Top-down particle filtering for Bayesian decision trees: Supplementary material

Balaji Lakshminarayanan BALAJI@GATSBY.UCL.AC.UK
Gatsby Unit, CSML, University College London

Daniel M. Roy D.ROY@ENG.CAM.AC.UK
University of Cambridge

Yee Whye Teh Y.W.TEH@STATS.OX.AC.UK
Department of Statistics, University of Oxford

A. SMC algorithm

Algorithm 1 SMC for Bayesian decision tree learning

Inputs: Training data (X, Y)
Number of particles M

Initialize: $\mathsf{T}_0^{(m)} = E_0^{(m)} = \{\epsilon\}$
$\tau_0^{(m)} = \kappa_0^{(m)} = \emptyset$
$w_0^{(m)} = f(Y|\mathcal{T}_0^{(m)})$
$W_0 = \sum_m w_0^{(m)}$

for $i = 1$: MAX-STAGES **do**
  for $m = 1 : M$ **do**
    Sample $\mathcal{T}_i^{(m)}$ from $\mathbb{Q}_i(\cdot \mid \mathcal{T}_{i-1}^{(m)})$
      where $\mathcal{T}_i^{(m)} := (\mathsf{T}_i^{(m)}, \kappa_i^{(m)}, \tau_i^{(m)}, E_i^{(m)})$
    Update weights: (Here $\mathbb{P}, \mathbb{Q}_i$ denote their densities.)

$$w_i^{(m)} = \frac{\mathbb{P}(\mathcal{T}_i^{(m)})\, g(Y \mid \mathcal{T}_i^{(m)}, X)}{\mathbb{Q}_i(\mathcal{T}_i^{(m)} \mid \mathcal{T}_{i-1}^{(m)})\, \mathbb{P}(\mathcal{T}_{i-1}^{(m)})} \tag{1}$$

$$= w_{i-1}^{(m)} \frac{\mathbb{P}(\mathcal{T}_i^{(m)} \mid \mathcal{T}_{i-1}^{(m)})}{\mathbb{Q}_i(\mathcal{T}_i^{(m)} \mid \mathcal{T}_{i-1}^{(m)})} \frac{g(Y \mid \mathcal{T}_i^{(m)}, X)}{g(Y \mid \mathcal{T}_{i-1}^{(m)}, X)} \tag{2}$$

  end for
  Compute normalization: $W_i = \sum_m w_i^{(m)}$
  Normalize weights: $(\forall m)\ \bar{w}_i^{(m)} = w_i^{(m)} / W_i$
  if $\left(\sum_m (\bar{w}_i^{(m)})^2\right)^{-1} <$ ESS-THRESHOLD **then**
    $(\forall m)$ Resample indices j_m from $\sum_{m'} \bar{w}_i^{(m')} \delta_{m'}$
    $(\forall m)\ \mathcal{T}_i^{(m)} \leftarrow \mathcal{T}_i^{(j_m)};\ w_i^{(m)} \leftarrow W_i / M$
  end if
  if $(\forall m)\ E_i^{(m)} = \emptyset$ **then**
    exit for loop
  end if
end for
return Estimated marginal probability W_i / M and weighted samples $\{w_i^{(m)}, \mathsf{T}_i^{(m)}, \kappa_i^{(m)}, \tau_i^{(m)}\}_{m=1}^M$.

B. Effect of SMC proposal and expansion strategy on test accuracy

The results are shown in Figure 1.

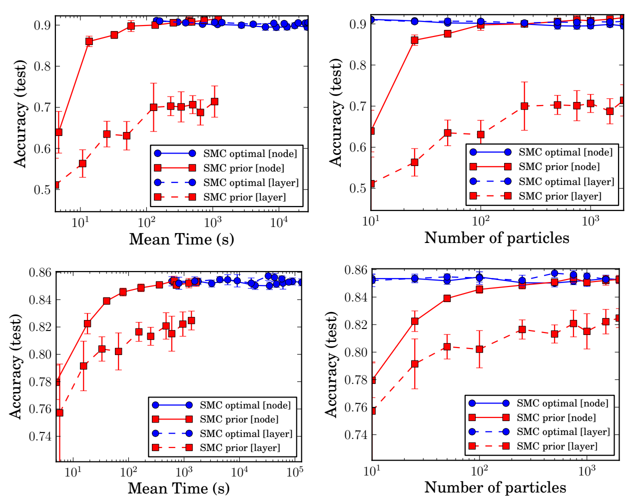

Figure 1. Results on *pen-digits* (top), and *magic-04* (bottom). Left column plots test accuracy vs runtime, while right column plots test accuracy vs number of particles. The blue circles and red squares represent *optimal* and *prior* proposals respectively. The solid and dashed lines represent *node-wise* and *layer-wise* proposals respectively.

