5

BAYESIAN OPTIMISATION BY CLASSIFICATION WITH DEEP LEARNING AND BEYOND

PREFACE

New material has been added to address the intricacies of how density-ratios relate to the probability of improvement (PI) and expected improvement (EI), and to examine the extension to this work known as likelihood-free BO (LFBO) [235].

5.1 INTRODUCTION

We introduced Bayesian optimisation (BO) in Section 2.5 as a highly effective approach for the global optimisation of expensive blackbox functions [20, 222]. In particular, we saw how BO proposes candidate solutions according to an *acquisition function* that encodes a degree of balance between exploration and exploitation. At the heart of BO lies a probabilistic surrogate model from which the acquisition function is derived.

Among the many acquisition functions that have been devised, the *improvement-based* ones, such as the PI and EI [114, 171] have remained prevalent due in large to their effectiveness despite their relative simplicity. Notably, while acquisition functions are generally challenging to compute, let alone optimise [283], PI/EI offers a closed-form expression when the posterior predictive density of the model follows a Gaussian distribution. However, while this condition makes these acquisition functions easier to work with, it can also preclude the use of richer families of models, as one must ensure analytical tractability of the predictive, often at the expense of expressiveness, or otherwise by resorting to sampling-based approximations [7].

By virtue of its flexibility, desirable conjugacy properties, and ability to produce well-calibrated predictive uncertainty, GP regression [279] is a widely-used probabilistic model in BO. To extend GP-based BO to problems with discrete variables [76], structures with conditional dependencies [113], or to capture nonstationary phenomenon [232], it is common to apply simple modifications to the covariance function, as this can often be done without compromising the tractability of the predictive. Suffice it to say, certain estimators, such as decision trees in the case of discrete variables, are naturally better equipped to deal with these scenarios. Indeed, to scale BO to problem settings that produce vast numbers of observations, such as in transfer learning [249], existing approaches have resorted to alternative model families like

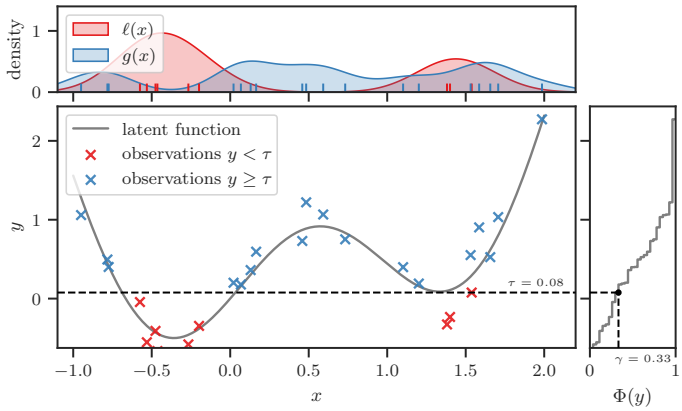


Figure 5.1: Optimizing a synthetic function $f(x) = \sin(3x) + x^2 - 0.7x$ with observation noise $\varepsilon \sim \mathcal{N}(0, 0.2^2)$. In the main pane, the noise-free function is represented by the solid gray curve, and $N = 27$ noisy observations are represented by the crosses '$\times$'. Observations with output y in the top-performing $\gamma = 1/3$ proportion are shown in *red*; otherwise, they are shown in *blue*. Their corresponding densities, $\ell(x)$ and $g(x)$, respectively, are shown in the top pane. Bayesian optimisation by density-ratio estimation (BORE) exploits the correspondence between the PI acquisition function and the *ratio* of densities $\ell(x)/g(x)$.

random forests (RFS) [108] and BNNS [194, 231, 237]. However, these are often bound by constraints and simplifying assumptions, or must rely on Monte Carlo (MC) methods that make the acquisition function more cumbersome to evaluate and optimise.

Recognizing that the surrogate model primarily serves as a means to construct the acquisition function, we shift the usual focus away from the model and toward the acquisition function itself. To this end, we seek an alternative formulation of the acquisition function, specifically, one that potentially opens the door to more powerful estimators for which the predictive density would otherwise be unwieldy or simply intractable to compute. In particular, Bergstra et al. [14] demonstrate that the PI function[1] can be expressed as the *relative* ratio between two densities [285]. To estimate this ratio, they propose a method known as the TPE, which naturally handles discrete and tree-structured inputs, and scales linearly with the number of observations. However, in spite of its many advantages, TPE is not without deficiencies.

In the work summarised in this chapter, we make the following contributions: (i) We revisit the TPE approach from first principles and identify its shortcomings in tackling the general DRE problem (Section 5.2). (ii) We propose a simple yet powerful alternative that casts the computation of PI as probabilistic classification (Section 5.3).

1 In fact, they make the stronger claim that this holds true for EI, but this assertion could be considered the outcome of spurious mathematical reasoning [74] – we elaborate on these intricacies in Section 5.2.2.2.

This approach is built on the aforementioned link between PI and the relative density-ratio, and the correspondence between DRE and CPE. As such, it retains the strengths of the TPE method while mitigating many of its weaknesses. Perhaps most significantly, it enables one to leverage virtually any state-of-the-art classification method available. In Section 5.4, we demonstrate through extensive experiments that our approach competes well with these methods on a diverse range of problems.

5.2 OPTIMISATION POLICIES AND DENSITY-RATIO ESTIMATION

5.2.1 *Relative Density-Ratio*

We introduced the *ordinary* density-ratio earlier in Section 2.3. Now let us generalise this to what is commonly known as the *relative* density-ratio [285]. Namely, for a given pair of densities $\ell(\mathbf{x})$ and $g(\mathbf{x})$, their *γ-relative density-ratio* is defined as

$$r_\gamma(\mathbf{x}) \triangleq \frac{\ell(\mathbf{x})}{\gamma\ell(\mathbf{x}) + (1-\gamma)g(\mathbf{x})}, \tag{5.1}$$

where $\gamma\ell(\mathbf{x}) + (1-\gamma)g(\mathbf{x})$ denotes the *γ-mixture density* with mixing proportion $0 \leq \gamma < 1$. Note that for $\gamma = 0$, we recover the ordinary density-ratio, which we denote $r_0(\mathbf{x}) \triangleq {}^{\ell(\mathbf{x})}/_{g(\mathbf{x})}$. Further, observe that the relative ratio is related to the ordinary ratio, $r_\gamma(\mathbf{x}) = h_\gamma(r_0(\mathbf{x}))$, where

$$h_\gamma(u) \triangleq \left(\gamma + u^{-1}(1-\gamma)\right)^{-1}$$

for $u > 0$.

5.2.2 *Improvement-based Acquisition Functions*

We now discuss how the improvement-based acquisition functions introduced in Section 2.5.2 relate to the ratio in Equation (5.1). First, let the threshold τ be the γ-th quantile of the observed y values, $\tau \triangleq \Phi^{-1}(\gamma)$ where $\gamma = \Phi(\tau) \triangleq p(y \leq \tau; \mathcal{D}_N)$. Thereafter, let the pair of densities be defined as $\ell(\mathbf{x}) \triangleq p(\mathbf{x} \,|\, y \leq \tau; \mathcal{D}_N)$ and $g(\mathbf{x}) \triangleq p(\mathbf{x} \,|\, y > \tau; \mathcal{D}_N)$. An illustrated example of this is shown in Figure 5.1.

5.2.2.1 *Probability of Improvement as a Density-Ratio*

Recall from Section 2.5.2.1 that the PI criterion can be expressed as $\alpha_{\text{PI}}(\mathbf{x}; \mathcal{D}_N, \tau) = p(y \leq \tau \,|\, \mathbf{x}, \mathcal{D}_N)$. By Bayes' rule, we have

$$\alpha_{\text{PI}}(\mathbf{x}; \mathcal{D}_N, \tau) = \frac{p(\mathbf{x} \,|\, y \leq \tau; \mathcal{D}_N)p(y \leq \tau \,|\, \mathcal{D}_N)}{p(\mathbf{x} \,|\, \mathcal{D}_N)}.$$

By definition, the numerator is simply

$$p(\mathbf{x} \,|\, y \leq \tau; \mathcal{D}_N)p(y \leq \tau \,|\, \mathcal{D}_N) = \gamma \cdot \ell(\mathbf{x}),$$

while, similarly, the denominator is

$$\begin{aligned} p(\mathbf{x} \mid \mathcal{D}_N) &= \int_{-\infty}^{\infty} p(\mathbf{x} \mid y, \mathcal{D}_N) p(y \mid \mathcal{D}_N) \, \mathrm{d}y \\ &= \ell(\mathbf{x}) \int_{-\infty}^{\tau} p(y \mid \mathcal{D}_N) \, \mathrm{d}y + g(\mathbf{x}) \int_{\tau}^{\infty} p(y \mid \mathcal{D}_N) \, \mathrm{d}y \\ &= \gamma \ell(\mathbf{x}) + (1 - \gamma) g(\mathbf{x}). \end{aligned} \tag{5.2}$$

Hence, the PI function can be expressed as the relative density-ratio, up to a constant factor γ,

$$\alpha_{\mathrm{PI}}\left(\mathbf{x}; \mathcal{D}_N, \Phi^{-1}(\gamma)\right) \propto r_\gamma(\mathbf{x}). \tag{5.3}$$

Crucially, this reduces the problem of maximizing PI to that of maximizing the relative density-ratio,

$$\mathbf{x}_{N+1} = \operatorname*{arg\,max}_{\mathbf{x} \in \mathcal{X}} \alpha_{\mathrm{PI}}\left(\mathbf{x}; \mathcal{D}_N, \Phi^{-1}(\gamma)\right) = \operatorname*{arg\,max}_{\mathbf{x} \in \mathcal{X}} r_\gamma(\mathbf{x}). \tag{5.4}$$

To estimate the unknown relative density-ratio, one can appeal to a wide variety of approaches from the DRE literature [245]. We broadly refer to this strategy as Bayesian optimisation by density-ratio estimation (BORE).

5.2.2.2 *Expected Improvement as a Density-Ratio?*

Bergstra et al. [14] assert that, under certain additional assumptions, the EI function can similarly be expressed as the relative density-ratio up to some constant factor. It goes without saying that this directly contradicts the results we have just presented, since clearly PI and EI are by definition not equivalent.

