To appear in Proc. ICANN 99

Tree-structured Belief Networks as Models of Images

Christopher K. I. Williams Xiaojuan Feng*
Institute for Adaptive and Neural Computation
Division of Informatics, University of Edinburgh
5 Forrest Hill, Edinburgh EH1 2QL.
ckiw@dai.ed.ac.uk xiaojuan@dai.ed.ac.uk

Abstract

In this paper we are concerned with using a tree-structured belief network (TSBN) as a prior model in segmenting a natural image into a number of predefined classes. The TSBN was trained using the EM algorithm based on a set of training label images. The average log likelihood (or bit rate) of a test set of images shows that the learned TSBN is a better model for images than models based on independent blocks of varying sizes. We also analyze the relative advantages obtained by modelling correlations at different length scales in the tree.

1 Introduction

We are concerned with the problem of segmenting an image into a number of predefined classes. This requires fusing together local predictions for the class labels with a prior model of segmentations; here we investigate the use of tree-structured belief networks (TSBNs) as prior models which can capture the local and spatial structure of images. The use of this kind of prior knowledge would help, for example, when pixelwise predictions are locally ambiguous, but the correct answer can be inferred using contextual information.

One commonly-used alternative to the TSBN prior is a Markov Random Field (MRF) model (see, e.g. Geman and Geman, 1984), where at each pixel the random variable would be the choice of label. However, there are some problems with MRF models, particularly that inference procedures are NP-hard. In comparison, for TSBNs inference can be carried out in time linear in the number of pixels using Pearl's message-passing scheme [Pearl, 1988]. Also, with MRFs it can be difficult to incorporate longer-range information into the prior if small neighbourhood sizes are used, in contrast to the multi-scale nature of TSBNs. Although there has been a considerable amount of other work on statistical models of images (e.g. Luettgen and Willsky, 1995, for a TSBN model using Gaussian random variables), this mostly concerns models for intensity levels rather than class labels, and so is therefore not directly applicable to our problem.

The structure of the paper is as follows: In section 2 we describe the basic TSBN architecture, and how to train it. Section 3 describes the assessment of the quality of the TSBN model in terms of coding cost, and demonstrates that the TSBN model is superior to block-based coding methods. In section 4 we explain how these models can be used to segment images, and show some results.

2 Tree-structured belief network

2.1 Generative model

Our model for the data is illustrated in Figure 1. The observed data $\boldsymbol{Y}$ (e.g. the rgb values of the pixels) is assumed to have been generated from an underlying process $\boldsymbol{X}$. $\boldsymbol{X}$ is a tree-structured belief network. At the highest level (level 0) there is one node X^0, which has children in level 1. Typically in our experiments each parent node has four children, giving rise to a quadtree-type TSBN architecture as used by Bouman &

* Current address: Informatics Laboratory, National Institute for Biological Standards and Control, Blanche Lane, South Mimms, Potters Bar, Herts. EN6 3QG, UK. Email: xfeng@nibsc.ac.uk

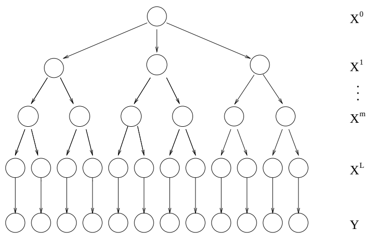

Figure 1: A 1-D graphical model illustrating a small tree-structured belief network. The network nodes are partitioned into three categories: $\boldsymbol{Y}$ denotes the raw image information; $\boldsymbol{X}^L$ denotes the leaf nodes of $\boldsymbol{X}$ which are observed during training. The nodes in the layers above labelled $X^0, \ldots, X^m$ are always unobserved.

Shapiro (1994) . Each X-node is a multinomial variable, taking on one of C class labels. These labels are those used for the segmentation, e.g. road, sky, vehicle etc. The links between the nodes are defined by conditional probability tables (CPTs).

