Exploiting Informative Priors for Bayesian Classification and Regression Trees

Nicos Angelopoulos and **James Cussens**
Department of Computer Science
University of York
Heslington, York YO10 5DD, UK
{nicos,jc}@cs.york.ac.uk

Abstract

A general method for defining informative priors on statistical models is presented and applied specifically to the space of classification and regression trees. A Bayesian approach to learning such models from data is taken, with the Metropolis-Hastings algorithm being used to approximately sample from the posterior. By only using proposal distributions closely tied to the prior, acceptance probabilities are easily computable via marginal likelihood ratios, whatever the prior used. Our approach is empirically tested by varying (i) the data, (ii) the prior and (iii) the proposal distribution. A comparison with related work is given.

1 Introduction

A key feature of the Bayesian approach to statistical inference is that prior knowledge (i.e. relevant information distinct from the data) can be incorporated into the learning process in a mathematically rigorous and conceptually clear manner. On the assumption that a space of possible statistical models can be defined, prior knowledge is expressed via a *prior distribution* over this space. This distribution is then updated with the data using Bayes theorem to produce a posterior distribution. The posterior distribution is the end result of learning and can then be used to make predictions about future data or, once suitably summarised, to give the data analyst insight into the domain of interest.

The Bayesian framework is compellingly simple, but there are many complexities in applying it to real data analysis. Many of these difficulties are computational. For example, in high-dimensional spaces the posterior will be too complex to report directly so features of the distribution (e.g. mean and variance) must be extracted—requiring, in general, a difficult integration. These problems are being progressively alleviated by increased computing power and the subsequent use of Markov chain Monte Carlo (MCMC) techniques to approximately sample from the posterior.

In this paper an existing Bayesian method for defining and using a very flexible class of prior distributions over models is further developed and applied specifically to classification and regression tree (C&RT) models. C&RT models are so well known in machine learning (and increasingly statistics) that this paper assumes familiarity with the basic properties of these models.

The method is fully Bayesian using a version of the Metropolis-Hastings MCMC algorithm to approximately sample from posterior distributions over the space of all possible C&RT models. Even after eliminating *a priori* uninteresting trees (for example, those containing sparsely populated leaves) this will be a very large space. The key feature of our approach is that the user can declare structural prior knowledge to restrict and/or bias the prior distribution over this space.

The rest of the paper is structured as follows. Section 2 describes our language for defining priors. Section 3 describes the specific priors that have been used for Bayesian inference of C&RT models. Section 4 presents our version of the Metropolis-Hastings (MH) algorithm and gives a convergence result for one special case. Section 5 gives a representative sample of the results of our experiments. The paper concludes with a comparison with related work (Section 6) followed by pointers to future work (Section 7).

2 Defining priors with stochastic logic programs

Our approach to defining priors is related to a line of research which was independently initiated by [Chipman *et al.*, 1998] and [Denison *et al.*, 1998]. The basic idea is to specify a prior distribution over a space of models with a *stochastic program* rather than by some closed-form expression:

> Instead of specifying a closed-form expression for the tree prior, $p(T)$, we specify $p(T)$ implicitly by a tree-generating stochastic process. Each realization of such a process can simply be considered a random draw from this prior. [Chipman *et al.*, 1998]

A convenient language for defining such stochastic programs is *stochastic logic programming* [Muggleton, 1996]. A stochastic logic program (SLP) can be used to define a prior over a given space of statistical models by a two-step process. Firstly, a standard logic program is written which defines the desired model space (the *hypothesis space* in machine learning parlance). We simply define a unary predicate `cart/1` in a logic program such that each instantiation for `T` which makes `cart(T)` true is a term in first-order logic which represents a C&RT model. We can exploit the expressiveness of first-order logic to represent C&RT models, in whichever logical representation is most convenient.