This particular issue has sparked recent discussions, and we analyse the arguments here. We proceed by reproducing the original derivations of Bergstra et al. [14]. Recall from Equation (2.45) that the EI function is defined as the expectation of the improvement utility function $U_{\mathrm{EI}}(y, \tau)$ over the posterior predictive density $p(y \mid \mathbf{x}, \mathcal{D}_N)$. Expanding this out, we have

$$\begin{aligned} \alpha_{\mathrm{EI}}(\mathbf{x}; \mathcal{D}_N, \tau) &\triangleq \mathbb{E}_{p(y \mid \mathbf{x}, \mathcal{D}_N)}[U_{\mathrm{EI}}(y, \tau)] \\ &= \int_{-\infty}^{\infty} U_{\mathrm{EI}}(y, \tau) p(y \mid \mathbf{x}, \mathcal{D}_N) \, \mathrm{d}y \\ &= \int_{-\infty}^{\tau} (\tau - y) p(y \mid \mathbf{x}, \mathcal{D}_N) \, \mathrm{d}y \\ &= \frac{1}{p(\mathbf{x} \mid \mathcal{D}_N)} \int_{-\infty}^{\tau} (\tau - y) p(\mathbf{x} \mid y, \mathcal{D}_N) p(y \mid \mathcal{D}_N) \, \mathrm{d}y. \end{aligned}$$

We've already simplified the denominator $p(\mathbf{x} \mid \mathcal{D}_N)$ in Equation (5.2), and the numerator simplifies to

$$\begin{aligned}
&\int_{-\infty}^{\tau} (\tau - y) p(\mathbf{x} \mid y, \mathcal{D}_N) p(y \mid \mathcal{D}_N) \, \mathrm{d}y \\
&\quad \approx \ell(\mathbf{x}) \int_{-\infty}^{\tau} (\tau - y) p(y \mid \mathcal{D}_N) \, \mathrm{d}y \\
&\quad = \ell(\mathbf{x}) \left(\tau \int_{-\infty}^{\tau} p(y \mid \mathcal{D}_N) \, \mathrm{d}y - \int_{-\infty}^{\tau} y p(y \mid \mathcal{D}_N) \, \mathrm{d}y \right) \\
&\quad = K \cdot \ell(\mathbf{x}),
\end{aligned} \tag{5.5}$$

where

$$K \triangleq \gamma\tau - \int_{-\infty}^{\tau} y p(y \mid \mathcal{D}_N) \, \mathrm{d}y.$$

In contrast with the original derivation, there is not a strict equality in Equation (5.5) because, in general, $p(\mathbf{x} \mid y, \mathcal{D}_N) \neq p(\mathbf{x} | y \leq \tau; \mathcal{D}_N) = \ell(\mathbf{x})$. That is to say, the conditional $p(\mathbf{x} \mid y, \mathcal{D}_N)$ is not constant wrt to y. While Garnett [74] perceives this as a "minor mathematical error" on the part of Bergstra et al. [14], it may also be intepreted as a strong simplifying modelling assumption. Specifically, the assumption states that $p(\mathbf{x} \mid y, \mathcal{D}_N)$ is piecewise constant where $p(\mathbf{x} \mid y, \mathcal{D}_N) = \ell(\mathbf{x})$ for $y \leq \tau$. This approximation is not unreasonable, especially when τ is in close proximity to the global minimum y^*.

The interested reader is referred to the issues thread on the public GitHub repository associated with the BO textbook by Garnett [74] for further discussion. The discourse is further extended by Song and Ermon [234] who subsequently proposed an alternative method [235] that encompasses, in a stricter sense, both PI/EI, and, more generally, any acquisition function that assumes the form of the expected utility in Equation (2.45). The approach is named likelihood-free BO (LFBO) by virtue of its ability to sidestep the cumbersome calculations that such acquisition functions often entail. As we shall see in Section 5.3.3, LFBO is similar to BORE in spirit, but distinct in a few mathematical particulars.

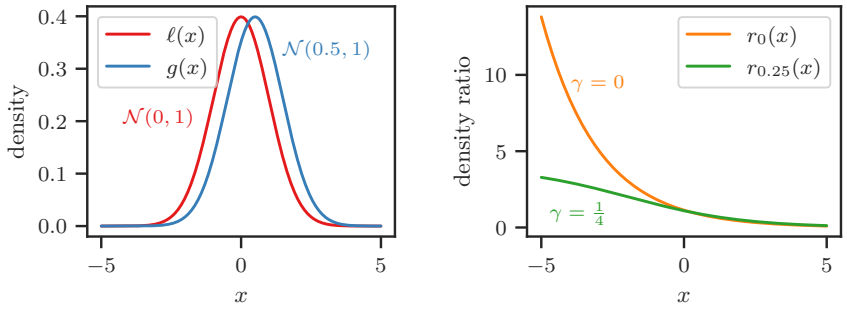


Figure 5.2: Gaussian densities (left) and their γ-relative density-ratios (right), which diverges when $\gamma = 0$ and converges to 4 when $\gamma = 1/4$.

5.2.3 *Tree-structured Parzen Estimator*

The tree-structured Parzen estimator (TPE) [14] is an instance of the BORE framework that seeks to solve the optimisation problem of Equation (5.4) by taking the following approach:

1. Since $r_\gamma(\mathbf{x}) = h_\gamma(r_0(\mathbf{x}))$ where h_γ is strictly non-decreasing, focus instead on maximizing[2] $r_0(\mathbf{x})$,

$$\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathcal{X}} r_0(\mathbf{x}).$$

2. Estimate the ordinary density-ratio $r_0(\mathbf{x})$ by separately estimating its constituent numerator $\ell(\mathbf{x})$ and denominator $g(\mathbf{x})$, using a tree-based variant of KDE [227].

It is not hard to see why TPE might be favorable compared to methods based on GP regression – one now incurs an $\mathcal{O}(N)$ computational cost as opposed to the $\mathcal{O}(N^3)$ cost of GP posterior inference. Furthermore, it is equipped to deal with tree-structured, mixed continuous, ordered, and unordered discrete inputs. In spite of its advantages, TPE is not without shortcomings.

5.2.4 *Potential Pitfalls*

The shortcomings of this approach are already well-documented in the DRE literature [245]. Nonetheless, we reiterate here a select few that are particularly detrimental in the context of global optimisation. Namely, the first major drawback of TPE lies within step 1:

SINGULARITIES. Relying on the ordinary density-ratio can result in numerical instabilities since it is unbounded – often diverging to infinity, even in simple toy scenarios (see Figure 5.2 for a simple example). In contrast, the γ-relative density-ratio is always bounded above by γ^{-1} when $\gamma > 0$ [285]. The other potential problems of TPE lie within step 2:

VAPNIK'S PRINCIPLE. Conceptually, independently estimating the densities is actually a more cumbersome approach that violates Vapnik's principle – namely, that when solving a problem of interest, one should refrain from solving a more general problem as an intermediate step [267]. In this instance, *density* estimation is a more general problem that is arguably more difficult than *density-ratio* estimation [120].

2 $r_0(\mathbf{x})$ denotes $\gamma = 0$ solely in $r_\gamma(\mathbf{x})$ of Equation (5.1) – it does *not* signify threshold $\tau \triangleq \Phi^{-1}(0)$, which would lead to density $\ell(\mathbf{x})$ containing no mass. We address this subtlety in Section 5.A.

KERNEL BANDWIDTH. KDE depends crucially on the selection of an appropriate kernel bandwidth, which is notoriously difficult [190, 223]. Furthermore, even with an optimal selection of a single fixed bandwidth, it cannot simultaneously adapt to low- and high-density regions [250].

ERROR SENSITIVITY. These difficulties are exacerbated by the fact that one is required to select *two* bandwidths, whereby the optimal bandwidth for one individual density is not necessarily appropriate for estimating the density-*ratio* – indeed, it may even have deleterious effects. This also makes the approach unforgiving to misspecification of the respective estimators, particularly in that of the denominator $g(\mathbf{x})$, which has a disproportionately large influence on the resulting density-ratio.

CURSE OF DIMENSIONALITY. For these reasons and more, KDE often falls short in high-dimensional regimes. In contrast, direct DRE methods have consistently been shown to scale better with dimensionality [244].

OPTIMISATION. Ultimately, we care not only about *estimating* the density-ratio, but also *optimizing* it wrt to inputs for the purpose of candidate suggestion. Being nondifferentiable, the ratio of TPEs is cumbersome to optimise.

5.3 BAYESIAN OPTIMISATION BY PROBABILISTIC CLASSIFICATION

We propose a different approach to BORE, importantly, one that circumvents the issues of TPE, by seeking to *directly* estimate the unknown ratio $r_\gamma(\mathbf{x})$.

As we alluded to in Section 2.3, there exists a multitude of direct DRE methods. Here, we focus on the conceptually simple and widely-used method based n class-probability estimation (CPE) [15, 37, 166, 197, 245], which we first introduced in Section 2.3.2. In this section, we extend the analysis to the more general case of the *relative* density-ratio, and to settings in which the classification problem is *unbalanced*.

First, let $\pi(\mathbf{x}) = p(z = 1 \mid \mathbf{x})$ denote the *class-posterior probability*, where z is the binary class label

$$z \triangleq \begin{cases} 1 & \text{if } y \leq \tau, \\ 0 & \text{if } y > \tau. \end{cases}$$

By definition, we have $\ell(\mathbf{x}) = p(\mathbf{x} \mid z = 1)$ and $g(\mathbf{x}) = p(\mathbf{x} \mid z = 0)$. We plug these into Equation (5.1) and apply Bayes' rule, letting the $p(\mathbf{x})$ terms cancel each other out to give

$$r_\gamma(\mathbf{x}) = \frac{p(z=1 \mid \mathbf{x})}{p(z=1)} \left(\gamma \cdot \frac{p(z=1 \mid \mathbf{x})}{p(z=1)} + (1-\gamma) \cdot \frac{p(z=0 \mid \mathbf{x})}{p(z=0)} \right)^{-1} \tag{5.6}$$

Since, by definition, $p(z=1) = \gamma$, Equation (5.6) simplifies to

$$r_\gamma(\mathbf{x}) = \gamma^{-1} \pi(\mathbf{x}). \tag{5.7}$$

Refer to Section 5.B for derivations. Thus, Equation (5.7) establishes the link between the class-posterior probability and the relative density-ratio. In particular, the latter is equivalent to the former up to constant factor γ^{-1}.

