Tempering for Bayesian C&RT

Nicos Angelopoulos NICOS@CS.YORK.AC.UK
James Cussens JC@CS.YORK.AC.UK
Department of Computer Science, University of York, Heslington, York, YO10 5AY, UK

Abstract

This paper concerns the experimental assessment of *tempering* as a technique for improving Bayesian inference for C&RT models. Full Bayesian inference requires the computation of a posterior over all possible trees. Since exact computation is not possible Markov chain Monte Carlo (MCMC) methods are used to produce an approximation. C&RT posteriors have many local modes: tempering aims to prevent the Markov chain getting stuck in these modes. Our results show that a clear improvement is achieved using tempering.

1. Introduction

The purpose of this paper is to experimentally assess the degree to which *tempering* can improve the performance of a Markov chain Monte Carlo (MCMC) approach to doing Bayesian inference for Classification and Regression Tree (C&RT) models. In the Bayesian approach to C&RT learning the ultimate goal is to compute an entire posterior distribution over all possible C&RT models, rather than generate a single 'best guess' model as does the standard greedy C&RT algorithm (Breiman et al., 1984). In all realistic cases the space of possible C&RT models is too big to allow exact computation of the posterior, so (in all research of which we are aware) an approximate sample from the posterior is generated using MCMC sampling. (MCMC sampling is explained in Section 4.)

2. On getting stuck and unstuck

The key difficulty with this MCMC approach is that the posterior distribution will have many local modes, making it difficult for any MCMC algorithm to produce a representative sample from the posterior. In short, there is a tendency for the Markov chain to 'get stuck' at a local mode. Chipman et al. (1998) sum up the issues nicely:

> ...the algorithm gravitates quickly towards [regions of large posterior probability] and then stabilizes, moving locally in that region for a long time. Once a tree has reasonable fit, the chain is unlikely to move away from a sharp local mode by small steps. ...Although different move types might be implemented, we believe that any MH algorithm for CART models will have difficulty moving between local modes.

More recently, Chipman et al. (2003) have explored generalised tree models where the same basic MCMC approach is used and it remains the case that "the chain may get trapped in local maxima". Denison et al. (2002, p.165) note similarly that "...in practice it is because the waiting times in local modes are so large that full sampling is infeasible." In our own recent work (Angelopoulos & Cussens, 2005), we have made some progress on this problem, but there remains plenty of room for further improvement.

Bayesian C&RT is not the only line of research applying MCMC to Bayesian inference of tree-based models: there is also bioinformatics research on *Bayesian estimation of phylogeny*. Phylogenetic trees are like family trees, but they relate species rather than individuals. The species are situated at the leaves of the tree, and common ancestors are denoted by internal nodes of the tree. Phylogenetic trees are inferred from aligned sequence data (usually genes) from the various species of interest. A number of approaches exist, including maximum likelihood and compression-based learning. Bayesian inference using MCMC is becoming increasingly popular, e.g. using the freely available MrBayes package (`http://morphbank.ebc.uu.se/mrbayes/`). A description of MrBayes is provided by Altekar et al. (2004) and references contained therein.

Appearing in *Proceedings of the 22nd International Conference on Machine Learning*, Bonn, Germany, 2005.

Bayesian phylogenetic inference has the same essential problem as Bayesian C&RT:

> The posterior probability of trees can contain multiple peaks. ... MCMC can be prone to entrapment in local optima; a Markov chain currently exploring a peak of high probability may experience difficulty crossing valleys to explore other peaks. (Altekar et al., 2004).

The approach successfully taken in the MrBayes package to address this 'stickiness' is to use *Metropolis-coupled MCMC* also known as *tempering*. Tempering, described in Section 5, uses extra 'hot' chains in addition to the normal 'cold' MCMC chain. The hot chains move through the space more easily, and, together with probabilistic 'swaps' between hot and cold chains, allow more rapid convergence to the true posterior.

Our goal is to see whether this successful 'unsticking' method will also succeed for C&RT models. To address this issue we first, in Section 3, consider the basic ingredients of Bayesian C&RT: priors and likelihoods. In Section 4 we describe how to use the Metropolis-Hastings algorithm to explore the posterior distribution determined by a particular prior and likelihood. Section 5 describes how tempering works and Section 6 describes our datasets. The key section of the paper is Section 7 where we compare results obtained with and without tempering. Section 8 contains conclusions and plans for future work.