At the lowest level L of the tree, we find the leaf nodes denoted $\boldsymbol{X}^L$. The ith leaf node is denoted X_i^L. The leaf nodes correspond to small regions of the image (in our case 4×4 pixel regions). The model for the observation Y_i in each region is that it is generated according to $P(Y_i|\boldsymbol{X}) = P(Y_i|X_i^L)$, i.e. that Y_i depends only on the corresponding leaf node X_i^L. In addition we assume that the distribution $P(Y_i|X_i^L)$ is independent of i. The overall model for $P(\boldsymbol{X}, \boldsymbol{Y})$ is

$$P(\boldsymbol{X}, \boldsymbol{Y}) = P(\boldsymbol{X}) \prod_{i=1}^{K} P(Y_i|X_i^L).$$

2.2 Inference (Segmentation)

Given a new image $\boldsymbol{Y} = \boldsymbol{y}$ we wish to carry out inference on $\boldsymbol{X}^L$, given the probabilistic model. Computing the posterior $P(\boldsymbol{X}^L = \boldsymbol{x}^L|\boldsymbol{Y} = \boldsymbol{y})$ would be highly expensive, as it would require enumerating all possible C^K states in $\boldsymbol{X}^L$, where K is the total number of pixels in an image. There are two alternatives that are computationally feasible, (i) the computation of the posterior marginals $P(X_i^L = x_i^L|\boldsymbol{Y} = \boldsymbol{y})$, giving rise to inference based on the maximum posterior marginals (MPM) and (ii) the overall most likely interpretation of the data $\boldsymbol{x}^* = \mathrm{argmax}_{\boldsymbol{x}} P(\boldsymbol{X} = \boldsymbol{x}|\boldsymbol{Y} = \boldsymbol{y})$, the *maximum a posteriori* (MAP) configuration. These can be achieved by Pearl's message passing schemes, as described in [Pearl, 1988]. The scheme for (i) defines two vectors $\lambda(z) = P(\boldsymbol{e}_z^-|z)$ and $\pi(z) = P(z|\boldsymbol{e}_z^+)$ for each node z in the tree, where $\boldsymbol{e}_z^-$ is the instantiated nodes below node z in the tree, and $\boldsymbol{e}_z^+$ denotes the remaining instantiated nodes in the tree. These can be computed recursively in sweeps up and down the tree.

Computations (i) and (ii) above require a generative model for $P(Y_i|X_i^L)$. An alternative is to make use of predictions giving $\hat{P}(X_i^L|Y_i)$, as may be obtained from a neural network or some other classifier. As $P(Y_i|X_i^L) = P(X_i^L|Y_i)P(Y_i)/P(X_i^L)$ and $P(Y_i)$ is fixed when performing inference, we can define the *scaled likelihood* for location i as $L(Y_i) = P(X_i^L|Y_i)/P(X_i^L)$. To make use of our predictor we replace $L(Y_i)$ with $\hat{L}(Y_i) = \hat{P}(X_i^L|Y_i)/\hat{P}(X_i^L)$, where $\hat{P}$ denotes an estimated probability. $\hat{P}(X_i^L)$ can be obtained in a number of ways, e.g. from the (smoothed) frequency of each class at pixel i in a set of training images. This method of combining neural networks with belief networks has been suggested in [Morgan and Bourlard, 1995] and [Smyth, 1994]. A potential advantage of using the scaled-likelihood method is that the generative model for $P(Y_i|X_i)$ may be quite complex, although the predictive distribution $P(X_i|Y_i)$ is actually quite simple, i.e. the generative approach may spend a lot of resources on modelling $P(Y_i|X_i)$ which are not particularly relevant to the task of inferring $\boldsymbol{X}$.

2.3 Training a tree-structured belief network

Above it was assumed that the CPTs (denoted $\boldsymbol{\theta}$) used to define $P(\boldsymbol{X})$ are known. In fact we estimated these from training data, using a maximum likelihood procedure. Let x_{ik}, $k = 1, \ldots, C$ denote the possible values of X_i, and let pa_{ij}, $j = 1, \ldots, C$ denote the set of possible values taken on by Pa_i, the parent of X_i. The parameter θ_{ijk} denotes the CPT entry $P(X_i = x_{ik}|Pa_i = pa_{ij})$.