The astute reader will recognise from Equation (5.3) that, in fact,

$$\alpha_{\text{PI}}\left(\mathbf{x}; \mathcal{D}_N, \Phi^{-1}(\gamma)\right) = \gamma \cdot r_\gamma(\mathbf{x}) = \pi(\mathbf{x}).$$

Therefore, maximising the PI criterion amounts to maximising the class-posterior probability $\pi(\mathbf{x})$, which we can estimate using a probabilistic classifier – a function $\pi_{\boldsymbol{\theta}} : \mathcal{X} \to [0,1]$ parameterised by $\boldsymbol{\theta}$. To recover the true class-posterior probability, we minimise a proper scoring rule [83] such as the log loss

$$\hat{\mathcal{L}}(\boldsymbol{\theta}) \triangleq -\frac{1}{N} \left(\sum_{n=1}^{N} z_n \log \pi_{\boldsymbol{\theta}}(\mathbf{x}_n) \sum_{n=1}^{N} + (1-z_n) \log\left(1 - \pi_{\boldsymbol{\theta}}(\mathbf{x}_n)\right) \right). \tag{5.8}$$

Thereafter, we can use $\pi_{\boldsymbol{\theta}}(\mathbf{x})$ as a proxy to the PI criterion,

$$\pi_{\boldsymbol{\theta}}(\mathbf{x}) \approx \alpha_{\text{PI}}\left(\mathbf{x}; \mathcal{D}_N, \Phi^{-1}(\gamma)\right) \tag{5.9}$$

where the approximation is tight at $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \hat{\mathcal{L}}(\boldsymbol{\theta})$. Note $\hat{\mathcal{L}}$ is an unbiased estimate of the log loss $\mathcal{L}$ that first appeared in Equation (5.13). Refer to Section 5.C for details.

Hence, in the so-called BO loop (summarised in Algorithm 2), we alternately optimise (i) the classifier parameters $\boldsymbol{\theta}$ wrt to the log loss (to improve the approximation of Equation (5.9); Line 6), and (ii) the classifier input $\mathbf{x}$ wrt to its output (to suggest the next candidate to evaluate; Line 8).

In traditional GP-based PI, Line 8 typically consists of maximizing the PI criterion expressed in the form of Equation (2.48), while Line 6 consists of optimizing the GP hyperparameters wrt the marginal likelihood. By analogy with our approach, the parameterised function $\pi_{\boldsymbol{\theta}}(\mathbf{x})$ is *itself* an approximation to the PI criterion to be maximised directly, while the approximation is tightened through by optimizing the classifier parameters wrt the log loss. In short, we have reduced the problem of computing PI to that of learning a probabilistic classifier, thereby unlocking a broad range of estimators beyond those so

Algorithm 2: Bayesian optimisation by density-ratio estimation (BORE).

Input: blackbox $f : \mathcal{X} \to \mathbb{R}$, proportion $\gamma \in (0, 1)$, probabilistic classifier $\pi_{\boldsymbol{\theta}} : \mathcal{X} \to [0, 1]$.

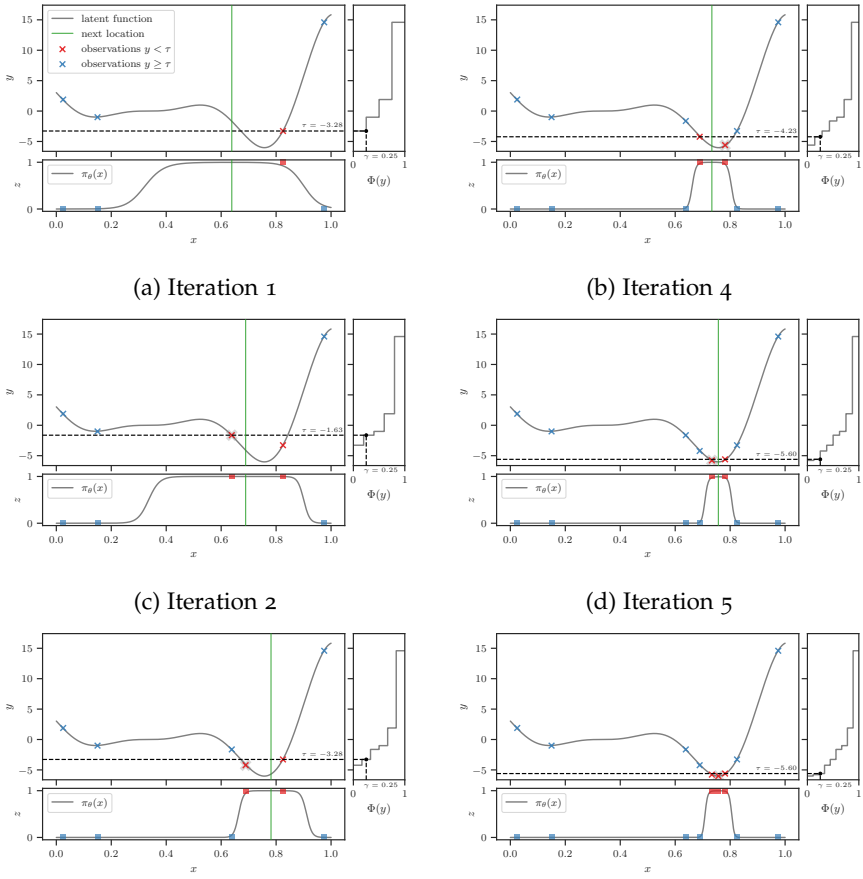
1 **while** *under budget* **do**
2 $\quad \tau \leftarrow \Phi^{-1}(\gamma)$ // compute γ-th quantile of $\{y_n\}_{n=1}^N$
3 $\quad z_n \leftarrow \mathbb{I}[y_n \leq \tau]$ for $n = 1, \dots, N$ // assign labels
4 $\quad \tilde{\mathcal{D}}_N \leftarrow \{(\mathbf{x}_n, z_n)\}_{n=1}^N$ // construct auxiliary dataset
5 $\quad$ /* update classifier by optimizing parameters $\boldsymbol{\theta}$ wrt log loss */
6 $\quad \boldsymbol{\theta}^* \leftarrow \arg\min_{\boldsymbol{\theta}} \hat{\mathcal{L}}(\boldsymbol{\theta})$ // depends on $\tilde{\mathcal{D}}_N$, see Equation (5.8)
7 $\quad$ /* suggest candidate by optimizing input $\mathbf{x}$ wrt classifier */
8 $\quad \mathbf{x}_N \leftarrow \arg\max_{\mathbf{x} \in \mathcal{X}} \pi_{\boldsymbol{\theta}^*}(\mathbf{x})$ // see Equation (5.9)
9 $\quad y_N \leftarrow f(\mathbf{x}_N)$ // evaluate blackbox function
10 $\quad \mathcal{D}_N \leftarrow \mathcal{D}_{N-1} \cup \{(\mathbf{x}_N, y_N)\}$ // update dataset
11 $\quad N \leftarrow N + 1$
12 **end**

far used in BO. Importantly, this enables one to employ virtually any state-of-the-art classification method available and to parameterise the classifier using arbitrarily expressive approximators that potentially have the capacity to deal with non-linear, non-stationary, and heteroscedastic phenomena frequently encountered in practice.

TOY 1D EXAMPLE. To illustrate, in Figure 5.3, we animate Algorithm 2 step by step on a synthetic problem for a half dozen iterations. Specifically, we minimise the FORRESTER function

$$f(x) \triangleq (6x - 2)^2 \sin(12x - 4),$$

in the domain $x \in [0, 1]$ with observation noise $\varepsilon \sim \mathcal{N}(0, 0.05^2)$. The algorithm is started with 4 random initial designs. Each subfigure depicts the state after Lines 6 and 8 – namely, after *updating* and *maximizing* the classifier, respectively. In every subfigure, the main pane depicts the noise-free function, represented by the *solid gray* curve, and the set of observations, represented by *crosses* '$\times$'. The location that was evaluated in the previous iteration is highlighted with a *gray outline*. The right pane shows the empirical cdf (ECDF) of the observed y values. The *vertical dashed black line* in this pane is located at $\gamma = \frac{1}{4}$. The *horizontal dashed black line* is located at τ, the value of y such that $\Phi(y) = \frac{1}{4}$, i. e., $\tau = \Phi^{-1}\left(\frac{1}{4}\right)$. The instances below this horizontal line are assigned binary label $z = 1$, while those above are assigned $z = 0$. This is visualised in the bottom pane, alongside the probabilistic classifier $\pi_{\boldsymbol{\theta}}(\mathbf{x})$, represented by the *solid gray* curve. Finally, the maximiser of the classifier is represented by the *vertical solid green* line – this denotes the location to be evaluated in the next iteration.



(a) Iteration 1

(b) Iteration 4

(c) Iteration 2

(d) Iteration 5

(e) Iteration 3

(f) Iteration 6

Figure 5.3: Step-by-step animation of Algorithm 2 on the FORRESTER synthetic problem.

5.3.1 *Choice of Proportion γ*

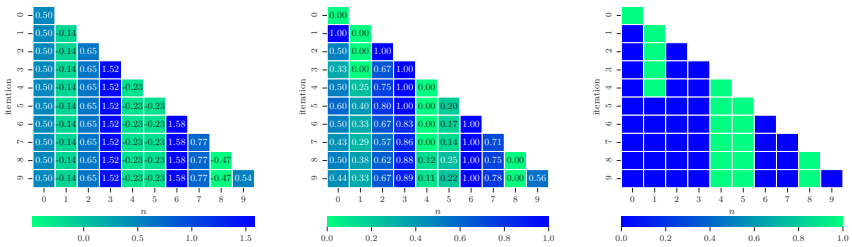
The proportion $\gamma \in (0, 1)$ influences the explore-exploit trade-off. Intuitively, a smaller setting of γ encourages exploitation and leads to fewer modes and sharper peaks in the acquisition function. To see this, consider that there are by definition fewer candidate inputs $\mathbf{x}$ for which its corresponding output y can be expected to improve over the first quartile ($\gamma = 1/4$) of the observed output values than, say, the third quartile ($\gamma = 3/4$). That being said, given that the class balance rate is by definition γ, a value too close to 0 may lead to instabilities in classifier learning. A potential strategy to combat this is to begin with a perfect balance ($\gamma = 1/2$) and then to decay γ as optimisation progresses.