At this point any C&RT models we wish to exclude from the hypothesis space can be so excluded by adding the appropriate constraints to the logic program. For example, suppose the user is faced with a classification task and knows (or wishes to assume) that the distribution over classes is constant over some rectangular region (a 'box') of the attribute space. It follows that the box must be contained within a leaf of the 'true' tree so only trees with a leaf containing the box need be considered. In general, the user may wish to declare several such boxes. We have effected such constraints in our experiments by declaring facts such as those given in Fig 1, and altering the definition of `cart/1` to check that boxes determined by our declarations are not split in any tree.

```
%const_class_probs(boxname,var,min,max).
const_class_probs(box28,x2,-inf,127).
const_class_probs(box28,x8,-inf,28).

const_class_probs(box26,x2,128,inf).
const_class_probs(box26,x6,-inf,29.8).
```

Figure 1: Declaring rectangular regions within which the distribution over classes is constant.

The stochastic element of the SLP is most easily understood by first considering how the logic programming language Prolog searches for logical consequences of a logic program. Given a *query* such as `:-cart(T)` ("find `T` such that `cart(T)` is true"), Prolog will search for suitable `T`s with a deterministic depth-first search. When there is a choice of rules each of which might lead to a proof, Prolog always chooses whichever rule occurs first in the logic program. If this rule leads to a dead-end, then Prolog *backtracks* and tries the second rule, and so on.

The basic SLP idea is simple: probability labels are attached to rules in the logic program thus converting it into an SLP. Now when the search for proofs has to choose between rules, a rule is chosen probabilistically according to the probability labels. In this paper, if a chosen rule does not lead to a proof, then backtracking will try from amongst untried rules, revisiting the most recent choice-points first and renormalising the probabilities as necessary: we call this approach, introduced by [Cussens, 2000], *backtrackable sampling*.

Now when the query `:-cart(T)` is asked, there is a distribution over possible instantiations for `T`. Since each `T` represents a C&RT model, this provides a prior distribution over this model space.

In this paper we have extended the representational power of SLPs so that probability labels may be computed 'on the fly'. SLP rules can now be of the form `ExpProb : Vars : Rule`. At each point at which `Rule` is a possible rule to use, it is required that `Vars` is a ground list and that `ExpProb` is a probability computable from `Vars`. Fig 2 has an example SLP where the probability of growing a tree by splitting a leaf depends on the depth of that leaf.

SLPs define a distribution over proofs, and to understand this distribution it is useful to represent each proof as a *proof tree*. Fig 3 displays one such proof tree which proves that `cart(x1-[x2-[leaf,leaf],leaf])` is true, i.e. that this 3-leaf tree is in our model space. (Figs 2 and 3 use an abbreviated representation of C&RT models, with split values omitted.) Each proof tree simply consists of the rules and facts used in the proof 'bolted together' by equality constraints. The initial query appears in an equality constraint in the root node. Conceptually, SLPs execute by probabilistically growing a proof tree, backtracking if the constraints become insoluble.

```
1 - 1/D : [D] : cart(leaf).
1/D : [D] : cart(Split-[L,R]) :-
   NewD is D+1, splt(Split),
   [NewD] : cart(L), [NewD] : cart(R).

0.4:splt(x1). 0.3:splt(x2). 0.3:splt(x3).
```

Figure 2: Prior distribution over C&RT models where the probability of splitting a leaf depends on its depth `D`.

To sum up, our priors are defined by a sampling algorithm within a logical framework. The sampling algorithm could, of course, be implemented in any programming language. It involves progressively building up a data structure representing the C&RT model by (i) making choices, (ii) checking that constraints are not violated and (iii) backtracking if they have. The motivation for a logic programming implementation is that procedures (ii) and (iii) are built-in, and that logic is a convenient language for declaring constraints (such as our boxes).

Not all distributions can be directly sampled from (hence MCMC), so such distributions cannot be effectively represented by defining a direct sampling algorithm. However, when a statistical model can be viewed as the end result of some generative process, a generative description is well-motivated. Bayes nets can be generated by adding arcs [Angelopoulos and Cussens, 2001] and C&RT models by splitting leaves. Our addition of backtracking extends the applicability of this generative view.