We can obtain a maximum likelihood estimate of $\boldsymbol{\theta}$ by using the EM algorithm. (This is a generalisation of the EM algorithm for Hidden Markov Models to tree-structured networks; see, e.g. Castillo *et al* , 1997). Let the segmented training images be denoted by $\boldsymbol{x}^{Lm}$, $m = 1, \ldots, M$. Then the update for each CPT is given by

$$\tilde{\theta}_{ijk} = \frac{\sum_{m=1}^{M} P(X_i = x_{ik}, Pa_i = pa_{ij} | \boldsymbol{x}^{Lm}, \boldsymbol{\theta})}{\sum_{m=1}^{M} \sum_{k'} P(X_i = x_{ik'}, Pa_i = pa_{ij} | \boldsymbol{x}^{Lm}, \boldsymbol{\theta})}.$$

The joint probability $P(X_i = x_{ik}, Pa_i = pa_{ij} | \boldsymbol{x}^{Lm}, \boldsymbol{\theta})$ can be obtained locally using

$$P(X_i = x_{ik}, Pa_i = pa_{ij} | \boldsymbol{x}^{Lm}, \boldsymbol{\theta}) = \frac{\lambda(x_{ik}|\boldsymbol{\theta})\ \theta_{ijk}\ \pi(pa_{ij}|\boldsymbol{\theta}) \prod_{y \in s(X_i)} \lambda_y(pa_{ij}|\boldsymbol{\theta})}{\sum_{k'} \pi(pa_{ik'})\lambda(pa(ik'))},$$

where we have used $\lambda(x_{ik}|\boldsymbol{\theta})$ for $\lambda(X_i = x_{ik}|\boldsymbol{\theta})$, similarly for Pa_i. $s(X_i)$ is the set of nodes that are siblings of node X_i. $\lambda_y(.)$ is the λ-message sent to node Pa_i by node y.

This update gives a separate update for each link in the tree. Given limited training data this is undesirable. If the set of variables sharing a CPT is denoted as $\boldsymbol{X}_I$, then the update for $\tilde{\theta}_{Ijk}$ is as above for $\tilde{\theta}_{ijk}$ except that the summation over m is replaced by a double summation over m and $X_i \in \boldsymbol{X}_I$ in both the denominator and numerator. In our experiments all of the CPTs in each level were constrained to be equal, except for the transition from layer 0 to layer 1, where each table was separate. The layer 0 to layer 1 CPTs in the trained model were found to produce high marginal probabilities for the "sky" label in the top half of the image, and for the "vegetation", "road marking" and "road surface" labels for the bottom half of the image. Below that, the CPTs were strongly diagonal in the main, indicating a preference for the child to have the same label as the parent.

3 Assessing the TSBN model: Coding cost

The Sowerby Image Database that we are using for our experiments provides both rgb and label images. This means that we can evaluate a model for the label process by evaluating the likelihood of a test set of label images under the model. By calculating $-\log_2 P(\boldsymbol{X}^L)/(\#\text{labelled pixels in the image})$ we obtain the coding cost in bits/pixel. The minimum attainable coding cost is the entropy (in bits/pixel) of the generating process. As the computation of $P(\boldsymbol{x}^L)$ is intractable in MRF models, we compare the TSBN results to those derived by coding blocks of the image independently.

3.1 Coding cost under TSBN model

By using a TSBN to model the distribution of images, the marginal likelihood of a label image $\boldsymbol{x}^L$ can be calculated efficiently at the root node of the tree, giving

$$P_{TSBN}(\boldsymbol{x}^L) = \sum_{x^0} \lambda(x^0)\pi(x^0), \qquad (1)$$

where the summation is extended over all values in X^0. In practice the dynamic range of the λ-value computation will exceed the precision range of essentially any machine; this is overcome by using a scaling procedure, where the λ-values at each node are normalized and the scaled λ-values are passed up the tree.