C. Effect of the number of islands: *magic-04* dataset

The results are shown in Figure 2.

D. Marginal likelihood

The log marginal likelihood of the training data for different proposals is shown in Figure 3. As the num-

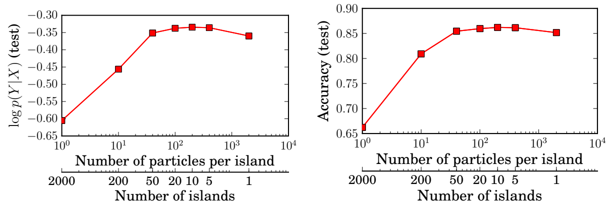

Figure 2. Results on *magic-04*: Test $\log p(y|x)$ (left) and accuracy (right) vs I and M/I for fixed $M = 2000$.

ber of particles increases, the log marginal likelihood of *prior* and *optimal* proposals converge to the same value (as expected).

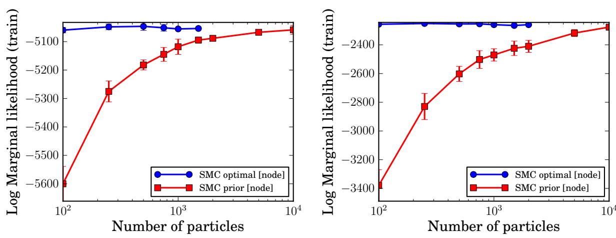

Figure 3. Results on *pen-digits* (left), and *magic-04* (right). Mean log marginal likelihood (i.e., mean $\log p(Y|X)$ for training data averaged across 10 runs) vs number of particles. The blue circles and red squares represent *optimal* and *prior* proposals respectively.

E. Sensitivity of results to choice of hyperparameters

In this experiment, we evaluate the sensitivity of the runtime vs predictive performance comparison between SMC (*prior* and *optimal* proposals), MCMC and CART to the choice of hyper parameters α (Dirichlet concentration parameter) and α_s, β_s (tree priors). We consider only *node-wise* expansion since it consistently outperformed *layer-wise* expansion in our previous experiments. In the first variant, we fix $\alpha = 5.0$ (since we do not expect it to affect the timing results) and vary the hyper parameters from $\alpha_s = 0.95, \beta_s = 0.5$ to $\boldsymbol{\alpha_s = 0.8, \beta_s = 0.2}$ (bold reflects changes) and also consider intermediate configurations $\alpha_s = 0.95, \boldsymbol{\beta_s = 0.2}$ and $\boldsymbol{\alpha_s = 0.8}, \beta_s = 0.5$. In the second variant, we fix $\alpha_s = 0.95, \beta_s = 0.5$ and set $\boldsymbol{\alpha = 1.0}$. Figures 4, 5, 6 and 7 display the results on *pen-digits* (top row), and *magic-04* (bottom row). The left column plots test $\log p(y|x)$ vs runtime, while the right column plots test accuracy vs runtime. The blue circles and red squares represent *optimal* and *prior* proposals respectively. Comparing the results to Figure 5 (in main text), we observe that the trends are qualitatively similar to those observed for $\alpha = 5.0, \alpha_s = 0.95, \beta_s = 0.5$ in Section 4.2 (in main text): (i) SMC consistently offers a better runtime vs predictive performance tradeoff than MCMC, (ii) the *prior* proposal offers a better runtime vs predictive performance tradeoff than the *optimal* proposal, (iii) $\alpha = 1.0$ leads to similar test accuracies as $\alpha = 5.0$ (the predictive probabilities are obviously not comparable).