3. Bayesian C&RT

In this paper a C&RT model consists of a tree structure T and its parameters Θ. The model (Θ, T) defines a *probability model*: where the parameters Θ define a distribution for each leaf of T. This follows the standard approach in Bayesian C&RT, and indeed all of this section, with the exception of that which concerns stochastic logic programs, sets up the standard Bayesian C&RT framework as found in the literature. We use the same notation as Chipman et al. (1998) throughout.

In the case of classification trees, Θ defines a distribution over the classes for each leaf of the tree T. The tree is defined in the normal way via splitting rules: note that threshold values in splitting rules form part of the tree T, not Θ. T maps an unlabelled example x to a unique leaf, so that (Θ, T) defines a conditional probability model $P(y|\Theta, T, x)$ where y is the class for x. In this paper we restrict attention to classification trees, but trees can be used to define many other probability models, see (Chipman et al., 2003). If T has b leaves, then $\Theta = (\theta_1, \ldots \theta_b)$. If there are K possible classes, then for each leaf i $(1 \leq i \leq b)$, $\theta_i = (p_{i1}, \ldots p_{iK})$ is just the vector of class probabilities for that leaf.

As for any application of the Bayesian approach, the goal in Bayesian C&RT is to compute a posterior distribution over models given a prior and some data. Here, each model is a parameterised tree (Θ, T), and the data is of the form (X, Y). X is a vector of n unlabelled examples, and Y is the corresponding vector of n class labels, so that Y_j is the class label for example X_j.

The posterior distribution over models is thus $P(\Theta, T|X, Y)$. We take advantage of the factorisation $P(\Theta, T|X, Y) = P(\Theta|T, X, Y)P(T|X, Y)$. We will chose conjugate priors so that $P(\Theta|T, X, Y)$ is analytically computable for any T, X, Y. Interest then focuses on the *model structure posterior* $P(T|X, Y)$. From Bayes' theorem we have that $P(T|X, Y) \propto P(T|X)P(Y|X, T)$. The *model structure prior* $P(T|X)$ and *marginal likelihood* $P(Y|X, T)$ are now defined in Sections 3.1 and 3.2, respectively.

3.1. Priors for Bayesian C&RT

The model structure prior $P(T|X)$ is defined via a stochastic sampling procedure. In this we follow Chipman et al. (1998). However, our C&RT priors are just a specific instance of a general logic programming based method for defining priors over model structure. We define priors using *stochastic logic programs* (SLPs). The set of permissible models is determined by defining a first-order predicate `cart/1` in a logic program, and then adding probabilities to the logic program so that there is a probability distribution over instantiations of that predicate. One principal rationale for such an approach is that the logic-based formalism is sufficiently flexible to allow the same system to be used for different sorts of Bayesian model inference. Our system has been applied to learning Bayes nets, as well as C&RT models: in each case we just use an appropriate SLP to define a prior, and plug in the necessary likelihood function.

The focus on this paper is to assess the utility of tempering, rather than address the knowledge engineering issues of using SLPs to encode priors encoding domain knowledge. The latter issue is addressed elsewhere (Angelopoulos & Cussens, 2005). Consequently, we have chosen to use the same family of priors (encoded as SLPs) as Chipman et al. (1998). Such a 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, the splitting rule for that split is chosen uniformly. Since $P(T|X)$ is conditional on X we can (and do) give zero probability to any tree which has a leaf which contains fewer examples than some user-defined threshold. If the node-splitting procedure produces such a leaf, our SLP prior invokes Prolog-based backtracking to (probabilistically) search for a tree which does meet the constraint.

3.2. Likelihoods for Bayesian C&RT

The marginal likelihood $P(Y|X,T)$, requires Θ to be integrated out of the full likelihood $P(Y|X,\Theta,T)$: that is what makes it 'marginal'. This integration is with respect to the structure-conditional parameter prior $P(\Theta|T,X)$, which we define in the standard way. Firstly, we choose to define the prior to be independent of X so that $P(\Theta|T,X) = P(\Theta|T)$. Secondly, we assume independence between the θ_i in Θ, so that $P(\Theta|T) = \prod_{i=1}^{b} P(\theta_i|T)$ where b is the number of leaves in T. Recall that θ_i is $(p_{i1},\dots p_{iK})$: a class probability distribution. For all i, we set $P(\theta_i|T)$ to the same distribution: a Dirichlet distribution with parameters $(\alpha_1,\dots,\alpha_K)$, for some user-defined choice of α_k $(1 \le k \le K)$.