Below we will also consider the effect of truncating the tree at a level below the root of the tree. In this case, instead of having one large tree, the image model consists of a number of smaller trees (and correlations between the different trees are then ignored). This allows us to quantify the benefits of using the higher levels in the tree, which correspond to longer-range correlations. The priors for each of the smaller trees are calculated by propagating the prior $\pi(x^0)$ through the downward CPTs to obtain a prior for each root. The likelihood of an image under the truncated model is simply the product of terms like equation 1 evaluated at every root.

3.2 A block coding model

Under this model, an image is divided into blocks of size $N \times M$ pixels. There is a probability model within each block, but each block is assumed to be coded independently. The probability of image $\boldsymbol{x}^L$ under this model is given by

$$P_{block}(\boldsymbol{x}^L) = \prod_{i=1}^{k_b} P(\boldsymbol{x}_{bi}^L) = \prod_{j=1}^{c_b} [P(c_{bj})]^{n_{bj}}, \qquad (2)$$

where k_b is the total number of blocks in an image and $\boldsymbol{x}_{bi}^L$ denotes the ith block. $c_b = C^{NM}$ is the total number of block-labelling patterns and n_{bj} is the total number of blocks of pattern C_{bj} in image $\boldsymbol{x}^L$.

In order to use this scheme we have to estimate $P(c_{bj})$, the probability of the jth pattern, for all j. The obvious way to do this is to compute the frequencies of each pattern in the training data—this is the maximum likelihood estimator. This method works well when the block size is small (e.g. 1×1), but rapidly runs into problems even for 2×2 blocks due to the sparse data problem—some patterns never occur in the training data. This problem occurs in a number of fields (e.g. language modelling) and several solutions have been proposed. We have used one of the simplest, adding a prior amount ϵ to each count before the data is observed, so that $\hat{P}(c_{bj}) = (n_{bj} + \epsilon)/\sum_k (n_{bk} + \epsilon)$.

The simplest block coding schemes use the same probabilities for each block. However, in order to make a fairer comparison against the TSBN method, we also investigated a scheme where the coding probabilities can vary from location to location in the image. This is inspired by the fact that in our TSBNs the marginal probabilities of nodes in the $\boldsymbol{X}^L$ layer depend on spatial location. More precisely, there are six different sets of marginal probabilities in the six square regions defined by a 2×3 grid. These derive from the fact that the TSBN parameters at the highest level are not tied.

In the labelled images, some of the pixels have undefined labels. We treat these as missing at random, and therefore compute the probability for all the labelled pixels, and divide by that number. Under the block coding scheme, we call blocks missing one or more pixels partial blocks. The probability of a partial block can then be obtained by marginalizing over the undefined pixels.

3.3 Experimental results

Colour images of out-door scenes from the Sowerby Image Database of British Aerospace are used in our experiments. Both colour images and their corresponding label images are provided in this database. The original 104 images were divided randomly into independent training and test sets of size 61 and 43 respectively. We describe the results of experiments using the TSBN and block coding methods on the label images. There are 7 different labels in all, namely "sky", "vegetation", "road markings", "road surface", "building", "street furniture" and "mobile object".

The original label images of size 512 by 768 pixels were subsampled into 128 by 192 regions to form the reduced label images. The label of the reduced region was chosen by majority vote, with ties being resolved by an ordering on the label categories. The performance of coding models was then tested using the reduced label images. From now on we will refer to the reduced label images as label images because the original label images will no longer be used.

If we have an infinite amount of training data, the larger the block size the better the coding should be. However, with finite amounts of training data there is a trade-off between estimating the probabilities accurately and block size. We tried blocks with sizes of 1×1, 2×2, 2×4, 4×4, 4×8, 8×8 and 16×16. The values of the smoothing parameter ϵ were optimized on the test data (giving the block coding methods a slight advantage over the TSBN architecture)[1].

The belief network structure was basically a quadtree, except that there were six children of root node to take into account the $2 : 3$ aspect ratio of the images. A tree-structured belief network with 8 levels was trained using the training set of label images. In the training phase, the network parameters $\boldsymbol{\theta}$ were initialized in a number of different ways. It was found that the highest marginal likelihood of the training data was obtained when the initial values of $\boldsymbol{\theta}$ were computed using probabilities derived from downsampled versions of the images. A plot of the marginal likelihood against iteration number on the training set levelled off after 50 iterations. The upward (λ) propagation in the tree takes around 10s, and the downward (π) propagation around 40s on a SGI R10000 processor; the tree has over 30,000 nodes.