In this work, we keep γ fixed throughout optimisation, which, on the other hand, has the benefit of providing guarantees about how the classification task evolves. In particular, in each iteration, after having observed a new evaluation, we are guaranteed that the binary label of *at most* one existing instance can flip. This property can be exploited

to make classifier learning of Line 6 more efficient. More specifically, assuming the proportion γ is fixed across iterations, then, in each iteration, we are guaranteed the following changes:

1. a new input and its corresponding output $(\mathbf{x}_N, y_N)$ will be added to the dataset, thus
2. creating a shift in the rankings and, by extension, quantiles of the observed y values, in turn
3. leading to the binary label of *at most* one instance to flip.

Therefore, between consecutive iterations, changes to the classification dataset are fairly incremental. One can leverage this to make classifier training more efficient, especially in families of classifiers for which re-training entirely from scratch in each iteration is superfluous and wasteful. See Figure 5.4 for an illustrative example, in which the task is to optimise a contrived, synthetic "noise-only" function $f(x) = 0$ with observation noise $\varepsilon \sim \mathcal{N}(0, 1)$, and the proportion is set to $\gamma = 1/4$.



(a) Continuous targets y_n (b) Empirical distribution $\Phi(y_n)$ (c) Binary labels z_n

Figure 5.4: Optimising a synthetic "noise-only" function. As we iterate through the BO loop from top to bottom, the array of targets grows from left to right. In each iteration, the size of the array increases by one, resulting in a re-shuffling of the rankings and, by extension, quantiles. This in turn leads to the label for *at most* one instance to flip. Hence, between consecutive iterations, changes to the classification dataset are fairly incremental. This property can be exploited to make classifier training more efficient in each iteration.

Some viable strategies for reducing per-iteration classier learning overhead may include speeding up convergence by (i) *importance sampling* (e. g.re-weighting new samples and those for which the label have flipped), (ii) *early-stopping* (stop training early if either the loss or accuracy have not changed for some number of epochs) and (iii) *annealing* (decaying the number of epochs or batch-wise training steps as optimisation progresses).

5.3.2 *Choice of Probabilistic Classifier*

We examine a few variations of BORE that differ in the choice of classifier and discuss their strengths and weaknesses across different global optimisation problem settings.

MULTI-LAYER PERCEPTRONS. We propose BORE-MLP, a variant based on MLPs. This choice is appealing not only for (i) its flexibility and universal approximation guarantees [104] but because (ii) one can easily adopt stochastic gradient descent (SGD) methods to scale up its parameter learning [138], and (iii) it is differentiable end-to-end, thus enabling the use of quasi-Newton methods such as L-BFGS [146] for candidate suggestion. Lastly, since SGD is online by nature, (iv) it is feasible to adapt weights from previous iterations instead of training from scratch. A notable weakness is that MLPs can be over-parameterised and therefore considerably data-hungry.

TREE-BASED ENSEMBLES. We consider two further variants: BORE-RF and BORE-XGB, both based on ensembles of decision trees – namely, random forest (RF) [19] and XGBOOST [34], respectively. These variants are attractive since they inherit from decision trees the ability to (i) deal with discrete and conditional inputs by design, (ii) work well in high-dimensions, and (iii) are scalable and easily parallelizable. Further, (iv) online extensions of RFs [216] may be applied to avoid training from scratch. A caveat is that, since their response surfaces are discontinuous and nondifferentiable, decision trees are difficult to maximise. Therefore, we appeal to random search and evolutionary strategies for candidate suggestion. Further details and a comparison of various approaches is included in Section 5.4.5.2.

In theory, for the approximation of Equation (5.9) to be tight, the classifier is required to produce well-calibrated probabilities [166]. A potential drawback of the BORE-RF variant is that RFs are generally not trained by minimizing a proper scoring rule. As such, additional techniques may be necessary to improve calibration [184].

GAUSSIAN PROCESSES. The last variant we consider is BORE-GP, based on a GP classifier (GPC) [278]. Like the GP regression model, GPC offers (i) a high degree of flexibility, at least on smooth functions up to moderate dimensionalities, and (ii) well-calibrated uncertainty estimates (useful for marginalizing out the hyperparameters from the acquisition function, as we discuss in Section 5.G). On the other hand, GPC not only loses one of the foremost appeals of GP regression, namely, analytical tractability of the predictive, but it is also not necessarily better equipped to deal with the more problematic settings we have discuseed (discrete variables, high-dimensionalities, etc..), and

its scalability is contingent on the choice of inference approximation utilised.

5.3.3 *Likelihood-Free BO by Weighted Classification*

We give a brief overview of the LFBO framework of Song and Ermon [234]. Recall from Section 2.3.1 that every convex, lower-semicontinuous function[3] f can be represented in terms of its convex dual $f^\star$,

$$f(u) = \max_s \left\{ uf'(s) - f^\star(f'(s)) \right\}$$

For any function $\alpha : \mathcal{X} \to \mathbb{R}$, we can leverage this variational representation to obtain a lower bound on expectations of the form $\mathbb{E}[f(\alpha(\mathbf{x}))]$,

$$\begin{aligned}
\mathbb{E}_{p(\mathbf{x})}\left[f(\alpha(\mathbf{x}))\right] &= \mathbb{E}_{p(\mathbf{x})}\left[\max_s \left\{\alpha(\mathbf{x})f'(s) - f^\star(f'(s))\right\}\right] \\
&\geq \max_s \mathbb{E}_{p(\mathbf{x})}\left[\alpha(\mathbf{x})f'(s) - f^\star(f'(s))\right] \\
&\geq \max_{S:\mathcal{X}\to\mathbb{R}} \mathbb{E}_{p(\mathbf{x})}\left[\alpha(\mathbf{x})f'(S(\mathbf{x})) - f^\star(f'(S(\mathbf{x})))\right]. \quad (5.10)
\end{aligned}$$

Using the convexity of f, it's easy to show that the maximiser of Equation (5.10) is

$$\begin{aligned}
S^* &\triangleq \arg\max_{S:\mathcal{X}\to\mathbb{R}} \mathbb{E}_{p(\mathbf{x})}\left[\alpha(\mathbf{x})f'(S(\mathbf{x})) - f^\star(f'(S(\mathbf{x})))\right] \\
&= \alpha
\end{aligned}$$

Let's now optimise Equation (5.10) over a family of functions parameterised by $\boldsymbol{\theta}$,

$$\boldsymbol{\theta}^* \triangleq \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{p(\mathbf{x})}\left[\alpha(\mathbf{x})f'(S_{\boldsymbol{\theta}}(\mathbf{x})) - f^\star(f'(S_{\boldsymbol{\theta}}(\mathbf{x})))\right].$$

This gives an approximation for function $\alpha(\mathbf{x})$,

$$S_{\boldsymbol{\theta}}(\mathbf{x}) \approx \alpha(\mathbf{x}),$$

that is tight at $\boldsymbol{\theta} = \boldsymbol{\theta}^*$. Suppose now the function of interest is the expected utility $\alpha(\mathbf{x}; \mathcal{D}_N, \tau) \triangleq \mathbb{E}_{p(y\,|\,\mathbf{x},\mathcal{D}_N)}[U(y;\tau)]$ from Equation (2.45). Then we have

$$\begin{aligned}
\boldsymbol{\theta}^* &\triangleq \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{p(\mathbf{x})}\left[\mathbb{E}_{p(y\,|\,\mathbf{x},\mathcal{D}_N)}[U(y;\tau)]f'(S_{\boldsymbol{\theta}}(\mathbf{x})) - f^\star(f'(S_{\boldsymbol{\theta}}(\mathbf{x})))\right] \\
&= \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{p(\mathbf{x},y)}\left[U(y;\tau)f'(S_{\boldsymbol{\theta}}(\mathbf{x})) - f^\star(f'(S_{\boldsymbol{\theta}}(\mathbf{x})))\right] \\
&= \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{p(\mathbf{x},y)}\left[U(y;\tau)\log \pi_{\boldsymbol{\theta}}(\mathbf{x}) + \log(1 - \pi_{\boldsymbol{\theta}}(\mathbf{x}))\right] \quad (5.11)
\end{aligned}$$

where we obtain Equation (5.11) by setting $\pi_{\boldsymbol{\theta}}(\mathbf{x}) \triangleq \sigma(\log S_{\boldsymbol{\theta}}(\mathbf{x}))$ and $f(u) \triangleq u\log u - (u+1)\log(u+1)$ from Equations (2.12) and (2.13).

3 we are overloading the notation f here, which has been used earlier in this chapter to denote the unknown blackbox function we're seeking to minimising

We can view Equation (5.11) as a weighted objective function for binary classification where the utility $U(y;\tau)$ acts as the nonnegative weight and $\pi_\theta(\mathbf{x})$ is the probabilistic classifier, as in Section 5.3. Finally, we obtain an approximation to the acquisition function,

$$S_\theta(\mathbf{x}) = \frac{\pi_\theta(\mathbf{x})}{1-\pi_\theta(\mathbf{x})} \approx \alpha(\mathbf{x};\tau).$$

Like with BORE, this circumvents the need to explicitly solve the integral in Equation (2.45) and thus places no restrictions on the form of $p(y \mid \mathbf{x}, \mathcal{D}_N)$. Furthermore, this approach can be applied to any utility function $U(y;\tau)$, not only PI, but EI, and beyond.

5.4 EXPERIMENTS

We describe the experiments conducted to empirically evaluate our method. To this end, we consider a variety of problems, ranging from automated machine learning (AUTOML), robotic arm control, to racing line optimisation.

We provide comparisons against a comprehensive selection of state-of-the-art baselines. Namely, across all problems, we consider random search (RS) [13], GP-BO (using EI with $\gamma = 0$) [114], TPE [14], and SMAC [108]. We also consider evolutionary strategies: differential evolution (DE) [243] for problems with continuous domains, and regularised evolution (RE) [204] for those with discrete domains. Further information about these baselines and the source code for their implementations are included in Section 5.D.