With such a parameter prior there is a closed form for $P(Y|X,T)$. Let n_i denote the number of examples at leaf i in tree T. Let n_{ik} be the number of examples of class k reaching leaf i, (so that $n_i = \sum_k n_{ik}$). Then the marginal likelihood is:

$$p(Y|X,T) = \left(\frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)}\right) \prod_{i=1}^{b} \frac{\prod_k \Gamma(n_{ik}+\alpha_k)}{\Gamma(n_i + \sum_k \alpha_k)} \quad (1)$$

4. Exploring posteriors with the Metropolis-Hastings algorithm

The space of possible C&RT trees is too large to allow the exact computation of posterior probabilities for all trees. Instead, we will produce an approximate sample from the posterior using MCMC. As in all other Bayesian C&RT work of which we are aware, we use the Metropolis-Hastings algorithm to produce this sample. The Metropolis-Hastings algorithm is an MCMC algorithm which defines a Markov chain $(T^{(i)})$ with a transition kernel $K(T^i, T^{i+1})$ as follows. $T^{(i+1)}$ is produced by generating $T' \sim q(\cdot|T^{(i)})$ for some proposal distribution q. An acceptance probability $\alpha(T^{(i)},T')$ is used to decide whether to accept the proposed T'. $T^{(i+1)} = T'$ with probability $\alpha(T^{(i)},T')$ (the proposed T' is accepted) and $T^{(i+1)} = T^{(i)}$ with probability $1-\alpha(T^{(i)},T')$. By setting the acceptance probability like this:

$$\alpha(T^{(i)},T') = \min\left\{\frac{P(T'|X,Y)}{P(T^{(i)}|X,Y)}\frac{q(T^{(i)}|T')}{q(T'|T^{(i)})},1\right\}$$

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). Defining $R_q(T^{(i)},T')$ to be $\frac{q(T^{(i)}|T')P(T'|X)}{q(T'|T^{(i)})P(T^{(i)}|X)}$ it is not difficult to see (via Bayes theorem) that

$$\alpha(T^{(i)},T') = \min\left\{R_q(T^{(i)},T')\frac{P(Y|T',X)}{P(Y|T^{(i)},X)},1\right\}$$

In our approach we choose proposal distributions q where the value $R_q(T^{(i)},T')$ is easily computable for any pair of trees $T^{(i)},T'$.

A more detailed account of our various proposal mechanisms can be found elsewhere (Angelopoulos & Cussens, 2005), here we just describe the essential features of one particular mechanism, called *uniform choice backtracking* (q_{uc}), which has proved the most successful in practice.

Recall that a tree is sampled from our prior by probabilistically splitting leaf nodes and probabilistically choosing splitting rules. Consider the sequence of probabilistic choices made to generate the tree T from the prior: this includes split/don't split choices as well as choices for splitting rules and splitting thresholds. Each such probabilistic choice is situated at a *choice point*. Let d_T be the number of choice points that were travelled through to produce tree T.

To propose a new tree T' from an existing one $T^{(i)}$ using q_{uc}, the fundamental idea is to probabilistically *backtrack* to one of these choice points according to a uniform distribution and then probabilistically regrow the tree from the chosen choice point according to the prior. However, we have recently (Angelopoulos & Cussens, 2005) refined this basic approach so that the choice points are arranged in a *tree* (a Prolog proof tree, not a C&RT tree!) rather than a *sequence*. This permits the proposal to prune and regrow at arbitrary internal nodes of the C&RT tree without altering other branches of the tree.

Because the prior is incorporated into the proposal mechanism in this way, it turns out that $R_{q_{uc}}(T^{(i)},T')$ is simply $d_{T^{(i)}}/d_{T'}$ and so the acceptance probability is $\alpha_{uc}(T^{(i)},T') = \min\left\{\frac{d_{T^{(i)}}}{d_{T'}}\frac{P(Y|T',X)}{P(Y|T^{(i)},X)},1\right\}$. $\alpha_{uc}(T^{(i)},T')$ is thus easily computable, particularly

since the data-independent part of the marginal likelihood (1) cancels out.

Denison et al. (2002, p.164) argue that to address the problem of 'stickiness' "The most obvious idea would be to add/delete whole branches of the tree, not just to use simple split/combine moves ..." but do not provide a way to achieve this. The Prolog proof tree based procedure just outlined does achieve this. However, this is at the expense of tying the proposal closely to the prior and so restricting how proposals can be made. For example, we cannot implement the CHANGE-SPLIT and SWAP proposals detailed by Denison et al. (2002, p.161).

5. Tempering

As will be seen in Section 7 in some cases straightforward uniform choice backtracking provides good posterior probability estimates, but in other cases there is room for improvement. In this section we describe the technique of *tempering* (aka *Metropolis-coupled MCMC*) (Geyer & Thompson, 1995) which aims to achieve improved mixing for Markov chains at the expense of greater computational effort.