The results of the experiments are shown in Table 1 . The minimum average bit rate for the block models was 0.3336 bpp for the 4×4 block, but this was still larger than the TSBN bit rate of 0.2307 bpp. This suggests that the longer-range correlations that can be encoded in the TSBN are important. For a rough comparison, the label images could be compressed with Lempel-Ziv coding using the Unix utility `gzip` to 0.398 bpp.

Using the "truncated tree" scheme dis-

[1] Although there are a huge number of possible patterns for the larger block sizes, ϵ can be computed as one only needs to record the frequencies of the relatively small number of patterns that actually appear in the data.

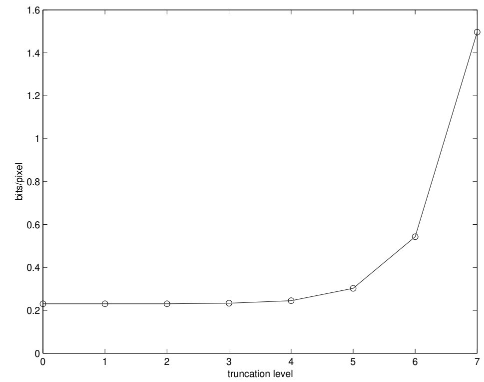

Figure 2: Bit rate (bits/pixel) as a function of the truncation level in the TSBN.

Table 1: Bit rates for the TSBN and block-coding methods on the 43 test images. For the block methods larger than 1×1, the optimal ϵ is also given.

Method	Bit rate bits/pixel	Optimal ϵ
TSBN	0.2307	
1×1	1.8768	
1×1 (6 regions)	1.4676	
2×2	0.6142	10^{-2}
2×4	0.3887	10^{-4}
4×4	0.3336	10^{-10}
4×8	0.4540	10^{-23}
8×8	0.6861	10^{-50}
16×16	1.2650	10^{-212}

cussed in section 3.1, we can analyze the TSBN results further. Figure 2 shows the bit rate (in bits/pixel) evaluated as a function of truncating the tree at levels 0 to 7. These figures interpolate between the 1×1 (6 regions) value (which is equivalent to truncation at level 7), and the TSBN bit rate. It can be seen that by the time level 4 has been reached, almost all of the benefit has been attained. This corresponds to a 8×8 block size, but using a learned, structured TSBN model for the probabilities in that block rather than an unstructured block model.

4 Segmentation experiments

We have used our models of label images along with neural networks that make label predictions at each image location to obtain segmentations of the test images, using the scaled likelihood method (see section 2.2). A multi-layer perceptron network with one hidden layer was trained to predict the posterior probabilities of the class labels using features based on properties such as the local intensity, colour and texture. Further details on the network training can be found in [Williams and Feng, 1998]. The scaled likelihoods were then fed to the TSBN to obtain the MAP configuration.

For the 4×4 blocks, there are 7^{16} possible configurations to be checked in each block. To avoid this astronomical amount of work, we only checked the 5139 configurations that occurred in the training set. Figure 3 shows the segmentations of one particular image from the test set. The overall classification accuracy on the test set was 83.5% using the 4×4 block coding method, and 87.1% using the TSBN.

5 Discussion

Our main contribution in this paper is to have evaluated the quality of TSBNs as models of label-images. We have shown that the TSBN is superior to block coding schemes trained on the same dataset. The truncated tree analysis of TSBN coding shows that for this data almost all of the coding gain can be achieved by going up to level 4 of the tree, corresponding to a 8×8 block size.

While TSBNs have many desirable properties, they are rather rigid models; for example, their architecture means that segmentation results are not translation invariant. Current work on overcoming these limitations focuses on creating priors over TSBN architectures [Williams and Adams, 1999] or by adding extra linkages to the TSBN architecture.

