Constructive Algorithms for Hierarchical Mixtures of Experts

S.R.Waterhouse **A.J.Robinson**
Cambridge University Engineering Department,
Trumpington St., Cambridge, CB2 1PZ, England.
Tel: [+44] 1223 332754, Fax: [+44] 1223 332662,
Email: srw1001, ajr @eng.cam.ac.uk

ABSTRACT

We present two additions to the hierarchical mixture of experts (HME) architecture. We view the HME as a tree structured classifier. Firstly, by applying a likelihood splitting criteria to each expert in the HME we "grow" the tree adaptively during training. Secondly, by considering only the most probable path through the tree we may "prune" branches away, either temporarily, or permanently if they become redundant. We demonstrate results for the growing and pruning algorithms which show significant speed ups and more efficient use of parameters over the conventional algorithms in discriminating between two interlocking spirals and classifying 8-bit parity patterns.

INTRODUCTION

The HME (Jordan & Jacobs 1994) is a tree structured network whose terminal nodes are simple function approximators in the case of regression or classifiers in the case of classification. The outputs of the terminal nodes or experts are recursively combined upwards towards the root node, to form the overall output of the network, by "gates" which are situated at the non-terminal nodes.

The HME has obvious similarities with model merging techniques such as stacked regression (Wolpert 1993) in which explicit partitions of the training set are combined. However the HME differs from model merging in that each expert considers the whole input space in forming its output. Whilst this allows the network more flexibility since each gate may implicitly partition the whole input space in a "soft" manner, it leads to unnecessarily long computation in the case of near optimally trained models. At any one time only a few paths through a large network may have high probability. In order to overcome this drawback, we introduce the idea of "path pruning" which considers only those paths from the root node which have probability greater than a certain threshold.

The HME also has similarities with tree based statistical methods such as Classification and Regression Trees (CART) (Breiman, Friedman, Olshen & Stone 1984). We may consider the gate as replacing the set of "questions" which are asked at each branch of CART. From this analogy, we may consider the application of the splitting rules used to build CART. We start with a simple tree consisting of two experts and one gate. After partially training this simple tree we apply the split-

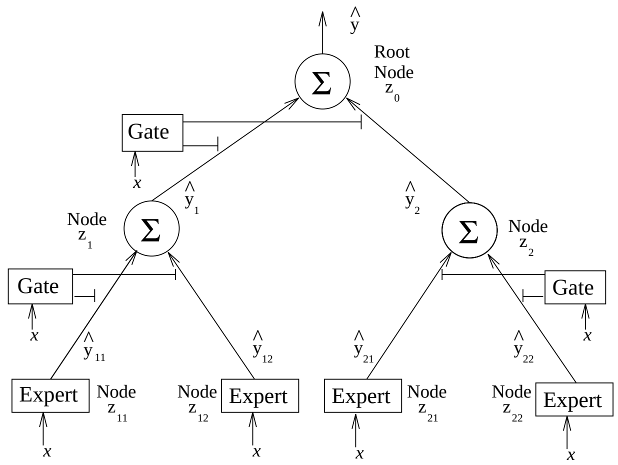


Figure 1: THE HIERARCHICAL MIXTURE OF EXPERTS

ting criterion to each terminal node. This evaluates the log-likelihood increase by splitting each expert into two experts and a gate. The split which yields the best increase in log-likelihood is then added permanently to the tree. This process of training followed by growing continues until the desired modelling power is reached. This approach is reminiscent of Cascade Correlation (Fahlman & Lebiere 1990) in which new hidden nodes are added to a multi-layer perceptron and trained while the rest of the network is kept fixed.

HIERARCHICAL MIXTURES OF EXPERTS

The HME is a supervised feedforward network which may be used for classification or regression. The architecture of the basic HME is fixed during training and evaluation. Figure 1 shows an HME with 2 branches at each node and with a depth of 2. We will refer to an HME with constant branching factors of 2 at each node as a "binary branching HME". Each node z_i consists of a parameter vector η_i. Terminal nodes also contain a parameter vector (or matrix in the case of multiple outputs) β_i. We shall denote the general set of parameters for node z_i as $\boldsymbol{\theta_i}$ and the total set of parameters for the HME as $\boldsymbol{\Theta}$.