The basic idea is simple and is described in numerous papers; here we follow the account given by (Altekar et al., 2004). As well as running a 'cold' chain which has the desired posterior $P(T|X,Y)$ as its stationary distribution, we also run 'hot' chains with stationary distributions $P(T|X,Y)^\beta$ for different values of a *heat value* β ($0 < \beta < 1$). Note that the smaller β is, the flatter $P(T|X,Y)^\beta$ is, and the smaller is the risk of a chain getting stuck in local modes.

To run a hot chain with heat value β using our uniform choice backtracking proposal, it suffices to alter the acceptance probability to:

$$\alpha_{uc}^{\beta}(T^{(i)}, T') = \min\left\{\frac{d_{T^{(i)}}}{d_{T'}}\left(\frac{P(Y|T',X)}{P(Y|T^{(i)},X)}\right)^{\beta}, 1\right\}$$

The closer β is to 0, the greater the acceptance probability and the more easily the chain moves through the space. Smaller values of β correspond to hotter chains.

Tempering also involves swaps between chains. After a given number of iterations of each chain, two chains are chosen at random and their states are swapped with a certain swap-acceptance probability α_{swap}. In our approach a swap is proposed after every iteration. Suppose that two chains with heat values β_1 and β_2 are proposed for swapping and these two chains are currently visiting trees T_1 and T_2, respectively. Then in our case:

$$\alpha_{\text{swap}} = \min\left\{\left(\frac{P(Y|T_2,X)}{P(Y|T_1,X)}\right)^{(\beta_1-\beta_2)}, 1\right\}$$

It remains to choose how many chains to run and what their heat values should be. Given that the MrBayes package had successfully used tempering we simply opted for the (default) approach used there. Four chains were run with heat values $\beta_i = 1/(1+\Delta T(i-1))$ for $i = 1, 2, 3, 4$ and where the temperature ΔT was set to 0.2. Note that chain $i = 1$ is the cold chain. Only trees visited by the cold chain are collected to form the MCMC sample: the hot chains are there only to allow the cold chain to make 'bigger jumps' via state swapping.

6. Datasets

In our experiments we have used 5 datasets: Wisconsin breast cancer (BCW), Kyphosis (K), Pima (PIMA), Letter Recognition (LR) and Waveform (WF). BCW was originally donated to the UCI depository by Wolberg and Mangasarian (1990) and was used by Chipman et al. (1998). BCW contains 16 missing data values. Following (Chipman et al., 1998) we have simply deleted datapoints which contain missing values. Dataset K comes as part of the `rpart` R package for building and manipulating classification and regression trees. PIMA is a UCI dataset which Denison et al. (2002) used for extensive Bayesian C&RT analysis. LR and WF are both datasets from the UCI repository.

With the exception of WF, each dataset was divided arbitrarily into a training set (80%) and a hold-out set (20%). Due to an unfortunate error, for WF these proportions were swapped. The hold-out set is not a 'test set' in the conventional sense, since our goal—as is normal in MCMC research—is to maximise the accuracy of our approximation to the posterior, not to maximise predictive accuracy. The hold-out set is used to evaluate this approximation, as described in Section 7.2.

The datasets are summarised in Table 1. For each dataset, one of the values of the class variable was arbitrarily designated as 'positive'. Note that, by chance, there are significantly fewer positives in the hold-out set for K than in the training set. Note also that K is something of a 'toy' dataset: small and with very few attributes.

7. Experimental results

The same prior was used in all experiments: $\alpha = 0.95, \beta = 1$ with all Dirichlet prior parameters set to 1,

Name	Size	$\|x\|$	$\|Y\|$	Pos%(Tr)	Pos%(HO)
K	81	3	2	81.5%	68.8%
BCW	683	9	2	66.2%	60.3%
PIMA	768	8	2	65.4%	64.1%
LR	20000	16	26	3.85%	4.3%
WF	5000	40	3	35.6%	33.4%

Table 1. Description of datasets. Size denotes the number of examples in both training and hold-out set. $|x|$ denotes the number of attributes. $|Y|$ is the number of classes. The last 2 columns give the percentage of 'positives' in the train and hold-out set, respectively.

and where leaves have to contain at least 5 examples. For datasets K, BCW and PIMA, each Markov chain we ran was determined by 3 parameters: dataset (=K, BCW or PIMA); number of iterations (=50000, 125000 or 250000) and tempering (=TRUE or FALSE). Only the 'training set' portion of each dataset was used. Each of these $3 \times 3 \times 2 = 18$ Markov chains was realised 3 times using different random seeds, leading to $18 \times 3 = 54$ runs. For datasets LR and WF only a subset of such experiments have been performed. For datasets K, BCW, PIMA, LR and WF, each 1000 iterations took, respectively: 1.8s, 8.1s, 39s, 530s and 291s without tempering. With tempering turned on each 1000 iterations took: 4.9s, 17s, 129s, 2368s and 1151s respectively. Runs could thus run for many hours. For each run, we collected the trees visited (if tempering=TRUE only cold chain trees) and plotted the marginal likelihood, number of leaves and tree depth against the iteration number. We use no 'burn-in': all trees are kept. Such *trajectories* often allow 'sticking' to be visually obvious. Unfortunately there is no space here to present and analyse these trajectories.