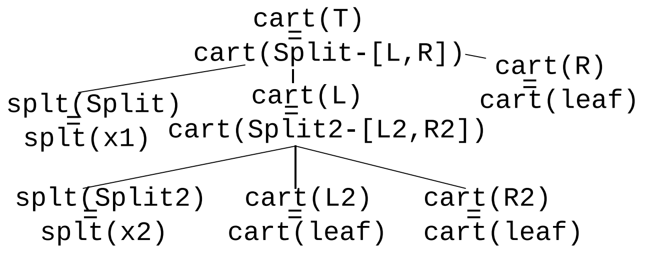

Figure 3: A proof tree which proves `cart(x1-[x2-[leaf,leaf],leaf])`

3 Priors for Bayesian C&RT

In this section we describe the actual priors we have used for the experiments described in Section 5. Fixing some notation, note that a given C&RT model t maps an example, described by a set of p predictor variables $x = x_1, x_2, \ldots x_p$, to a distribution over a response variable y. In the case of classification trees (also known as decision trees) y is discrete and finite, and in the case of regression trees y is continuous. Some C&RT implementations throw away information given by the distribution of y and simply return the majority class or mean value as appropriate, but this is just a design decision. Fundamentally it is a distribution over y that is given.

It follows that each tree t defines a conditional probability distribution $P(y|t,x)$. This conditional distribution is the likelihood function of the following version of Bayes theorem: $P(t|y,x) = P(t|x)P(y|t,x)/P(y|x)$ whose LHS gives the desired posterior distribution over trees. Note that the prior $P(t|x)$ is only prior to y, it is conditional on the predictor variables so that observed values of x in the data can be used to define sensible priors.

Importantly, this allows us to rule out *a priori* trees with leaves which contain few examples. There is nothing to stop a user allowing trees with sparsely populated leaves, but ruling out such trees is normal in this research area. It ensures that the space of possible tree structures is a reasonable (finite) size. In our experiments only trees whose leaves have at least 5 examples are permitted, following [Denison *et al.*, 1998]. The predictors x could also be used to define a prior biassed towards even splits of the data if so desired, but we have yet to investigate the desirability of such an option.

For most of our experiments we decided to use what we call a *GROWTREE prior*, which is essentially that used by [Chipman *et al.*, 1998]. Attempting to use an SLP to model an externally provided prior provides a test of the representational flexibility of SLPs: indeed we were unable to pass this test before we extended SLPs to have variable probability labels. Using the GROWTREE prior also allows better comparison with others' results.

The GROWTREE prior grows a C&RT tree by starting with a single leaf node and then repeatedly splits each leaf node η with a probability $\alpha(1+d_\eta)^{-\beta}$, where d_η is the depth of node η and α and β are prior parameters set by the user to control the size of trees. Unsplit nodes become leaves of the tree. If a node is split, then the splitting rule for that split is chosen uniformly. An abbreviated fragment of the SLP which expresses this prior is given in Fig 4.

```
Splt:[Splt]:splt_lf(D,..,A/B,..) :- ..
 D1 is D + 1,
 NwSplit is A * exp((1+D1),-B), ...
 [NwSplit]:splt_lf(D1,..,A/B,..),
 [NwSplit]:splt_lf( D1,..,A/B,..).
(1-Splt):[Splt]:splt_lf(D,..,AB,Lf) :-
        Lf = leaf(D,..).
```

Figure 4: GROWTREE prior fragment defined with an SLP

We have also experimented with what we call *EDITTREE* priors. Here an 'initial' C&RT model is supplied. This can be any tree, for example the tree produced by the standard greedy algorithm or one manually entered by the user. This tree is then probabilistically 'edited', by either growing, pruning or changing a splitting rule. The number of such edits is also probabilistic.