Given a set of input:output pairs $\{\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}\}_{t=1}^{T}$, the network is trained to maximise the log likelihood $L = \sum_t \log P(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t)}, \boldsymbol{\Theta})$. The overall output of the network $\hat{\boldsymbol{y}}$ is the weighted sum of outputs $\hat{\boldsymbol{y}}_i$ from the daughter nodes z_i and the probabilities of these daughter nodes given the input and the parameters of the daughter node, $P(z_i|\boldsymbol{x}, \boldsymbol{\eta_i})$. The outputs of the daughter nodes are in turn generated by the weighted sum of outputs $\hat{\boldsymbol{y}}_{ij}$ of their daughter nodes, z_{ij} and the probabilities of these daughters $P(z_{ij}|z_j, \boldsymbol{x}^{(t)}, \boldsymbol{\eta_{ij}})$, so that the overall output at time t is given by,

$$\hat{\boldsymbol{y}} = \sum_j P(z_j|\boldsymbol{x}, \boldsymbol{\eta_i}) \sum_i P(z_{ij}|z_j, \boldsymbol{x}, \boldsymbol{\eta_{ij}}) \hat{\boldsymbol{y}}_{ij}. \tag{1}$$

Each non-terminal node z_i consists of a "gate" whose outputs are the conditional probabilities of the daughters of this node given the input, the parameter vectors of the daughter nodes $\boldsymbol{\eta_{ij}}$ and the path R_i taken to that node from the root node.

This conditional probability is computed using the Softmax function (Bridle 1989),

$$P(z_{ij}|\boldsymbol{x}, \eta_{\boldsymbol{ij}}) = \exp(\eta_{\boldsymbol{ij}}{}^T \boldsymbol{x}) \Bigg/ \sum_k \exp(\eta_{\boldsymbol{ik}}{}^T \boldsymbol{x}) \ ,$$

where the sum is over all daughters of the non-terminal node z_i

Each terminal node z_{ij} consists of an "expert" whose outputs are the conditional expected value of the correct output given the input and its parameters $\beta_{\boldsymbol{ij}}$. In the case of classification experts, the outputs are the conditional probabilities of different classes. These are also obtained via a softmax function. Given this framework, we may consider the HME as a probabilistic model of the observed data, such that the probability of the correct output is the weighted sum of probabilities from daughter nodes and the probabilities of these daughter nodes,

$$P(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\Theta}) = \sum_i P(z_i|\boldsymbol{x}, \eta_{\boldsymbol{i}}) \sum_j P(z_{ij}|z_i, \boldsymbol{x}, \eta_{\boldsymbol{ij}}) P(\boldsymbol{y}|z_i, z_{ij}, \boldsymbol{x}, \beta_{\boldsymbol{ij}}).$$

This is a finite mixture model in which the mixture weights are the gate outputs, and the mixture coefficients are the probabilities of the daughter nodes generating the correct output. By taking the expected value of this expression we obtain equation (1). This confirms that given this probability model, the output of the network is the conditional expected value of the correct output given the input and the parameters of the network $\boldsymbol{\Theta}$.

A common method for training mixture models is the Expectation Maximisation (EM) algorithm (Dempster, Laird & Rubin 1977). When applied to the HME (Jordan & Jacobs 1994), EM de-couples the maximum likelihood problem into a series of independent ML problems for each gate and expert. The E step reduces to computing the set of conditional *posterior* probabilities of all nodes, given the input and output data. For node z_{ij} the posterior probability h_{ij} is:

$$h_{ij} \equiv P(z_{ij}|z_i, \boldsymbol{x}, \boldsymbol{y}, \eta_{\boldsymbol{ij}}) = \frac{P(z_{ij}|\boldsymbol{x}, z_i, \eta_{\boldsymbol{ij}}) P(\boldsymbol{y}|\boldsymbol{x}, z_{ij}, z_i, \beta_{\boldsymbol{ij}})}{\sum_k P(z_{ik}|\boldsymbol{x}, z_i, \eta_{\boldsymbol{ik}}) P(\boldsymbol{y}|\boldsymbol{x}, z_{ik}, z_i, \beta_{\boldsymbol{ik}})}.$$

The M step reduces to maximising the set of independent weighted maximum likelihood problems for each node in the HME.