Our goal is to produce a good approximation to the true posterior distribution over C&RT models for any given prior and set of training data. Since the true posterior is unavailable—which is why we resort to MCMC—a direct evaluation of the accuracy of our approximations is not possible. Instead we compare posterior probability estimates produced from distinct realisations of a Markov chain. If different runs produce similar (resp. dissimilar) results it is evidence that they are (resp. aren't) both reasonable approximations to the true posterior.

7.1. Robustness of posterior tree probability estimates

The simplest comparison is to compare estimates of posterior probability for *individual trees*. Considering only our longest runs of 250,000 iterations with tempering turned on, Tables 2–4 show the probability estimates produced for the most frequently visited trees for K, BCW and PIMA. Each row corresponds to a tree (tree not shown), each column to a different random seed for the MCMC run. We see that for dataset K, the estimated posterior probability for each tree is very similar for each of the 3 runs. Indeed Bayesian inference for K is so easy that 3 runs of only 50,000 iterations *without tempering* produce stable probability estimates as Table 5 shows.

In contrast, for PIMA and BCW frequently visited trees in one run are *virtually never* visited in other runs, so posterior probability estimates for individual trees are quite different. (We expect similarly poor results for LR and WF, but have yet to do this particular comparison). All other runs involving PIMA and BCW also display this behaviour. This is essentially because for PIMA and BCW the set of possible trees is much larger than for K.

Tree	$\hat{p}_{\text{seed1}}(T_i)$	$\hat{p}_{\text{seed2}}(T_i)$	$\hat{p}_{\text{seed3}}(T_i)$
T_1	0.086416	0.088524	0.087404
T_2	0.062296	0.062904	0.063624
T_3	0.060360	0.057648	0.058220
T_4	0.025808	0.025200	0.025764
T_5	0.024512	0.024404	0.025700

Table 2. Probability estimates for 5 most frequent trees for 3 different random seeds. Data=K, iterations=250,000, tempering=TRUE

Tree	$\hat{p}_{\text{seed1}}(T_i)$	$\hat{p}_{\text{seed2}}(T_i)$	$\hat{p}_{\text{seed3}}(T_i)$
T_1	0	0.008112	0
T_2	0.004612	0	0
T_3	0	0	0.004048
T_4	0	0	0.003600
T_5	0	0.003432	0

Table 3. Probability estimates for 5 most frequent trees for 3 different random seeds. Data=PIMA, iterations=250,000, tempering=TRUE.

Tree	$\hat{p}_{\text{seed1}}(T_i)$	$\hat{p}_{\text{seed2}}(T_i)$	$\hat{p}_{\text{seed3}}(T_i)$
T_1	0	0	0.088856
T_2	0	0	0.070996
T_3	0	0.04732	0
T_4	0	0	0.04316
T_5	0	0.03998	0

Table 4. Probability estimates for 5 most frequent trees for 3 different random seeds. Data=BCW, iterations=250,000, tempering=TRUE.

Tree	$\hat{p}_{\text{seed1}}(T_i)$	$\hat{p}_{\text{seed2}}(T_i)$	$\hat{p}_{\text{seed3}}(T_i)$
T_1	0.08326	0.07898	0.08338
T_2	0.05900	0.06154	0.06170
T_3	0.05574	0.05664	0.05610
T_4	0.02466	0.02724	0.02790
T_5	0.02564	0.02674	0.02504

Table 5. Probability estimates for 5 most frequent trees for 3 different random seeds. Data=K, iterations=50,000, tempering=FALSE.

7.2. Robustness of posterior class probability estimates

The results of the previous section suggest that the presented method only works for simple cases like dataset K. However, if we want to estimate the *posterior class distribution* for any given unlabelled example x', good results are available for all datasets. The posterior probability that x' has class y' is:

$$p(y'|x', X, Y) = \sum_T P(T|X, Y) \int p(y'|x', \Theta, T) P(\Theta|T, X, Y) d\Theta$$

Since we have chosen to use Dirichlet prior distribution for the parameters, the integral has an analytic solution: it is just the mean of the posterior Dirichlet distribution at the leaf where T sends x'. The posterior structure distribution $P(T|X, Y)$ is estimated, of course, by our MCMC samples: we pass through each tree in the sample and find the relevant mean value, add all these up, and divide by the number of trees.