4 Using the Metropolis-Hastings algorithm

The Metropolis-Hastings (MH) algorithm is an MCMC algorithm which defines a Markov chain $(T^{(i)})$ with a transition kernel $K(t^i, t^{i+1})$ as follows. If $T^{(i)} = t^{(i)}$, then $T^{(i+1)}$ is produced by generating $T'_i \sim q(t'|t^{(i)})$ for some proposal distribution q. An acceptance probability $\alpha(t,t')$ is used to decide whether to accept the proposed t'. $t^{(i+1)} = t'$ with probability $\alpha(t,t')$ (the proposed t' is accepted) and $t^{(i+1)} = t^{(i)}$ with probability $1-\alpha(t,t')$. By setting the acceptance probability like this:

$$\alpha(t,t') = \min\left\{\frac{P(t'|x,y)}{P(t|x,y)}\frac{q(t|t')}{q(t'|t)}, 1\right\} \quad (1)$$

the chain $(T^{(i)})$ will converge to sampling from the desired posterior $P(T|x,y)$ under weak conditions (whatever the starting point $t^{(0)}$ of the chain). It is not difficult to see (via Bayes theorem) that if $q(t|t')P(t'|x) = R_q(t,t')q(t'|t)P(t|x)$, for some function R_q, then:

$$\alpha(t,t') = \min\left\{R_q(t,t')\frac{P(y|t',x)}{P(y|t,x)}, 1\right\} \quad (2)$$

In our approach we choose proposal distributions q which permit this simplification of the acceptance probability, and where the value $R_q(t,t')$ is easily computable for each pair of trees t, t' . The crucial point is this: such a choice of q ensures that $\alpha(t,t')$ is easily computable even though computing the prior probability for any given C&RT model (as opposed to merely sampling from the prior) may be prohibitively expensive. This contrasts with [Chipman *et al.*, 1998] where "specifications [which] allow for straightforward evaluation of $p(t)$ for any t" are required. For us the computation of prior probabilities can be expensive due to backtracking. Without backtracking SLPs are similar to stochastic-context free grammars (SCFGs)—particularly when probability labels are fixed, and not computed 'on the fly'—and SCFG dynamic programming algorithms could presumably be used to compute prior probabilities. Ruling out backtracking greatly restricts the constraints the user can declare so we always allow for backtracking.

Since our trees do not explicitly define a distribution over classes at the leaves of the tree, our likelihoods are marginal likelihoods—we integrate over all possible class distributions. This integration automatically prevents over-fitting: trees with 100% training set accuracy do not generally have high marginal likelihood. To actually do the integration, we use exactly the same approach as [Chipman *et al.*, 1998; Denison *et al.*, 1998; 2002]. For each leaf, the uniform Dirichlet distribution (all parameters set to 1) is the distribution over all possible distributions over classes. The Dirichlet assumption allows a closed form for the marginal likelihood $P(y|t,x)$.

Our proposals can best be understood by first recalling that sampling C&RT trees from an SLP prior is effected by sampling proofs from the prior as explained in Section 2, and that each such proof has an associated proof tree, such as shown in Fig 3. Each proposal we have considered works by pruning the proof tree corresponding to the current C&RT model at some point, and then re-growing the proof tree from the pruning point using the sampling mechanism associated with the prior. Our proposals only differ in how they choose where to 'snip' the proof tree. Since only the sub-tree below the prune point is resampled, our proposals are *not* equivalent to proposing with the prior $P(t|x)$ (except for one special case to be discussed shortly). Instead each prune point k corresponds to proposing a tree according to $P(t|x, t \in T_k)$ where T_k is the set of trees which may differ from the current tree only in the sub-tree under prune point k.

One (overly) simple proposal corresponds to the *independent Metropolis-Hastings* algorithm which prunes away the current model completely so that a new model is sampled from the prior independently of the current model. For this proposal (denoted q_0) we have the following MCMC convergence result. Let μ_i be the distribution over trees produced at the ith iteration of this independent MH algorithm. Let $\hat{t}$ be a C&RT tree with maximal marginal likelihood: then