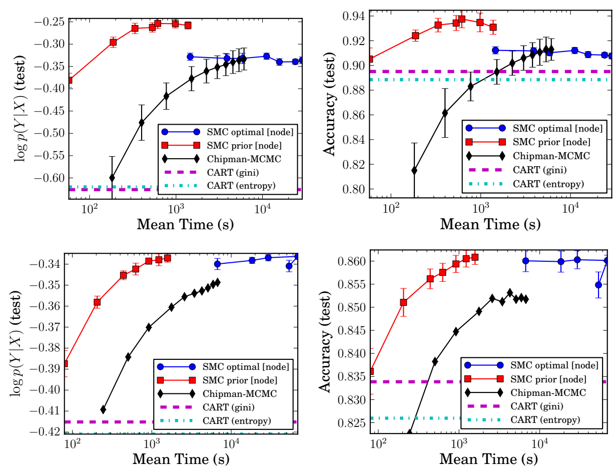

Figure 4. Hyperparameters: $\alpha = 5.0, \boldsymbol{\alpha_s = 0.8}, \beta_s = 0.5$ (see main text for additional information).

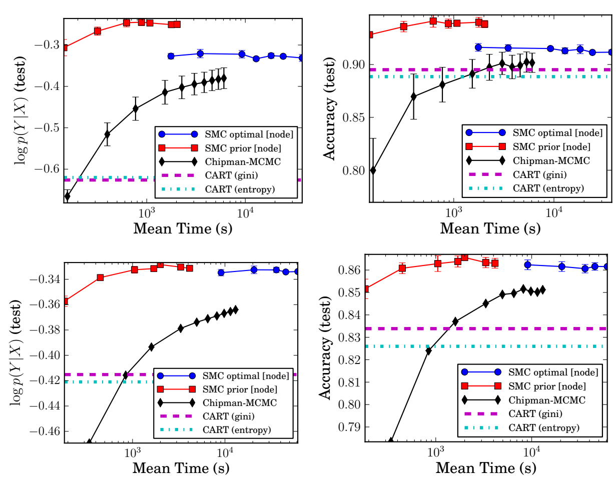

Figure 5. Hyperparameters: $\alpha = 5.0, \alpha_s = 0.95, \boldsymbol{\beta_s = 0.2}$ (see main text for additional information).

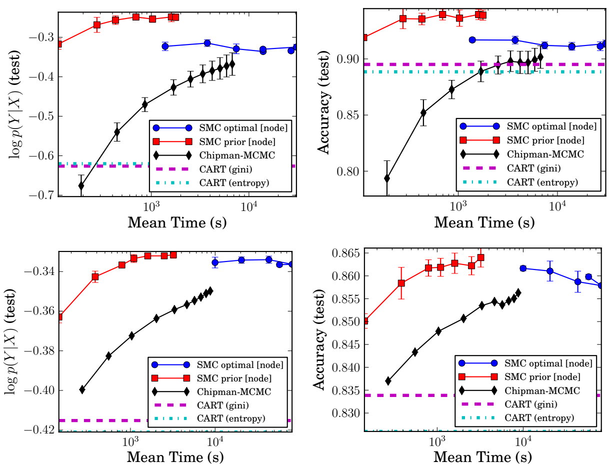

Figure 6. Hyperparameters: $\alpha = 5.0, \boldsymbol{\alpha_s = 0.8, \beta_s = 0.2}$ (see main text for additional information).

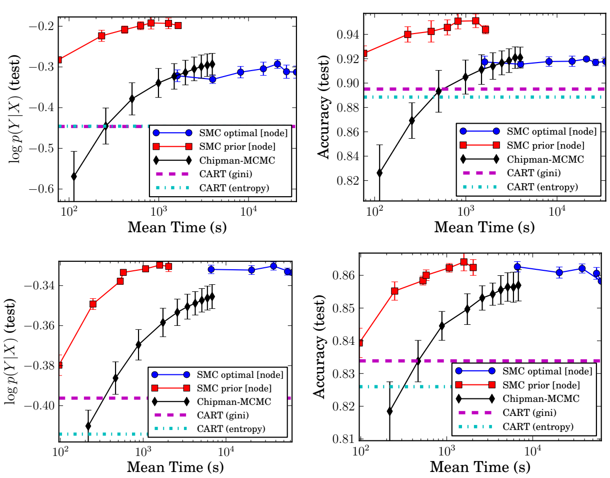

Figure 7. Hyperparameters: $\boldsymbol{\alpha = 1.0}, \alpha_s = 0.95, \beta_s = 0.5$ (see main text for additional information).