For each example in a test set, we find the estimated probability that it is 'positive' for 2 different MCMC runs differing only in the random seed used. We then plot these values against each other. Figs 1–9 show a representative selection of the results where we vary the dataset, the number of iterations and whether tempering is used or not. Clearly, we hope that the 2 estimates from the 2 different runs are close, so the more points near the diagonal the better.

As Fig 1 shows, all points for dataset K are right on the diagonal as expected, even for only 50,000 iterations with no tempering. Note also that all class=positive probabilities are estimated to be greater than 0.5, reflecting the preponderance of positives in the K training set. For PIMA and BCW, the (50k, tempering=T) results are noticeably better than the the (250k, tempering=F) ones. So even taking the extra computational effort into account, tempering helps here. For LR and WF a visual inspection of Figs 6–9 does not provide evidence in favour of tempering, however such evidence is provided by the quantitative approach of

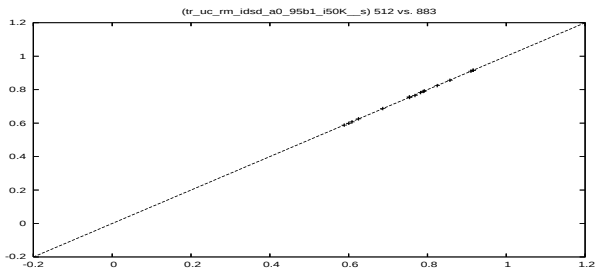

Figure 1. Comparison of class=positive probability estimates. Dataset=K, iterations=50,000, tempering=FALSE

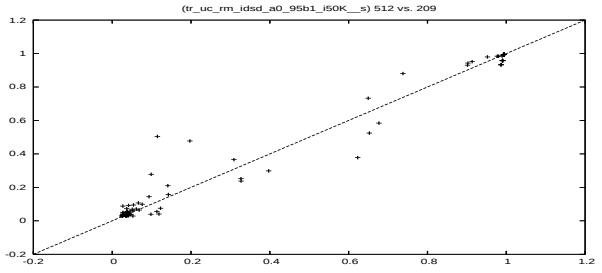

Figure 2. Comparison of class=positive probability estimates. Dataset=BCW, iterations=50,000, tempering=T

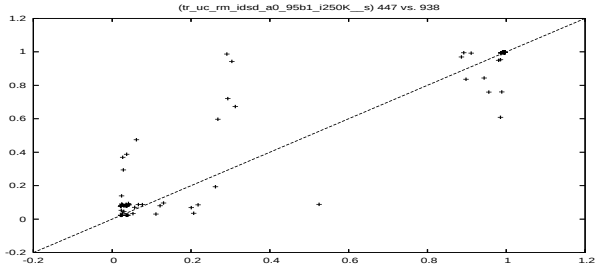

Figure 3. Comparison of class=positive probability estimates. Dataset=BCW, iterations=250,000, tempering=F

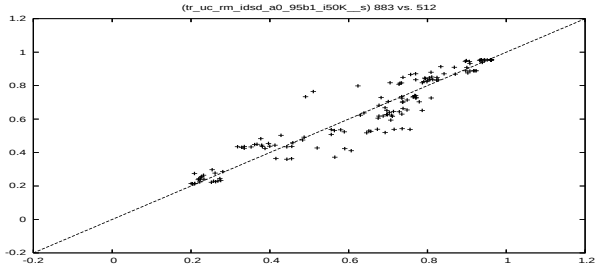

Figure 4. Comparison of class=positive probability estimates. Dataset=PIMA, iterations=50,000, tempering=T

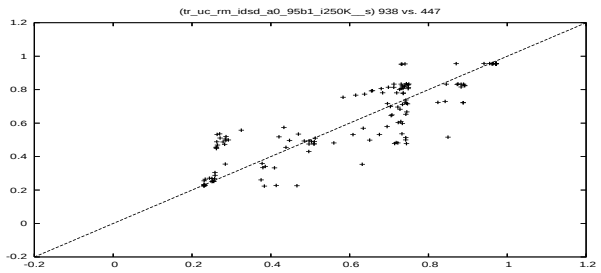

Figure 5. Comparison of class=positive probability estimates. Dataset=PIMA, iterations=250,000, tempering=F