$$|\mu_i - P(\cdot|x, y)| \leq [1 - P(\hat{t}|x)/P(\hat{t}|x, y)]^i \qquad (3)$$

where $|\cdot|$ denotes the total variation distance between two distributions. So there is exponentially fast convergence to the true posterior $P(\cdot|x, y)$ using this independent MH algorithm. The catch is that, $P(\hat{t}|x)/P(\hat{t}|x, y)$, the ratio between $\hat{t}$'s prior and posterior probability, is tiny for any reasonably sized dataset, so in experiments (not presented here) the independent M-H algorithm performs poorly. The convergence result can be derived from one by [Doob, 1953] which is given by [Rosenthal, 1995]. To apply Doob's result it is necessary to produce an upper bound on $\sum_{t'} \min_t K_{q_0}(t, t')$, where K_{q_0} is the transition kernel associated with q_0. In fact, we can show (proof omitted) that this sum is exactly $P(\hat{t}|x)/P(\hat{t}|x, y)$, thus establishing (3). This upper bound for our discrete finite space is exactly half the size of the bound for the general independent MH algorithm established by [Mengersen and Tweedie, 1996].

A somewhat better performing proposal (denoted q_{0-n}) cycles through proposals q_k for $k = 0, 1, 2, \ldots n$ for a user-supplied n, where q_k is the proposal which prunes the proof tree at the kth choice point. We have $R_{q_{0-n}}(t, t') \equiv 1$, so that the acceptance probability is simply a likelihood ratio. However, our best performing proposal (denoted q_{uc}) makes a *u*niform *c*hoice of a prune point from all those available in the current tree. For q_{uc}, $R_{q_{uc}}(t, t') = d_t/d_{t'}$, where d_t is the depth of tree t.

One recent innovation has greatly increased the efficiency with which the space is explored. In all previous work using SLPs for MCMC, the prune points to which our proposals jumped were not organised in a proof *tree*, but in a *sequence* (a branch of the relevant SLD-tree, in fact). Essentially, this was because our previous proposal mechanism was closely modelled on Prolog backtracking: if we backtracked to choice point k all C&RT building work done after that choice point was thrown away. Now, if we backtrack to point k then *only the proof tree below* k is (stochastically) rebuilt, *even if other parts of the proof tree were built chronologically later*. Translating this specifically to C&RT models, this means our proposals can now snip the current C&RT tree at any node, and re-grow the tree from there, leaving the rest of the C&RT tree unscathed. This makes it easier for the Markov chain to head in the direction of high likelihood trees: sub-trees with high-likelihood tend to be altered rarely whereas proposed replacements for low-likelihood sub-trees are accepted relatively often.

5 Experimental results

In our experiments we have used three datasets: the Wisconsin breast cancer data (BCW), the Kyphosis dataset (K) and the Pima dataset (P). BCW was originally donated to the UCI depository by [Wolberg and Mangasarian, 1990] and was used by [Chipman *et al.*, 1998]. Dataset K comes as part of the `rpart` R package for building and manipulating C&RTs. Dataset P is a UCI dataset which [Denison *et al.*, 2002] used for extensive Bayesian C&RT analysis. Following [Chipman *et al.*, 1998] we have simply deleted 16 datapoints from BCW which contain missing values. In all cases the machine learning task is binary classification using integer-valued predictors—9 predictors in the case of BCW, 3 for K and 8 for P. All splits are binary: made by splitting on some threshold. There are 683 datapoints for BCW, 81 for K and 768 for P.

We have performed a large number of experiments, and analysed the results in many ways. Only a small representative sample of this is reported here. Data and software for reproducing all of our experiments can be found at `http://www.cs.york.ac.uk/~nicos/sware/slps/mcmcms/`. We used Sicstus Prolog 3.11 running under Linux on two machines each with a 2.4GHz processor and at least 512Mb RAM. The computation of the log-likelihood ratio uses a C function called from Prolog. Running a chain for 50,000 iterations takes at most 30 minutes.

5.1 Comparison to the standard C&RT algorithm

The contrasts between Bayesian C&RT and the standard greedy algorithm for building a single C&RT model have been well explored in the existing Bayesian C&RT literature [Chipman *et al.*, 1998; Denison *et al.*, 1998; 2002]. The Bayesian method represents posterior uncertainty better and makes a more thorough exploration of the model space. This is at the cost of more computation. How well the Bayesian approach does in terms of predictive accuracy depends, of course, on the prior. When we do have useful prior knowledge it can help us if it can be incorporated into the prior distribution. Facilitating this is our central motivation.

