabelled pixels in each of the 43 test set images (a) per image and, (b) as histogram.

umn of the table gives the number of training epochs used to train each of the models before applying them on the test data. In an attempt to minimise over-training this usually occurred at the point where the training error first peaked (see Figure 5). We see from the third column that on the full test set JPEG-LS outperforms all except the full DT and exact EM models.

However, an examination of the percentage of unlabelled pixels in each of the images of the test set (Figure 6) shows 3 images to have greater than 33% unlabelled pixels which is extremely unusual. Ordinarily we might expect some unlabelled pixels and so need a robustness to them, but images degraded to that extent could reasonably be rejected as

of the compressed files.

bad data. Removing these 3 images gives average coding costs as listed in the final column of the table. As can be seen now even the fixed tree is comparable to JPEG-LS and the full *dynamic tree* offers significant improvements over them both – the DT model was found to have a higher variational log-likelihood than the fixed architecture model in 42/43 of the test images.

Exact EM learning of the fixed architecture quad-tree tree also performs well. Over the full set of test images it achieves a lower coding cost than even the full *dynamic tree.* Note that for the fixed architecture quad-tree tree it is possible to integrate out the unlabelled pixels exactly. In contrast, the mean field approximation is expected to be quite poor when there is little data so it is likely to perform badly on images with many missing pixels, and indeed we see this to be the case. (The variational log-likelihoods of the 3 images with more than 33% missing pixels are significantly lower than those of the other 40 images.) One reason for this poor performance is that with fewer constraints there is a greater chance of multimodality in the distribution $P(\mathbf{X}_h, \mathbf{Z}|\mathbf{X}_v)$, and this multimodality is typically poorly represented in the mean field distribution that is settled to. A similar effect was observed in [32] where the free energy of the an "unclamped" Boltzmann machine was not well approximated by mean field methods. So it is not surprising that the fixed architecture model learned by exact EM achieves a slightly lower coding cost than the *dynamic tree* over the full test set of 43 images where there are many unlabelled pixels – especially as it has the advantage of using the true probability distribution for the model. However, after removing the 3 images with more than 33% missing pixels the *dynamic tree* model achieves a significantly lower coding cost (0.294 bpp) than the exact-EM trained quad-tree model (0.325 bpp). On this set of 40 test images, lossless JPEG obtains 0.378 bpp compression.

5 Discussion

In this paper we have considered inference and learning in the *dynamic tree* model and applied it to learning a database of outdoor scenes.

We have seen that the *dynamic tree* model overcomes the blocky segmentation problems of TSBNs and that its dynamic architecture enables the creation of structures that well explain the image data under consideration. It was observed that the mean field inference algorithm was able to find good posterior solutions, rivalling the MAP structures found by simulated annealing (see section 4.1). We were also able to apply and train the *dynamic tree* model on real data, and demonstrate its superiority over the fixed-architecture TSBN model.

There are opportunities to further develop DTs in a number of directions:

1. Using real-valued nodes. The nodes in the DT architecture described above are discrete multinomial variables. However, the model can be developed so that real-valued variables are also used. For example, these real-valued variables could contain information about the instantiation parameters of objects. Some work has been done in this direction in [29] in which the nodes have position as well as discrete state. Alternatively real-valued node variables could replace discrete ones completely to create a real-valued version of the *dynamic tree* Gaussian distributions would be a obvious choice due to their tractable properties.

2. Developing a sparse prior over trees. In the prior over trees described above, we retained the number of nodes in each layer as used in a balanced tree. It would be desirable to have more nodes in each layer, with many being switched off if not needed. The simple prior used above where every node chooses its parent independently would not work in this case, as there would be a low probability of creating tree structures. Some initial exploration of this idea was carried out in [3], but more remains to be

done.

3. Using more complex variational approximations. The mean field method used above for $\mathbf{X}_h$ is the simplest variational approximation that can be used as it is fully factorised. It would be interesting to specify an alternative distribution $Q(\mathbf{X}_h)$ which allows correlations between the $\mathbf{X}_h$'s. One possibility, that of using a dynamic tree model for this distribution, is investigated in [29].

4. Other datasets. The methods described here are clearly not limited to the road-scene data that was used. For example, in remote sensing tasks there is often a need to produce pixelwise labels (e.g. of different land usage) from noisy observations. Here a spatial prior as provided by the *dynamic tree* would be very useful when combined with local pixelwise predictions of the labels.

Acknowledgements

This work started from a conversation between CW and Zoubin Ghahramani at the Isaac Newton Institute in October 1997. We thank Zoubin Ghahramani, Geoff Hinton and Peter Dayan for helpful conversations, and the Isaac Newton Institute for Mathematical Sciences (Cambridge, UK) for hospitality during the "Neural Networks and Machine Learning" programme. We also thank Amos Storkey for his input on the mean field theory for *dynamic trees*, and the anonymous referees whose comments helped to improve the paper. NJA was supported by an EPSRC research studentship, and the work of CW was supported by EPSRC grant GR/L78181, *Probabilistic Models for Sequences.*

References

[1] N. J. Adams. *Dynamic Trees: A Hierarchical Probabilistic Approach to Image Modelling.* PhD thesis, Institute for Adaptive and Neural Computation, Artificial Intelligence, Division of Informatics, University of Edinburgh, 5 Forrest Hill, Edinburgh, EH1 2QL, UK, 2001. Forthcoming.

[2] N. J. Adams, A. J. Storkey, Z. Ghahramani, and C. K. I. Williams. MFDTs: Mean Field

Dynamic Trees. In A. Sanfeliu, J. J. Villanueva, A. Vanrell, R. Alquézar, T. Huang, and J. Serra, editors, *Proceedings of 15th International Conference Pattern Recognition*, volume 3, *Image speech and Signal Processing*, pages 151–154. IEEE Computer Society, September 2000.