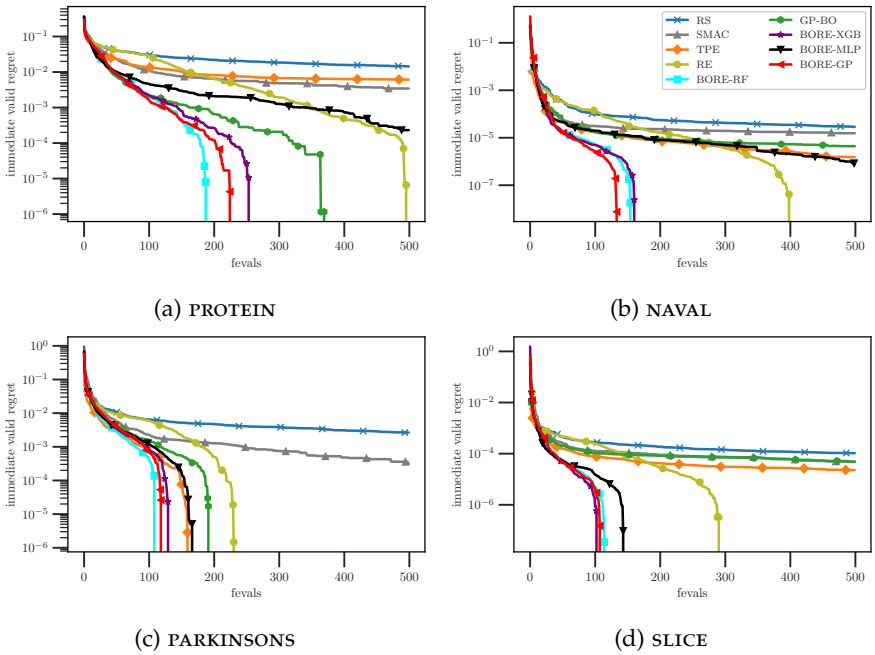
To quantitatively assess performance we report the *immediate regret* (in benchmarks for which the exact global minimum is known), defined as the absolute error between the global minimum and the lowest function value attained thus far. Unless otherwise stated we report, for each benchmark and method, results aggregated across 100 replicated runs.

We set $\gamma = 1/3$ across all variants and benchmarks. For candidate suggestion in the tree-based variants, we use RS with a function evaluation limit of 500 for problems with discrete domains, and DE with a limit of 2,000 for those with continuous domains. Our open-source implementation is available on GitHub at ltiao/bore. Further details concerning the experimental set-up and the implementation of each variant are included in Section 5.E.

5.4.1 *Neural Network Tuning (HPOBench)*

First, we consider the problem of training a two-layer feed-forward NN for regression. Specifically, a NN is trained for 100 epochs with the ADAM optimiser [123], and the objective is the validation MSE. The hyperparameters are the *initial learning rate*, *learning rate schedule*, *batch*

size, along with the layer-specific *widths*, *activations*, and *dropout rates*. We consider four datasets: PROTEIN, NAVAL, PARKINSONS, and SLICE, and utilise HPOBench [125] which tabulates, for each dataset, the MSEs resulting from all possible (62,208) configurations. Additional details are included in Section 5.F.1, and the results are shown in Figure 5.5. We see across all datasets that the BORE-RF and -XGB variants consis-

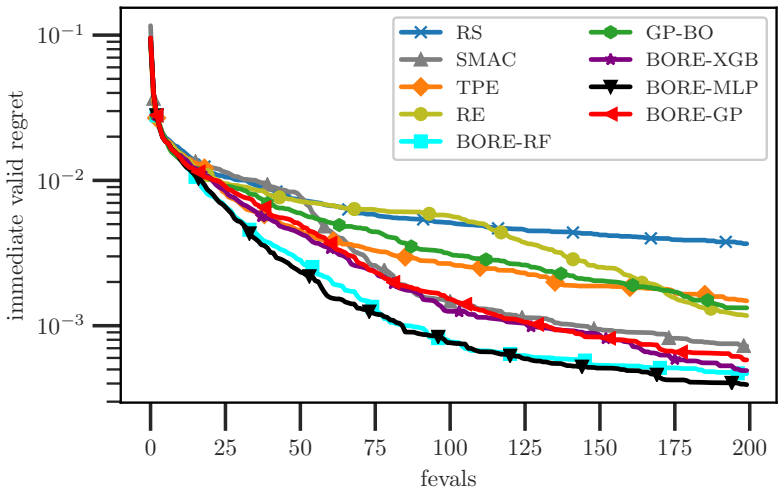


(a) PROTEIN (b) NAVAL

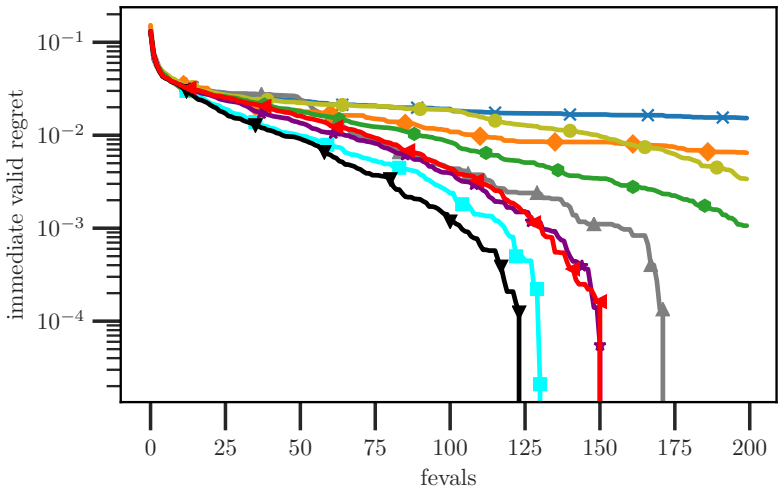
(c) PARKINSONS (d) SLICE

Figure 5.5: Immediate regret over function evaluations on the HPOBench neural network tuning problems ($D = 9$).

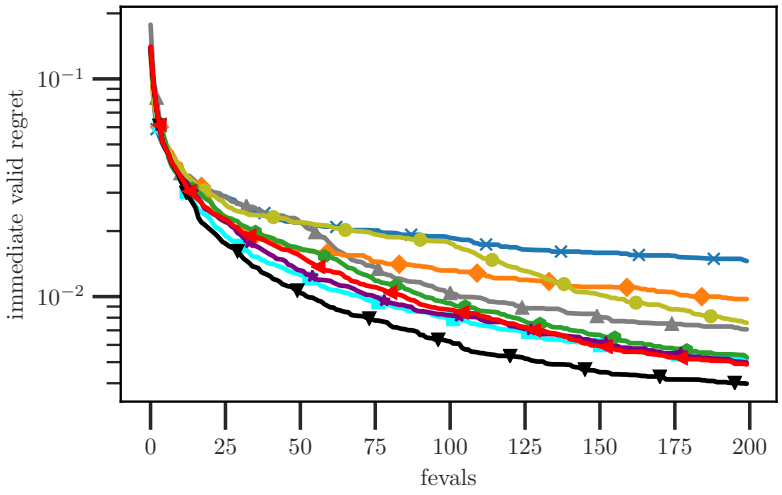
tently outperform all other baselines, converging rapidly toward the global minimum after 1-2 hundred evaluations – in some cases, earlier than any other baseline by over two hundred evaluations. Notably, with the exception being BORE-MLP on the PARKINSONS dataset, all BORE variants outperform TPE, in many cases by a sizable margin.



(a) CIFAR-10



(b) CIFAR-100



(c) ImageNet-16

Figure 5.6: Immediate regret over function evaluations on the NASBench201 NAS problems ($D = 6$).

5.4.2 *Neural Architecture Search (NASBench201)*

Next, we consider a NAS problem, namely, that of designing a neural cell. A cell is represented by a directed acyclic graph (DAG) with 4 nodes, and the task is to assign an *operation* to each of the 6 possible arcs from a set of five operations. We utilise NASBench201 [59], which tabulates precomputed results from all possible $5^6 = 15,625$ combinations for each of the three datasets: CIFAR-10, CIFAR-100 [129], and ImageNet-16 [42]. Additional details are included in Section 5.F.2, and the results are shown in Figure 5.6. We find across all datasets that the BORE variants consistently achieve the lowest final regret among all baselines. Not only that, the BORE variants, in particular BORE-MLP, maintains the lowest regret at anytime (i. e. at any optimisation iteration), followed by BORE-RF, then BORE-XGB/-GP. In this problem, the inputs are purely categorical, whereas in the previous problem they are a mix of categorical and ordinal. For the BORE-MLP variant, categorical inputs are one-hot encoded, while ordinal inputs are handled by simply rounding to their nearest integer index. The latter is known to have shortcomings [76], and might explain why BORE-MLP is the most effective variant in this problem but the least effective in the previous one.

5.4.3 *Robot Arm Pushing*

We consider the 14D control problem first studied by Wang and Jegelka [273]. The problem is concerned with tuning the controllers of robot hands to push objects to some desired locations. Specifically, there are two robots, each tasked with manipulating an object. For each robot, the control parameters include the *location* and *orientation* of its hands, the *moving direction*, *pushing speed*, and *duration*. Due to the prohibitively large number of function evaluations ($\sim$10,000) required to achieve reasonable performance, we omit all GP-based methods from our comparisons on this benchmark. Further, we reduce the number of replicated runs of each method to 50. Additional details are included in Section 5.F.3, and the results are shown in Figure 5.7. We see that BORE-XGB attains the highest reward, followed by BORE-RF and TPE (which attain roughly the same performance), and then BORE-MLP.