TREE GROWING

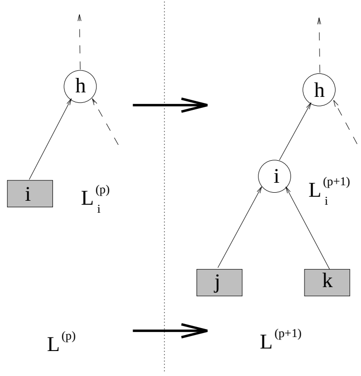
The standard HME differs from most tree based statistical models in that its architecture is fixed. By relaxing this constraint and allowing the tree to grow, we achieve a greater degree of flexibility in the network. Following the work on Classification and Regression Trees (Breiman et al. 1984) we start with a simple tree, for instance with two experts and one gate which we train for a small number of cycles. Given this semi-trained network, we then make a set of candidate splits $\mathcal{S}_i$ of terminal nodes z_i. We wish to select eventually only the "best" split $\tilde{\mathcal{S}}$ out of these candidate splits. Let us define the best split as being that which maximises the increase in log-likelihood due to the split, $\Delta L = L^{(p+1)} - L^{(p)}$ where $L^{(p)}$ is the likelihood at the p^{th} *generation* of the tree. If we make the constraint that the parameters of the rest of the tree remain unchanged when a candidate split is made, then the maximisation is simplified into a dependency on the increase in the likelihood L_i of node i:

$$\max_i \Delta L(\mathcal{S}_i) \quad \equiv \quad \max_i \Delta L_i \tag{2}$$

$$\text{where} \qquad L_i^{(p)} \quad = \quad \sum_t \log P(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t)}, z_i, \beta_i) \tag{3}$$

$$L_i^{(p+1)} = \sum_t \log \sum_j P(z_{ij}|\boldsymbol{x}^{(t)}, \eta_i, z_i) P(\boldsymbol{y}^{(t)}|\boldsymbol{x}^{(t)}, z_{ij}, \beta_{ij}) \quad (4)$$

This splitting rule is similar in form to the CART splitting criterion which uses maximisation of the entropy of the node, equivalent to our local increase in log-likelihood.



Given a tree $H^{(p)}$ of generation p,

Freeze all non-terminal nodes,

Split all terminal nodes z_i into two experts and a gate, initialising the new experts and gate with random weights,

Evaluate ΔL_i for all z_i by training using the standard EM method, but without making updates to the frozen parameters.

Select the best candidate split $\tilde{S}$. Make this split permanent and discard all other candidate splits.

Thaw the rest of the tree to give the new tree $H^{(p+1)}$ and re-train using the standard EM method until no further increase in likelihood is made.

Figure 2: Tree growing algorithm

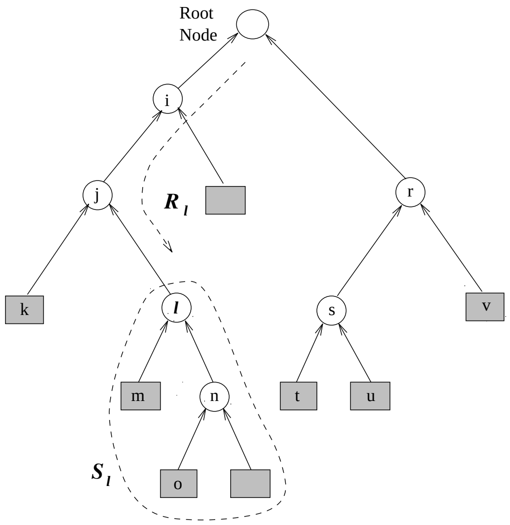
The splitting criterion leads to the algorithm shown in Figure 2. This algorithm was used to solve the 8-bit parity classification task. As can be seen in Figures 4(a) and (b), the factorisation enabled by the freezing algorithm significantly speeds up computation over the standard EM method. The final tree shape is also shown in Figure 4(c). We showed in an earlier paper (Waterhouse & Robinson 1994) that the XOR problem may be solved using at least 2 experts and a gate. The 8 bit parity problem is therefore being solved by a series of XOR classifiers, each gated by its parent node, which is an intuitively appealing form with an efficient use of parameters.