A tree with maximal marginal likelihood is a maximum *a posteriori* (MAP) model when a uniform prior is assumed. Call such trees MAP_{unif} trees. Any MCMC run produces an approximation to a MAP_{unif} tree: it is the maximum marginal likelihood tree in the MCMC sample. Since the standard greedy algorithm is closely connected to a search for a MAP_{unif} tree [Buntine, 1990] it is interesting to compare the marginal likelihood of trees produced using the greedy algorithm to that of the MAP_{unif} tree approximations found in the the MCMC sample.

The trees found by `rpart` for datasets BCW and K using default settings have (rounded) marginal log-likelihoods -97

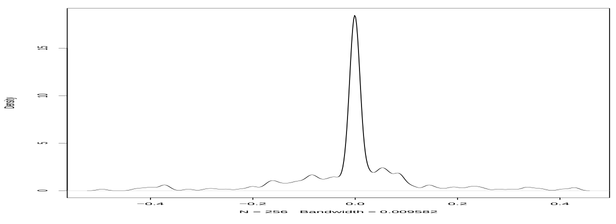

Figure 5: Distribution of differences in estimated class posterior probabilities from two MCMC runs only differing in random seed.

and -39, and sizes 7 and 5, respectively. Unsurprisingly, there are trees produced by MCMC runs with higher marginal log-likelihoods (i.e. with values closer to that of a MAP_{unif} tree). For BCW (resp. K) we can find a tree with 5 (resp. 3) leaves and log-likelihood of -86 (resp. -36).

5.2 Reproducibility of results

Since the aim of using MCMC is to approximate the posterior distribution, it is important that inferences drawn from any reasonably long realisation of the Markov chain are robust to changing the random seeds which determine the evolution of the chain. Using our original sequence-based proposals, there was strong evidence that this was *not* the case: for example, plots of log-likelihood against iteration number (*log-likelihood trajectories*) could be quite different for experiments which were identical except for the random seeds used.

To test our new proof-tree based proposals we performed the following experiment (amongst many others not reported here). Two MCMC runs of 50,000 iterations were performed using the GROWTREE prior ($\alpha = 0.95, \beta = 0.8$) using only 2/3 of the Pima dataset, differing only in the random seeds used. For each of the remaining 256 Pima dataset examples each of the two MCMC samples were used to approximate the posterior probability that the example had class label 0. This was done by simply getting each tree in the MCMC sample to make a class prediction for the example based on the majority class at the appropriate leaf and setting the class posterior probability equal to the relative frequency of class 0 predictions. Clearly, if both MCMC samples were perfect representations of the true posterior, then the difference in estimates of the class 0 posterior would be zero for each example. Naturally, this is not achieved but the distribution of the differences in probability estimates (shown in Fig 5) is highly concentrated about zero which provides evidence that our approach often produces values close to the true posterior, for different runs.

5.3 Using local jumps

Local jumping in tree space helps prevent getting stuck. The log-likelihood trajectory using q_{0-7} shown in Fig 6, shows that the big jumps which q_{0-7} proposes are rarely accepted, so the chain remains stuck at the same model for long periods. This problem is even more pronounced if q_0, the independent MH algorithm,is used. Compare this with other figures where the q_{uc} is being used.

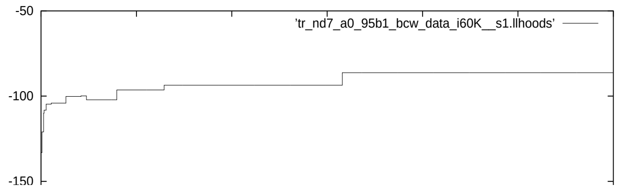

Figure 6: Log-likelihood trajectory. Prior=GROWTREE ($\beta = 1$), Data=BCW, MCMC=q_{0-7}.