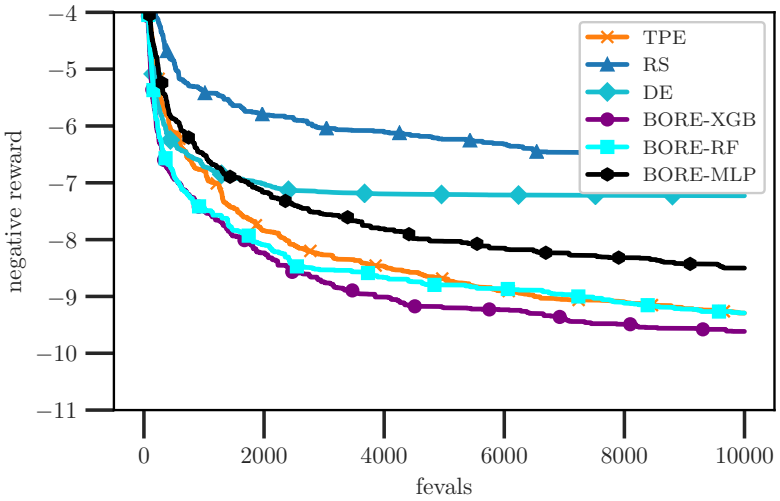
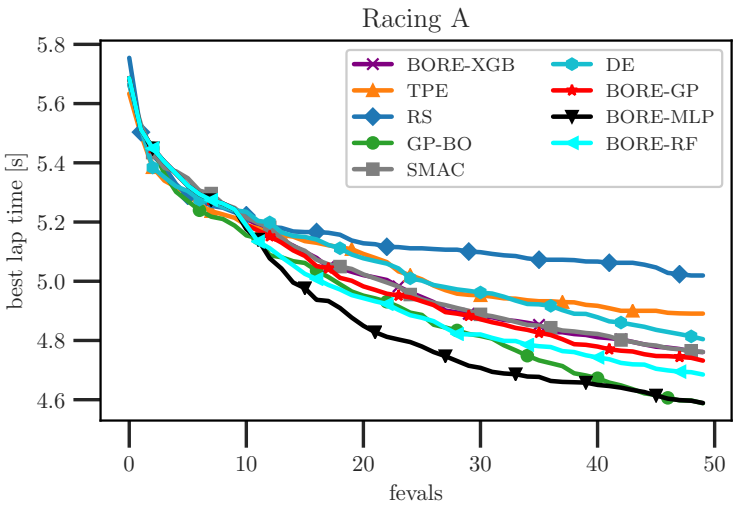
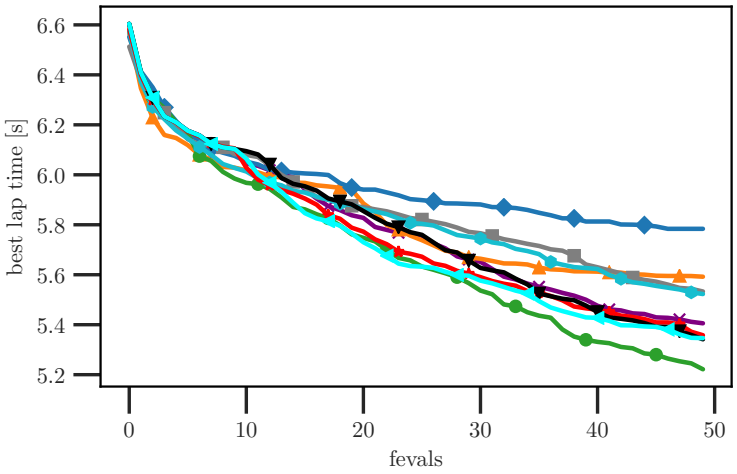


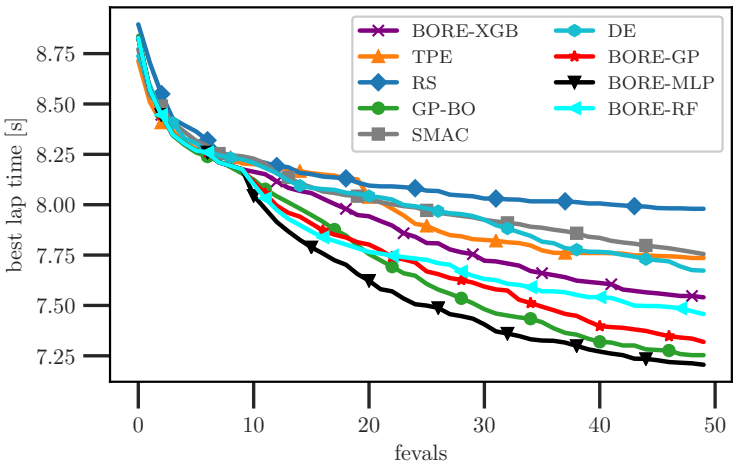
Figure 5.7: Negative reward over function evaluations on the Robot Pushing task ($D = 14$).



(a) UC BERKELEY ($D = 12$)



(b) ETH ZÜRICH A ($D = 20$)



(c) ETH ZÜRICH B ($D = 21$)

Figure 5.8: Best lap times (in seconds) over function evaluations in the racing line optimisation problem on various racetracks.

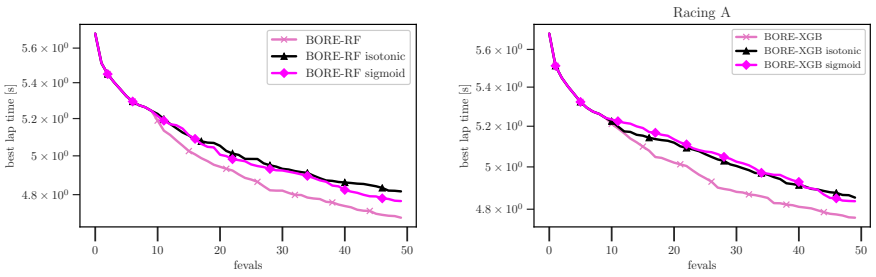
5.4.4 *Racing Line Optimisation.*

We consider the problem of computing the optimal racing line for a given track and vehicle with known dynamics. We adopt the set-up of Jain and Morari [110], who consider the dynamics of miniature scale cars traversing the tracks at UC BERKELEY and ETH ZÜRICH. The racing line is a trajectory determined by D waypoints placed along the length of the track, where the ith waypoint deviates from the centerline of the track by $x_i \in \left[-\frac{W}{2}, \frac{W}{2}\right]$ for some track width W. The task is to minimise the lap time $f(\mathbf{x})$, the minimum time required to traverse the trajectory parameterised by $\mathbf{x} = [x_1 \cdots x_D]^\top$. Additional details are included in Section 5.F.4, and the results are shown in Figure 5.8. First, we see that the BORE variants consistently outperform all baselines except for GP-BO. This is to be expected since the function is continuous, smooth, and has ∼20 dimensions or less. Nonetheless, we find that the BORE-MLP variant performs as well as, or marginally better than, GP-BO on two tracks. In particular, on the UC BERKELEY track, we see that BORE-MLP achieves the best lap times for the first ∼40 evaluations, and is caught up to by GP-BO in the final 10. On ETH ZÜRICH track B, BORE-MLP consistently maintains a narrow lead.

5.4.5 *Ablation Studies*

5.4.5.1 *Effects of calibration*

As discussed in Section 5.3.2, calibrating the classifier may have a profound effect on the tree-based variants of BORE, namely, BORE-RF and BORE-XGB. We consider two popular approaches [184], namely, Platt scaling [195] and isotonic regression [291, 292]. The results shown



(a) Calibrated RF classifiers.

(b) Calibrated XGBOOST classifiers.

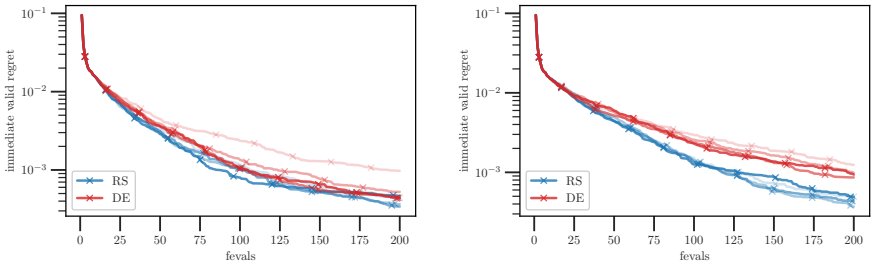
Figure 5.9: Effects of calibrating classifiers in the BORE-RF and BORE-XGB variants. Results of racing line optimisation on the UC BERKELEY track.

in Figure 5.9 suggest that applying these calibration techniques may actually have deleterious effects. However, this can also be adequately explained by overfitting due to insufficient calibration samples. In this particular problem, only a small number of function evaluations are required for convergence to the global minimum, so this produces

a small dataset with which to calibrate the classifier. In the case of isotonic regression, typically ∼1,000 samples are required. Therefore, it remains inconclusive whether calibration may in fact carry benefits, particularly in problem settings that produce large amounts of data.

5.4.5.2 *Effects of different strategies to maximise the acquisition function*

We examine different strategies for maximizing the acquisition function (i. e. the classifier) in the tree-based variants of BORE, namely, BORE-RF and BORE-XGB. Decision trees are difficult to maximise since their response surfaces are discontinuous and nondifferentiable. Hence, we consider the following methods: RS and DE. For each method, we further consider different evaluation budgets, i. e., limits on the number of evaluations of the acquisition function. Specifically, we consider the limits 50, 100, 200, and 500.



(a) BORE-RF on CIFAR-10 (b) BORE-XGB on CIFAR-10

Figure 5.10: A comparison of various acquisition optimisation strategies on the NASBench201 problem.

In Figures 5.10a and 5.10b, we show the results of BORE-RF and BORE-XGB, respectively, on the CIFAR-10 dataset of the NASBench201 benchmark, as described in Section 5.F.2. Each curve represents the mean across 100 repeated runs. The opacity is proportional to the function evaluation limit, with the most transparent having the lowest limit and the most opaque having the highest limit. We find that RS appears to outperform DE by a narrow margin. Additionally, for DE, a higher evaluation limit appears to be somewhat beneficial, while the opposite holds true for RS.

5.5 DISCUSSION

In this section, we discuss the limitations of our method and suggest potential approaches to address them. We also discuss the significant outcomes of this work at the time of writing and their potential impact.

EXPLORATION. Similar to the TPE method, BORE generally has a tendency to favor exploitation over exploration. In the case of TPE, the maximiser of the acquisition function $\ell(\mathbf{x})/g(\mathbf{x})$ will be located at the

mode of $\ell(\mathbf{x})$, which has mass concentrated around inputs for which its output value is within the smallest proportion γ of all observed output values (i. e. inputs with label $z = 1$). Recall the classical formulation of EI from Equation (2.50) in which the explore-exploit trade-off is explicitly encoded in mathematical terms. Assuming we had access to its global optimum, then by design the solution is a candidate that strikes a good balance between exploration and exploitation. Indeed, by virtue of having lower predictive uncertainty, previously evaluated candidates will tend to have lower acquisition values, which helps to encourage exploration. In contrast, for TPE and BORE, the previously evaluated candidates labeled $z = 1$ will tend to retain high acquisition values. Therefore, in the worst-case scenario, the global optimum of the acquisition function may become stuck at some local optimum of the blackbox function, or a point within some neighborhood thereof. In practice, implementations of TPE avoid this scenario by introducing stochasticity in the acquisition optimisation, e. g.by randomly sampling from $\ell(\mathbf{x})$ and suggesting the sample that maximises $\ell(\mathbf{x})/g(\mathbf{x})$. We surmise that BORE was able to avoid such pathological cases in our experiments due in part to the sources of randomness inherent to the acquisition optimisation method of choice.