The algorithm is sensitive to the initialisation of the parameters of the candidate splits. These are"cloned" from the original terminal node parameters with additional random perturbation. The gates are set to small random values to break symmetry.

PATH PRUNING

If we consider the HME to be a good model for the data generation process, the case for path pruning becomes clear. In a tree with sufficient depth to model the underlying sub-processes producing each data point, we would expect the activation of each expert to tend to binary values such that only one expert is selected at each time exemplar.

This suggest a path pruning scheme within the fixed HME architecture which utilises the "activation" of a node at each time step, which is the product of node probabilities along a path from the root node to the current node. We may prune paths through the tree either during evaluation or training. A path is pruned when its current activation falls below a threshold. No further descent of the tree is then made, although the same branch may be used at the next time step, with different paths being pruned instead.



During Training:
1. Descend tree, computing $J_i^{(t)} \sum_i \log P(z_i^{(t)}|R_i, \boldsymbol{x}^{(t)})$ over all nodes z_i, where R_i is the path taken to i from the root node z_0. 2. If $J_l^{(t)} < f_t$, where f_t is the pruning threshold used in training, then do not accumulate statistics of subtree S_l and backtrack up to node z_j, the parent node of z_l. **During evaluation:**
1. Descend tree, computing J_i for all nodes z_i as before, 2. If $J_l^{(t)} < f_e$, where f_e is the pruning threshold used in evaluation, ignore the subtree S_l and backtrack up to node z_j.

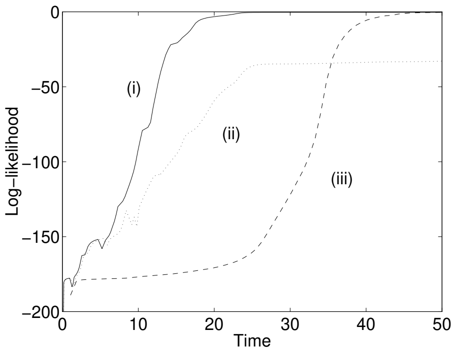
Figure 3: Path pruning algorithm

The path pruning algorithm is described in Figure 3. Figure 5 shows the application of the pruning algorithm to the task of discriminating between two interlocking spirals. With no pruning the solution to the two-spirals takes over 4,000 CPU seconds, whereas with pruning the solution is achieved in 155 CPU seconds.

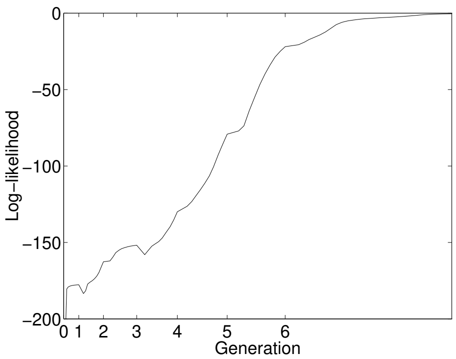
One problem which we encountered when implementing this algorithm was in computing updates for the parameters of the tree in the case of high pruning thresholds. If a node is visited too few times during a training pass, it will sometimes have too little data to form reliable statistics and thus the new parameter values may be unreliable and lead to instability. This is particularly likely when the gates are saturated. As described in Waterhouse, MacKay & Robinson (1995), the use of regularization can help prevent this over-fitting of the gates and this practice stabilises the pruning algorithm significantly. Some problems of instability are still evident, however, when very high pruning thresholds are used. In general, we find that pruning thresholds of less than 0.01 are generally stable, but this threshold also seems to be a function of the depth of the tree.