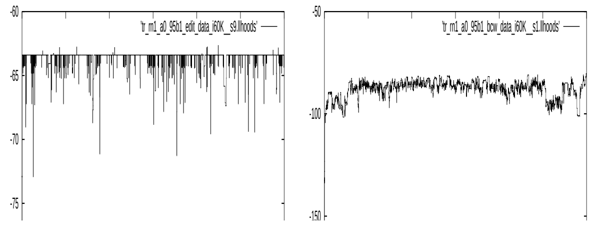

Figure 7: Log-likelihood trajectories. In both cases: Data=BCW, MCMC=q_{uc}, sequential. For LHS: Prior=EDITTREE. For RHS: Prior=GROWTREE ($\beta = 1$)

5.4 Influence of priors

Fig 7 compares log-likelihood trajectories using the EDITTREE prior and the GROWTREE prior respectively with all other parameters being equal. The distinct horizontal line in the EDITTREE trajectory is clear evidence that the EDITTREE prior is pulling the Markov chain back to the initial tree. Note that Figs 6-7 all contain initial very low log-likelihoods (corresponding to the chain's initial C&RT model) which have been truncated for space reasons.

[Denison *et al.*, 2002] examined how successful their chains were at finding C&RT trees with high marginal likelihood using the Pima (P) dataset. Their maximum marginal log-likelihood was -343; our highest was -347. To test a hypothesis that many high likelihood C&RT models did not have high posterior probability for our priors (and hence were unlikely to be visited), we started chains from the log-likelihood=-343 C&RT model of Denison *et al.*. Fig 8 show the results where the RHS trajectory corresponds to a prior with a stronger bias towards smaller trees. In both cases the prior pulls away from high-likelihood C&RT models, all the more rapidly for the stronger prior.

Since the goal of including prior knowledge is to improve decision making under uncertainty (e.g. classifying test examples), we performed experiments where a known 'true' tree was used to generate synthetic train and test data. We then included boxes (see Section 2) consistent with the true tree as constraints on a GROWTREE prior, produced MCMC samples using the synthetic training data, and measured predictive accuracy on the synthetic test data. Each of the 500 test examples was classified into its most probable class as estimated by the MCMC sample. This was all repeated for 3 random seeds. With no box constraints test-set accuracies were 74.8%, 76.2% and 74.4%. With the box constraints given in Fig 1, the test-set accuracy was exactly 78.6% for each random seed. These results indicate that accurate prior

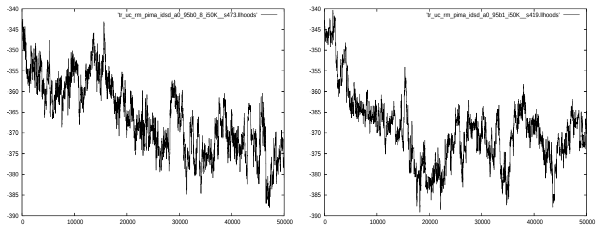

Figure 8: Log-likelihood trajectories. In both cases: Data=P, MCMC=q_{uc}, proof tree. Prior=GROWTREE. Initial tree = Denison's maximal likelihood. For LHS: $\beta = 0.8$. For RHS: $\beta = 1$

knowledge (i) helps convergence and (ii) increases predictive accuracy.

6 Comparison with related work

This paper lies at the intersection of two lines of work: that on Bayesian C&RT [Chipman *et al.*, 1998; Denison *et al.*, 1998; 2002] and that on using SLPs for general Bayesian model inference [Angelopoulos and Cussens, 2001]. The SLP work had claimed to provide a general Bayesian machine learning method but had only results for model spaces composed of Bayesian nets. This paper backs up this initial claim by applying the basic framework of [Angelopoulos and Cussens, 2001] to C&RT model space. The SLP method has also been improved in a number of ways: (i) probabilities no longer need be hard-coded constants, (ii) the use of *backtrackable sampling* adds an element of search to the prior and (iii) the proposal mechanism now works on a (proof) tree structure which makes it easier to move through the model space.