A further detail to note is that the labels z do not remain static throughout optimisation. In other words, the classification dataset is different for each new iteration. Recall that, by construction, only a fraction γ of the observations can have positive labels $z = 1$. With each iteration, observing a new value of y leads to a change in the threshold τ. Since only a fraction γ of observations can lie below this threshold, the labels of existing observations must accordingly flip intermittently throughout optimisation. Thus, as the probabilistic classifier $\pi_{\theta}(\mathbf{x})$ adapts to these updates, the regions in which it outputs high probabilities will also shift accordingly. Consequently, the classifier response surface will either become multimodal (leading to exploration) or become narrower and more sharply-peaked in the same region (leading to exploitation).

Although not discussed in this thesis, the behavior described above can make simple ϵ-greedy strategies particularly effective at stimulating exploration. Follow-up work by Oliveira, Tiao, and Ramos [187] has considered batch extensions using Stein variational gradient descent (SVGD) [147], which encourages greater diversity in the query batch and provides theoretical guarantees.

HYPERPARAMETER ESTIMATION. Firstly, a noteworthy consequence of seeking to directly approximate PI under its alternative formulation is that the classifier parameters θ in BORE can be interpreted as *hyperparameters* (in the same way that the parameters of the GP kernel are hyperparameters), a deterministic treatment of which based on point estimates can often be viable. For example, in the BORE-MLP

variant, $\boldsymbol{\theta}$ consists of the layer weights, which we are able to estimate using type-II maximum likelihood. In contrast, to utilise NNs in traditional BO, generally the layer weights $\boldsymbol{\omega}$ are parameters that must first be marginalised out in order to compute the predictive $p_{\boldsymbol{\theta}}(y \mid \mathbf{x}, \mathcal{D}_N) = \int p_{\boldsymbol{\theta}}(y \mid \mathbf{x}, \boldsymbol{\omega}) p_{\boldsymbol{\theta}}(\boldsymbol{\omega} \mid \mathcal{D}_N) \, \mathrm{d}\boldsymbol{\omega}$, while the hyperparameters $\boldsymbol{\theta}$, consisting of e. g.the prior and likelihood precisions, may optionally be marginalised out as well (though usually point estimates suffice). Refer to Section 5.G for an expanded discussion on this distinction. As with the GP hyperparameters in GP-BO, in order to encourage exploration, it may be beneficial to consider placing a prior on $\boldsymbol{\theta}$ and marginalizing out its uncertainty [230]. Further, compared against GP-BO, a potential downside of BORE is that there may be vastly more *meta-hyperparameters* settings from which to choose. Whereas in GP-BO these might consist of, e. g.the choice of kernel and its isotropy, there are potentially many more possibilities in BORE. In BORE-MLP, this may consist of, e. g.layer depth, widths, activations, etc. – the tuning of which is often the reason one appeals to BO in the first place. While we obtained remarkable results with the proposed variants without needing to deviate from the sensible defaults, in general, for further improvements in calibration and sample diversity, it may be beneficial to consider marginalizing out even the meta-hyperparameters [274].

Impact to Date

We discuss the impactful outcomes of this work to date. First, we examine the real-world uses of BORE. Despite the emergence of various players in the field, toolkits such as hyperopt and Optuna still remain the most widely used for HPO, especially in the domain of AUTOML. These libraries rely foremost on the TPE method as their default search algorithm. Indeed, in many settings, e. g., those of high-dimensionality, TPE often manages outperforms other paradigms such as evolutionary strategies or traditional GP-based BO. Having addressed several of the most profound shortcomings of TPE, BORE has proved, in turn, to consistently outperform TPE. This was not only demonstrated in the original paper but has been independently observed in subsequent works. Thus seen, BORE stands poised to be an ideal candidate to replace TPE as the leading method for hyperparameter search. In fact, BORE has already been adopted as one of the primary search algorithms in SyneTune [218], a rapidly growing open-source framework for HPO developed by Amazon Research.

Second, we highlight BORE as a new research avenue. Less than a year since its initial publication, it has garnered recognition and is set to be featured in the upcoming textbook on Bayesian optimisation by Garnett [74], scheduled to be published later this year. Furthermore, prominent research labs have already embarked on extending BORE's

capabilities, such as extensions to multiple objectives [52] and generalisations to other acquisition functions [235], as we saw in Section 5.3.3.

5.6 SUMMARY

We have presented a novel methodology for BO based on the relationship between improvement-based acquisition functions and probabilistic classifier. This observation is made through the well-known link between CPE and DRE, and the lesser-known insight that PI can be expressed as a relative density-ratio between two unknown distributions.

We discussed important ways in which TPE, an early attempt to exploit the latter link, falls short. Further, we demonstrated that our CPE-based approach to BORE, in particular, our variants based on the MLP, RF, XGBOOST, and GP classifiers, consistently outperform TPE, and compete well against the state-of-the-art derivative-free global optimisation methods.

Overall, the simplicity and effectiveness of BORE make it a promising approach for black-box optimisation, and its high degree of extensibility provides numerous exciting avenues for future work.

5.A RELATIVE DENSITY-RATIO: UNABRIDGED NOTATION

In Section 5.2, for notational simplicity, we had excluded the dependencies of ℓ, g and r_γ on τ. Let us now define these densities more explicitly as

$$\ell(\mathbf{x};\tau) \triangleq p\left(\mathbf{x} \,|\, y \leq \tau, \mathcal{D}_N\right), \qquad \text{and} \qquad g(\mathbf{x};\tau) \triangleq p\left(\mathbf{x} \,|\, y > \tau, \mathcal{D}_N\right),$$

and accordingly, the γ-relative density-ratio from Equation (5.1) as

$$r(\mathbf{x};\gamma,\tau) = \frac{\ell(\mathbf{x};\tau)}{\gamma\ell(\mathbf{x};\tau) + (1-\gamma)g(\mathbf{x};\tau)}.$$

Recall from Equation (5.3) that

$$a_{\text{PI}}\left(\mathbf{x};\mathcal{D}_N,\Phi^{-1}(\gamma)\right) \propto r\left(\mathbf{x};\gamma,\Phi^{-1}(\gamma)\right). \tag{5.12}$$

In step 1, Bergstra et al. [14] resort to optimizing $r(\mathbf{x};0,\Phi^{-1}(\gamma))$, which is justified by the fact that

$$r(\mathbf{x};\gamma,\Phi^{-1}(\gamma)) = h_\gamma\left[r\left(\mathbf{x};0,\Phi^{-1}(\gamma)\right)\right],$$

for strictly nondecreasing h_γ. Note we have used a blue and orange color coding to emphasise the differences in the setting of γ (best viewed on a computer screen). Recall that $\Phi^{-1}(0) = \min_n y_n$ corresponds to the conventional setting of threshold τ. However, make no mistake, for any $\gamma > 0$,

$$a_{\text{PI}}\left(\mathbf{x};\mathcal{D}_N,\Phi^{-1}(0)\right) \not\propto r\left(\mathbf{x};0,\Phi^{-1}(\gamma)\right).$$

Therefore, given the numerical instabilities associated with this approach, as discussed in Section 5.2.4, there is no advantage to be gained from taking this direction. Moreover, Equation (5.12) only holds for $\gamma > 0$. To see this, suppose $\gamma = 0$, which gives

$$a_{\text{PI}}\left(\mathbf{x};\mathcal{D}_N,\Phi^{-1}(0)\right) \propto r\left(\mathbf{x};0,\Phi^{-1}(0)\right).$$

However, since by definition $\ell\left(\mathbf{x};\Phi^{-1}(0)\right)$ has no mass, the RHS is undefined.

5.B CLASS-POSTERIOR PROBABILITY

We provide an unabridged derivation of the identity of Equation (5.7). First, the γ-relative density ratio is given by

$$\begin{aligned} r_\gamma(\mathbf{x}) &\triangleq \frac{\ell(\mathbf{x})}{\gamma\ell(\mathbf{x}) + (1-\gamma)g(\mathbf{x})} \\ &= \frac{p(\mathbf{x} \mid z=1)}{\gamma \cdot p(\mathbf{x} \mid z=1) + (1-\gamma) \cdot p(\mathbf{x} \mid z=0)} \\ &= \left(\frac{p(z=1 \mid \mathbf{x})\cancel{p(\mathbf{x})}}{p(z=1)}\right)\left(\gamma \cdot \frac{p(z=1 \mid \mathbf{x})\cancel{p(\mathbf{x})}}{p(z=1)} + (1-\gamma) \cdot \frac{p(z=0 \mid \mathbf{x})\cancel{p(\mathbf{x})}}{p(z=0)}\right)^{-1}. \end{aligned}$$

By construction, we have $p(z=1) \triangleq p(y \leq \tau) = \gamma$ and $\pi(\mathbf{x}) \triangleq p(z = 1 \mid \mathbf{x})$. Therefore,

$$\begin{aligned} r_\gamma(\mathbf{x}) &= \gamma^{-1}\pi(\mathbf{x})\left(\cancel{\gamma} \cdot \frac{\pi(\mathbf{x})}{\cancel{\gamma}} + \cancel{(1-\gamma)} \cdot \frac{1-\pi(\mathbf{x})}{\cancel{1-\gamma}}\right)^{-1} \\ &= \gamma^{-1}\pi(\mathbf{x}). \end{aligned}$$

Alternatively, we can also arrive at the same result by writing the ordinary density ratio $r_0(\mathbf{x})$ in terms of $\pi(\mathbf{x})$ and γ, which is well-known to be

$$r_0(\mathbf{x}) = \left(\frac{\gamma}{1-\gamma}\right)^{-1} \frac{\pi(\mathbf{x})}{1-\pi(\mathbf{x})}.$$

Plugging this into function h_γ, we get

$$\begin{aligned} r_\gamma(\mathbf{x}) = h_\gamma(r_0(\mathbf{x})) &= h_\gamma\left(\left(\frac{\gamma}{1-\gamma}\right)^{-1} \frac{\pi(\mathbf{x})}{1-\pi(\mathbf{x})}\right) \\ &= \left(\gamma + (1-\gamma)\left(\frac{\gamma}{1-\gamma}\right)\left(\frac{\pi(\mathbf{x})}{1-\pi(\mathbf{x})}\right)^{-1}\right)^{-1} \\ &= \gamma^{-1}\left(1 + \left(\frac{\pi(\mathbf{x})}{1-\pi(\mathbf{x})}\right)^{-1}\right)^{-1} \\ &= \gamma^{-1}\pi(\mathbf{x}). \end{aligned}$$