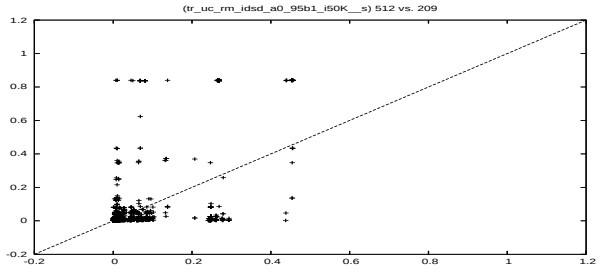

Figure 6. Comparison of class=positive probability estimates. Dataset=LR, iterations=50,000, tempering=F

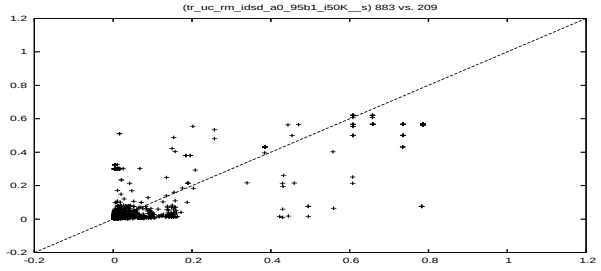

Figure 7. Comparison of class=positive probability estimates. Dataset=LR, iterations=50,000, tempering=T

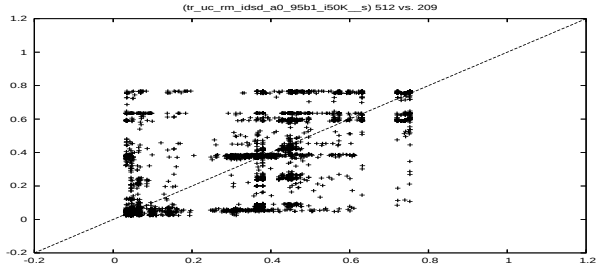

Figure 8. Comparison of class=positive probability estimates. Dataset=WF, iterations=50,000, tempering=F

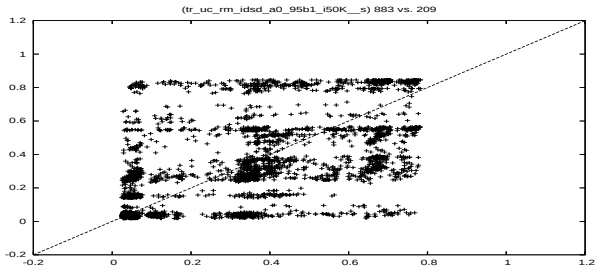

Figure 9. Comparison of class=positive probability estimates. Dataset=WF, iterations=50,000, tempering=T

the next section.

7.3. Robustness of predictive accuracy

It is instructive to summarise the effects of tempering quantitatively. To this end we compared predictive accuracies on the hold-out set using MCMC samples produced using 3 different random seeds. To predict the class of an example, one can either just choose the most probable class or one can predict probabilistically. In the latter case, x' is predicted to be positive with probability $P(y = \text{positive}|x', X, Y)$. Denote these two predictive accuracies as acc_{max} and acc_{prob}, respectively.

Predictive accuracies for dataset K were virtually constant for all random seeds, all chain lengths and whether or not tempering was used. For K, we have $\text{acc}_{\text{max}} = 11/16 = 68.8\%$ always, since all examples are more likely to be positive even though 5 are not. acc_{prob} was either 67.4% or 67.5% (to 3 s.f.).

The results for the other datasets in Table 6, are more interesting. The key columns are $\sigma_{\text{acc}_{\text{max}}}$ and $\sigma_{\text{acc}_{\text{prob}}}$ which give the standard deviation in acc_{max} and acc_{prob} for MCMC samples produced from 3 different random seeds. In 15 out of the 16 cases the standard deviation is smaller if tempering is used. This reconfirms what many of the plots show: that tempering leads to a better approximation to the posterior.

8. Conclusion and discussion

The results in the previous section allow us to conclude that tempering delivers improved results for Bayesian C&RT (at least when there is some improvement to make). Since these results have been produced using a general SLP-based framework for Bayesian inference, we can immediately apply tempering to other models. We intend to begin with Bayesian networks.