The most obvious difference from other Bayesian C&RT work is that here all (non-parameter) priors are defined by an SLP—however as we have seen SLPs can encode (at least some) priors originally expressed otherwise. Secondly, in our approach there is only one way of proposing new C&RT models: by pruning and re-growing the proof tree. In the other Bayesian C&RT work a variety of moves are used—five are listed in [Denison *et al.*, 2002, p. 161]. The compensation for our severely restricted proposals is that no prior terms complicate the acceptance probability. Constraining the proposal mechanism to produce such an acceptance probability is also a choice taken in [Denison *et al.*, 2002]—although for this to work the proposal mechanism has to depend upon global parameters of the current tree. The biggest contrast with previous Bayesian C&RT work is that we permit big jumps by pruning the proof tree near the root. In [Denison *et al.*, 2002] it is noted that pruning off any given branch of a tree is straightforward in their approach but that how to generate a similar branch (to maintain reversibility) "is not obvious". In the approach presented here it is obvious, we can just re-grow the proof tree *because the proposal is based on the prior*. In [Denison *et al.*, 2002] the danger that many big jumps will propose trees with sparsely populated leaves is also noted—we avoid this problem by simply compelling the proposal mechanism to search for trees (via backtracking) with adequately populated leaves.

7 Future work

The convergence result (3) for the independent MH algorithm is something we hope to generalise to other proposals. This will guide the choice of proposal mechanism in contrast to our current empirical approach. In ongoing work [Angelopoulos and Cussens, 2005] experimental results have been produced which show that *tempering* improves the rate of convergence of q_{uc} considerably. Thirdly, like all other work (that we know of) on Bayesian C&RT, we do not have full-blown convergence diagnostics implemented: a deficiency we hope to remedy.

Our current implementation does not fully exploit Prolog's built-in backtracking: if this were possible this would permit a significant speed-up. A more radical approach is to adapt an existing Prolog system to have our probabilistic mechanisms built-in.

Acknowledgements

Thanks to 3 anonymous reviewers for useful criticisms. This work was supported by the UK EPSRC MathFIT project *Stochastic Logic Programs for MCMC* (GR/S30993/01).

References

[Angelopoulos and Cussens, 2001] N. Angelopoulos and J. Cussens. Markov chain Monte Carlo using tree-based priors on model structure. In *Proc. UAI-01*, 2001.

[Angelopoulos and Cussens, 2005] N. Angelopoulos and J. Cussens. Tempering for Bayesian C&RT. Sub'd, 2005.

[Buntine, 1990] Wray Buntine. *A Theory of Learning Classification Rules*. PhD thesis, Uni. of Tech., Sydney, 1990.

[Chipman *et al.*, 1998] H. A. Chipman, E. I. George, and R. E. McCulloch. Bayesian CART model search. *JASA*, 39(443):935–960, 1998.

[Cussens, 2000] J. Cussens. Stochastic logic programs: Sampling, inference and applications. In *Proc. UAI-00*, 2000.

[Denison *et al.*, 1998] D. G. T. Denison, B. K. Mallick, and A. F. M. Smith. A Bayesian CART algorithm. *Biometrika*, 85(2):363–377, 1998.

[Denison *et al.*, 2002] D. G. T. Denison, C. C. Holmes, B. K Mallick, and A. F. M. Smith. *Bayesian Methods for Nonlinear Classification and Regression*. Wiley, 2002.

[Doob, 1953] J. I. Doob. *Stochastic Processes*. Wiley, New York, 1953.

[Mengersen and Tweedie, 1996] K. L. Mengersen and R. L. Tweedie. Rates of convergence of the Hastings & Metropolis algorithms. *Ann. Stats.*, 24:101–121, 1996.

[Muggleton, 1996] S. Muggleton. Stochastic logic programs. In *Advances in ILP*. 1996.

[Rosenthal, 1995] J. S. Rosenthal. Convergence rates for Markov chains. *SIAM Review*, 37(3):387–405, 1995.

[Wolberg and Mangasarian, 1990] W. H. Wolberg and O. L. Mangasarian. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *PNAS*, 87:9193–9196, 1990.