[3] N. J. Adams and C. K. I. Williams. SDTs: Sparse Dynamic Trees. In *Proceedings of 9th International Conference on Artificial Neural Networks*, pages 527–532. IEE, September 1999.

[4] N. J. Adams, C. K. I. Williams, and A. J. Storkey. Comparing Mean Field and Exact EM in Tree Structured Belief Networks. In *Fourth International ICSC Symposium on Soft Computing and Intelligent Systems for Industry.* ICSC-NAISO Adademic Press, June 2001.

[5] C. A. Bouman and M. Shapiro. A Multliscale Random Field Model for Bayesian Image Segmentation. *IEEE Transactions on Image Processing*, 3(2):162–177, March 1994.

[6] E. Charniak. *Statistical Language Learning.* MIT Press, Cambridge, Massachusetts, 1993.

[7] R. Chellappa and S. Chatterie. Classification of Textures using Guassian Markov Random Fields. In *IEEE Trans. Accoust., Speech and Signal Processing*, volume 33, pages 959–963, 1985.

[8] P. A. Chou. Recgonition of Equations Using a Two-Dimensional Stochastic Context-Free Grammar. *Visual Communications and Image Processing IV*, 1199:852–863, 1989.

[9] M. Crouse, R. Nowak, and R. Baraniuk. Wavelet-based statistical signal proccessing using hidden Markov models. *IEEE Transactions on Signal Processing*, 46:886–902, 1998.

[10] P. Dayan, G. E. Hinton, R. M. Neal, and R. S. Zemel. The Helmholtz Machine. *Neural Computation*, 7(5), 1995.

[11] J. S. De Bonet and P. A. Viola. A Non-Parametric Multi-Scale Statistical Model for Natural Images. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems 10.* MIT Press, Cambridge, MA, 1998.

[12] X. Feng and C. K. I. Williams. Training Bayesian Networks for Image Segmentation. In *Proceedings of SPIE*, volume 3457, July 1998.

[13] X. Feng, C. K. I. Williams, and S. N. Felderhof. Combining Belief Networks and Neural Networks for Scene Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, March 2001. Accepted for publication.

[14] P. W. Fieguth, A. S. Willsky, and W. C. Karl. Multiresolution Stochastic Imaging of Satellite Oceanographic Altimetric data. *IEEE International Conference on Image Processing*, 2:1–5, 1994.

[15] N. Friedman. The bayesian structural em algorithm. In *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–98)*, pages 129–138, San Francisco, CA, 1998. Morgan Kaufmann Publishers.

[16] D. Geiger and D. Heckerman. Knowledge Representation and Inference in Similarity Networks and Bayesian Multinets. *Artificial Intelligence*, 82:45–74, 1996.

[17] S. Geman and D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 6, no. 6, pages 721–741, November 1984.

[18] S. Geman and K. Manbeck. Experiments in Syntactic Recognition. Technical Report CICS-P-411, Division of Applied Mathematics, Brown University, Providence, RI 02912 USA, March 1994.

[19] G. E. Hinton, B. Sallans, and Z. Ghahramani. A Hierarchical Community of Experts. In C. M. Bishop, editor, *Neural Networks and Machine Learning*. Springer-Verlag New York inc., 1998.

[20] G.E. Hinton, Z Ghahramani, and Y. W Teh. Learning to Parse Images. In S. A. Solla, T. K. Leen, and K. R. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 463–469. MIT Press, 2000.

[21] W. W. Irving, P. W. Fieguth, and A. S. Willsky. An Overlapping Tree Approach to Multiscale Stochastic Modeling and Estimation. *IEEE Transactions on Image Processing*, 6(11):1517–1529, November 1997.

[22] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An Introduction to Variational

Methods For Graphical Models. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 105–161. Kluwer Academic Publishers, 1998.

[23] S. L. Lauritzen. *Graphical Models.* Oxford University Press, 1996.

[24] R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data.* John Wiley, New York, USA, 1987.

[25] M. R. Luettgen and A. S. Willsky. Likelihood Calculation for a Class of Multiscale Stochastic Models, with Application to Texture Discrimination. *IEEE Transactions on Image Processing*, 4(2):194–207, February 1995.

[26] A. Montanvert, P. Meer, and A. Rosenfeld. Hierarchical Image Analysis Using Irregular Tessellations. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(4):307–316, April 1991.

[27] J. Pearl. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufman Publishers Inc., San Francisco, USA, 2nd edition, 1988.

[28] 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.

[29] A. J. Storkey and C. K. I. Williams. Dynamic Positional Trees for Structural Image Analysis. In *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics.* Morgan Kaufmann, 2001.

[30] C. von der Malsburg. The correlation theory of brain function. Internal Report 81-2, Max-Planck-Institut für Biophysikalische Chemie, 1981. Reprinted in *Models of Neural Networks*, eds. K. Schulten and H.-J. van Hemmen, 2nd. ed, Springer, 1994.

[31] C. von der Malsburg. Dynamic link architecture. In M. A. Arbib, editor, *Handbook of Brain Theory and Neural Networks*, pages 329–331. MIT Press, 1995.

[32] C. K. I. Williams. Using Deterministic Boltzmann machines for discriminating temporally distorted strings, 1990. MSc thesis, Dept. of Computer Science, University of Toronto.

[33] C. K. I. Williams and N. J. Adams. DTs: Dynamic Trees. In M. J. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 634–640. MIT Press, 1999.