CONCLUSIONS

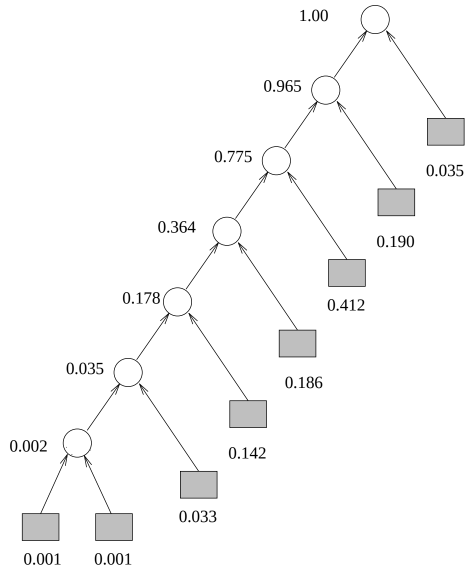
We have presented two extensions to the standard HME architecture. By pruning branches either during training or evaluation we may significantly reduce the computational requirements of the HME. By applying tree growing we allow greater flexibility in the HME which results in faster training and more efficient use of parameters.



(a) Evolution of log-likelihood vs. time in CPU seconds.



(b) Evolution of log-likelihood for (i) vs generations of tree.



(c) Final tree structure obtained from (i), showing utilisation U_i of each node where $U_i = \sum_t P(z_i^{(t)}, R_i | \boldsymbol{x}^{(t)}) / T$.

Figure 4: HME GROWING ON THE 8 BIT PARITY PROBLEM;(i) growing HME with 6 generations; (ii) growing HME without regularization, showing early saturation; (iii) 4 deep binary branching HME (no growing).

References

Breiman, L., Friedman, J., Olshen, R. & Stone, C. J. (1984), *Classification and Regression Trees*, Wadswoth and Brooks/Cole.

Bridle, J. S. (1989), Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition, *in* F. Fougelman-Soulie & J. Hérault, eds, 'Neuro-computing: Algorithms, Architectures and Applicatations', Springer-Verlag, pp. 227–236.

Dempster, A. P., Laird, N. M. & Rubin, D. B. (1977), 'Maximum likelihood from incomplete data via the EM algorithm', *Journal of the Royal Statistical Society, Series B* **39**, 1–38.

Fahlman, S. E. & Lebiere, C. (1990), The Cascade-Correlation learning architecture, Technical Report CMU-CS-90-100, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213.

Jordan, M. I. & Jacobs, R. A. (1994), 'Hierarchical Mixtures of Experts and the EM algorithm', *Neural Computation* **6**, 181–214.

Waterhouse, S. R. & Robinson, A. J. (1994), Classification using hierarchical mixtures of experts, *in* 'IEEE Workshop on Neural Networks for Signal Processing', pp. 177–186.

Waterhouse, S. R., MacKay, D. & Robinson, A. J. (1995), Bayesian inference of hierarchical mixtures of experts, Technical report, Cambridge University Engineering Department.

Wolpert, D. H. (1993), Stacked generalization, Technical Report LA-UR-90-3460, The Santa Fe Institute, 1660 Old Pecos Trail, Suite A, Santa Fe, NM, 87501.

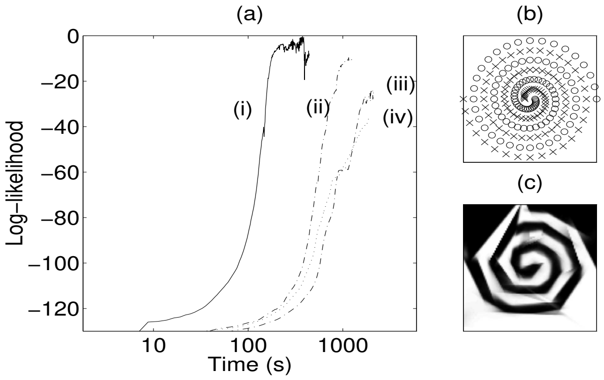


Figure 5: THE EFFECT OF PRUNING ON THE TWO SPIRALS CLASSIFICATION PROBLEM BY A 8 DEEP BINARY BRANCHING HME:(a) Log-likelihood vs. Time (CPU seconds), with log pruning thresholds for experts and gates f: (i) $f = -5.6$,(ii) $f = -10$,(iii) $f = -15$,(iv) no pruning, (b) training set for two-spirals task; the two classes are indicated by crosses and circles, (c) Solution to two spirals problem.