Acknowledgements

This work is funded by EPSRC grant GR/L03088, *Combining Spatially Distributed Predictions From Neural Networks.* The authors gratefully acknowledge the assistance of British Aerospace in the project and in making the Sowerby Image Database available to us. They also thank the Isaac Newton Institute (Cambridge, UK) for its hospitality and excellent working environment during the "Neural Networks and Machine Learning" programme, 1997, Ian Nabney for NETLAB routines and

Dr. John Elgy for introducing us to the work of [Bouman and Shapiro, 1994].

References

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

[Castillo et al., 1997] Castillo, E., Gutiérrez, J. M., and Hadi, A. S. (1997). *Expert Systems and Probabilistic Network Models*. Springer, New York.

[Geman and Geman, 1984] Geman, S. and Geman, D. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Trans. Pattern Analysis and Machine Intellligence*, 6(6):721–741.

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

[Morgan and Bourlard, 1995] Morgan, N. and Bourlard, H. A. (1995). Neural Networks for Statistical Recognition of Continuous Speech. *Proceedings of the IEEE*, 83(5):742–770.

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

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

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

[Williams and Feng, 1998] Williams, C. K. I. and Feng, X. (1998). Combining Neural Networks and Belief Networks for Image Segmentation. In Constantinides, T., Kung, S.-Y., Niranjan, M., and Wilson, E., editors, *Neural Networks for Signal Processing VIII*. IEEE.

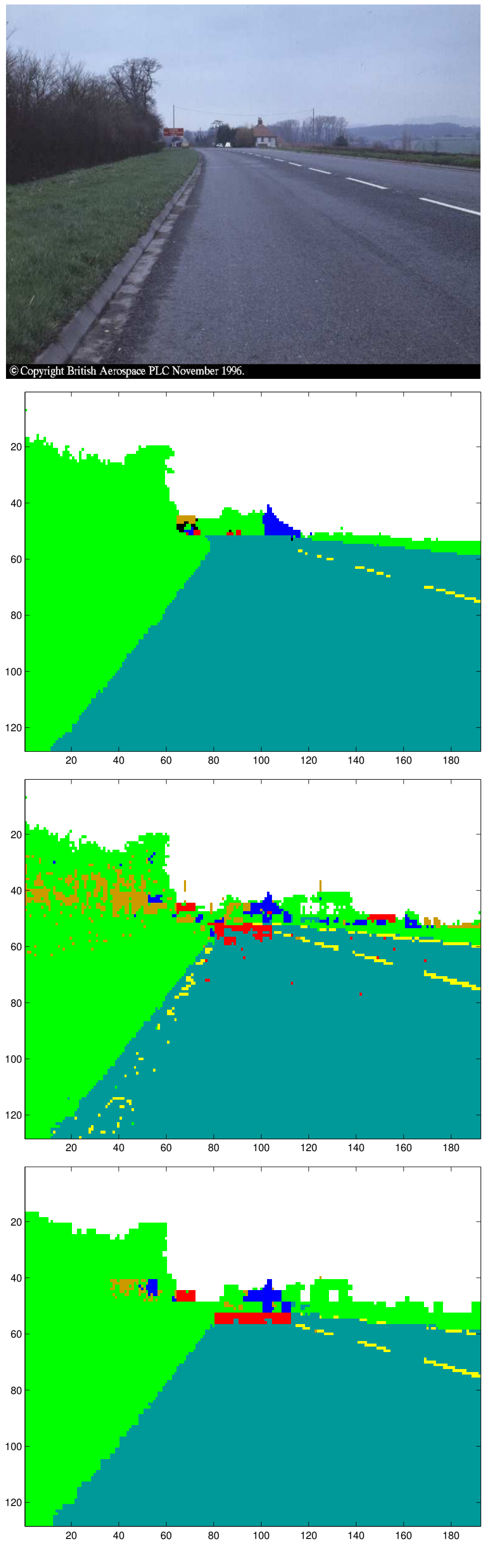

Figure 3: From the top: The original image, true label image, labelling by block method, labelling by TSBN method.