However, it is worth considering why one might wish to take a Bayesian approach to C&RT in the first

Data	iter	t	$\overline{\text{acc}_{\text{max}}}$	$\sigma_{\text{acc}_{\text{max}}}$	$\overline{\text{acc}_{\text{prob}}}$	$\sigma_{\text{acc}_{\text{prob}}}$
BCW	50K	F	96.1%	1.9%	93.2%	1.2%
BCW	50K	T	96.6%	0.9%	93.1%	0.4%
BCW	125K	F	95.1%	0.3%	92.9%	0.7%
BCW	125K	T	97.1%	1.2%	93.5%	0.5%
BCW	250K	F	96.1%	1.2%	92.8%	1.4%
BCW	250K	T	95.8%	0.3%	93.3%	0.1%
PIMA	50K	F	76.5%	2.3%	67.2%	0.7%
PIMA	50K	T	73.4%	1.7%	66.3%	0.4%
PIMA	125K	F	73.8%	2.4%	66.4%	1.0%
PIMA	125K	T	74.3%	1.9%	66.5%	0.8%
PIMA	250K	F	76.9%	3.2%	66.7%	0.5%
PIMA	250K	T	73.6%	1.6%	66.9%	0.3%
LR	50K	F	62.4%	3.6%	30.3%	1.1%
LR	50K	T	66.9%	0.1%	32.9%	0.5%
WF	50K	F	71.0%	3.7%	55.6%	1.2%
WF	50K	T	72.5%	2.9%	57.5%	0.8%

Table 6. Comparing variation in predictive accuracy for 3 MCMC runs using different random seeds. First 3 columns are dataset, number of iterations and whether tempering was used. Last 4 columns are means and standard deviations in acc_{max} and acc_{prob}

place. Bayesian C&RT is much slower than the standard greedy algorithm, and with a poorly chosen prior there is no reason why it should lead to good predictive accuracy. To see this, note that for our 5 datasets the greedy algorithm (using the R `rpart` package with default parameters, except for `minbucket`=5) has accuracies of 75.0%, 95.5%, 76.4%, 46.1% and 74.1% on the hold-out sets for K, BCW, PIMA, LR and WF, respectively.

Except for LR, these figures are either slightly better or not significantly worse than those achieved using our best approximation to the posterior (250K iterations with tempering). Finally, it is harder for a human to interpret a sample of, say, 250,000 trees than a single tree as is returned by the greedy approach.

However, extracting the single most visited tree and the single maximum marginal likelihood tree can help with interpretation in the same way as the tree returned by the greedy algorithm. The latter tree is an estimate of the maximum posterior probability tree assuming a uniform prior. In addition, 2nd, 3rd etc placed trees can also be used. Concerning predictive accuracy, the key, of course, is *not* to choose an arbitrary prior but one that (ideally) encapsulates all relevant non-data information. Doing so means that the class posterior probabilities eventually produced are conditional on all the available information, and so should be the optimal probabilities on which to base decisions. Our use of SLPs is intended to provide a flexible way of expressing this prior knowledge, and elsewhere (Angelopoulos & Cussens, 2005) we show that using prior knowledge does improve predictive accuracy. A final argument for the Bayesian approach is that making classification decisions with unequal misclassification costs is easy, as is drawing ROC curves, since we have class probabilities, not just predicted classes.

Acknowledgements

Thanks to our 3 anonymous reviewers. This work was supported by the UK EPSRC MathFIT project *Stochastic Logic Programs for MCMC* and also by *Applications of Probabilistic Inductive Logic Programming II* funded by the European Commission.

References

Altekar, G., Dwarkadas, S., Huelsenbeck, J. P., & Ronquist, F. (2004). Parallel Metropolis coupled Markov chain Monte Carlo for Bayesian phylogenetic inference. *Bioinformatics*, *20*, 407–415.

Angelopoulos, N., & Cussens, J. (2005). Exploiting informative priors for Bayesian classification and regression trees. *Proc. 19th International Joint Conference on AI (IJCAI-05)*. Edinburgh.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees*. New York: Chapman & Hall.

Chipman, H., George, E., & McCulloch, R. (2003). Bayesian treed generalized linear models. In J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman and A. F. M. Smith (Eds.), *Bayesian statistics 7*, 85–104. Oxford University Press.

Chipman, H. A., George, E. I., & McCulloch, R. E. (1998). Bayesian CART model search. *Journal of the American Statistical Association*, *39*, 935–960.

Denison, D. G. T., Holmes, C. C., Mallick, B. K., & Smith, A. F. M. (2002). *Bayesian methods for nonlinear classification and regression*. Wiley.

Geyer, C. J., & Thompson, E. A. (1995). Annealing Markov chain Monte Carlo with applications to ancestral inference. *Journal of the American Statistial Association*, *90*, 909–920.

Wolberg, W. H., & Mangasarian, O. L. (1990). Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proceedings of the National Academy of Science*, *87*, 9193–9196.