5.C LOG LOSS

Recall from Section 2.3.2 that the log loss, also known as the binary cross-entropy (BCE) loss, is given by

$$\mathcal{L}(\boldsymbol{\theta}) \triangleq -\beta \cdot \mathbb{E}_{\ell(\mathbf{x})}[\log \pi_{\boldsymbol{\theta}}(\mathbf{x})] - (1-\beta) \cdot \mathbb{E}_{g(\mathbf{x})}[\log(1 - \pi_{\boldsymbol{\theta}}(\mathbf{x}))]. \quad (5.13)$$

The astute reader will have noticed that, unlike before, we've introduced multipliers involving β, which denotes the class balance rate, to account for potential class imbalance. Indeed, we recover the log loss first introduced in Equation (2.11) when $\beta = 1/2$. In particular, let

N_ℓ and N_g be the sizes of the support of $\ell(\mathbf{x})$ and $g(\mathbf{x})$, respectively. Then, we have

$$\beta = \frac{N_\ell}{N}, \qquad \text{and} \qquad 1 - \beta = \frac{N_g}{N},$$

where $N = N_\ell + N_g$. In practice, we approximate the log loss $\mathcal{L}(\boldsymbol{\theta})$ by the empirical risk of Equation (5.8), given by

$$\hat{\mathcal{L}}(\boldsymbol{\theta}) \triangleq -\frac{1}{N}\left(\sum_{n=1}^{N} z_n \log \pi_{\boldsymbol{\theta}}(\mathbf{x}_n) + (1 - z_n) \log\left(1 - \pi_{\boldsymbol{\theta}}(\mathbf{x}_n)\right)\right).$$

In this section, we show that the approximation of Equation (5.9), that is,

$$\pi_{\boldsymbol{\theta}}(\mathbf{x}) \approx \gamma \cdot r_\gamma(\mathbf{x}),$$

attains equality at $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$.

5.C.1 *Optimum*

Taking the functional derivative of $\mathcal{L}^*$ in Equation (5.13), we get

$$\begin{aligned}\frac{\partial \mathcal{L}^*}{\partial \pi_{\boldsymbol{\theta}}} &= -\mathbb{E}_{\ell(\mathbf{x})}\left[\frac{\beta}{\pi_{\boldsymbol{\theta}}(\mathbf{x})}\right] + \mathbb{E}_{g(\mathbf{x})}\left[\frac{1-\beta}{1-\pi_{\boldsymbol{\theta}}(\mathbf{x})}\right] \\ &= \int \left(-\beta \frac{\ell(\mathbf{x})}{\pi_{\boldsymbol{\theta}}(\mathbf{x})} + (1-\beta)\frac{g(\mathbf{x})}{1-\pi_{\boldsymbol{\theta}}(\mathbf{x})}\right) \mathrm{d}\mathbf{x}\end{aligned}$$

This integral evaluates to zero iff the integrand itself evaluates to zero. Hence, we solve the following for $\pi_{\boldsymbol{\theta}^*}(\mathbf{x})$,

$$\beta \frac{\ell(\mathbf{x})}{\pi_{\boldsymbol{\theta}^*}(\mathbf{x})} = (1-\beta)\frac{g(\mathbf{x})}{1-\pi_{\boldsymbol{\theta}^*}(\mathbf{x})}.$$

We re-arrange this expression to give

$$\frac{1-\pi_{\boldsymbol{\theta}^*}(\mathbf{x})}{\pi_{\boldsymbol{\theta}^*}(\mathbf{x})} = \left(\frac{1-\beta}{\beta}\right)\frac{g(\mathbf{x})}{\ell(\mathbf{x})} \qquad \Leftrightarrow \qquad \frac{1}{\pi_{\boldsymbol{\theta}^*}(\mathbf{x})} - 1 = \frac{\beta\ell(\mathbf{x}) + (1-\beta)g(\mathbf{x})}{\beta\ell(\mathbf{x})} - 1.$$

Finally, we add one to both sides and invert the result to give

$$\begin{aligned}\pi_{\boldsymbol{\theta}^*}(\mathbf{x}) &= \frac{\beta\ell(\mathbf{x})}{\beta\ell(\mathbf{x}) + (1-\beta)g(\mathbf{x})} \\ &= \beta \cdot r_\beta(\mathbf{x}).\end{aligned}$$

Since, by definition $\beta = \gamma$, this leads to $\pi_{\boldsymbol{\theta}^*}(\mathbf{x}) = \gamma \cdot r_\gamma(\mathbf{x})$ as required.

5.C.2 *Empirical Risk Minimisation*

For completeness, we show that the log loss $\mathcal{L}(\boldsymbol{\theta})$ of Equation (5.13) can be approximated by $\hat{\mathcal{L}}(\boldsymbol{\theta})$ of Equation (5.8). First, let ρ be the permutation of the set $\{1, \ldots, N\}$, i.e. the bijection from $\{1, \ldots, N\}$

to itself, such that $y_{\rho(n)} \leq \tau$ if $0 < \rho(n) \leq N_\ell$, and $y_{\rho(n)} > \tau$ if $N_\ell < \rho(n) \leq N_g$. That is to say,

$$\mathbf{x}_{\rho(n)} \sim \begin{cases} \ell(\mathbf{x}) & \text{if } 0 < \rho(n) \leq N_\ell, \\ g(\mathbf{x}) & \text{if } N_\ell < \rho(n) \leq N_g. \end{cases} \quad \text{and} \quad z_{\rho(n)} \triangleq \begin{cases} 1 & \text{if } 0 < \rho(n) \leq N_\ell, \\ 0 & \text{if } N_\ell < \rho(n) \leq N_g. \end{cases}$$

Then, we have

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}) &\triangleq -\frac{1}{N}\left(N_\ell \cdot \mathbb{E}_{\ell(\mathbf{x})}[\log \pi_{\boldsymbol{\theta}}(\mathbf{x})] + N_g \cdot \mathbb{E}_{g(\mathbf{x})}[\log(1-\pi_{\boldsymbol{\theta}}(\mathbf{x}))]\right) \\ &\approx -\frac{1}{N}\left(\cancel{N_\ell} \cdot \frac{1}{\cancel{N_\ell}} \sum_{n=1}^{N_\ell} \log \pi_{\boldsymbol{\theta}}(\mathbf{x}_{\rho(n)}) + \cancel{N_g} \cdot \frac{1}{\cancel{N_g}} \sum_{n=N_\ell+1}^{N_g} \log(1-\pi_{\boldsymbol{\theta}}(\mathbf{x}_{\rho(n)}))\right) \\ &= -\frac{1}{N}\left(\sum_{n=1}^{N} z_{\rho(n)} \log \pi_{\boldsymbol{\theta}}(\mathbf{x}_{\rho(n)}) + (1-z_{\rho(n)}) \log(1-\pi_{\boldsymbol{\theta}}(\mathbf{x}_{\rho(n)}))\right) \\ &= -\frac{1}{N}\left(\sum_{n=1}^{N} z_n \log \pi_{\boldsymbol{\theta}}(\mathbf{x}_n) + (1-z_n) \log(1-\pi_{\boldsymbol{\theta}}(\mathbf{x}_n))\right) \triangleq \hat{\mathcal{L}}(\boldsymbol{\theta}), \end{aligned}$$

as required.

5.D IMPLEMENTATION OF BASELINES

The software implementations of the baseline methods considered in our comparisons are described in Table 5.D.1.

Table 5.D.1: Implementations of baseline methods.

Method	Software Library	URL (github.com/*)	Notes
TPE	HyperOpt	hyperopt/hyperopt	
SMAC	SMAC3	automl/SMAC3	
GP-BO	AutoGluon	awslabs/autogluon	in `autogluon.searcher.GPFIFOSearcher`
DE	-	-	custom implementation
RE	NASBench-101	automl/nas_benchmarks	in `experiment_scripts/run_regularized_evolution.py`

5.E EXPERIMENTAL SET-UP AND IMPLEMENTATION DETAILS

HARDWARE. In our experiments, we employ `m4.xlarge` AWS EC2 instances, which have the following specifications:

- **CPU:** Intel(R) Xeon(R) E5-2676 v3 (4 Cores) @ 2.4 GHz
- **Memory:** 16GiB (DDR3)

SOFTWARE. Our method is implemented as a *configuration generator* plug-in in the HpBandSter library of Falkner, Klein, and Hutter [67]. The code will be released as open-source software upon publication.

The implementations of the classifiers on which the proposed variants of BORE are based are described in Table 5.E.1.

Table 5.E.1: Implementations of classifiers.

Model	Software Library	URL
Multi-layer perceptron (MLP)	Keras	`keras.io`
Random forest (RF)	scikit-learn	`scikit-learn.org`
Extreme gradient-boosting (XGBOOST)	XGBoost	`xgboost.readthedocs.io`

We set out with the aim of devising a practical method that is not only agnostic to the of choice classifier, but also robust to underlying implementation details – down to the choice of algorithmic settings. Ideally, any instantiation of BORE should work well out-of-the-box without the need to tweak the sensible default settings that are typically provided by software libraries. Therefore, unless otherwise stated, we emphasise that made no effort was made to adjust any settings and all reported results were obtained using the defaults. For reproducibility, we explicitly enumerate them in turn for each of the proposed variants.

5.E.1 *BORE-RF*

We limit our description to the most salient hyperparameters. We do not deviate from the default settings which, at the time of this writing, are:

- *number of trees* – 100
- *minimum number of samples required to split an internal node* (`min_samples_split`) – 2
- *maximum depth* – unspecified (nodes are expanded until all leaves contain less than `min_samples_split` samples)

5.E.2 *BORE-XGB*

- *number of trees (boosting rounds)* – 100
- *learning rate* (η) – 0.3
- *minimum sum of instance weight (Hessian) needed in a child* (`min_child_weight`) – 1
- *maximum depth* – 6

5.E.3 *BORE-MLP*

In the BORE-MLP variant, the classifier is a MLP with 2 hidden layers, each with 32 units. We consistently found `elu` activations [43] to be particularly effective for lower-dimensional problems, with `relu` remaining otherwise the best choice. We optimise the weights with ADAM [123] using batch size of $B = 64$. For candidate suggestion, we optimise the input of the classifier wrt to its output using multi-started L-BFGS with three random restarts.

EPOCHS PER ITERATION. To ensure the training time on BO iteration N is nonincreasing as a function of N, instead of directly specifying the number of epochs (i. e. full passes over the data), we specify the number of (batch-wise gradient) steps S to train for in each iteration. Since the number of steps per epoch is $M = \lceil N/B \rceil$, the effective number of epochs on the N-th BO iteration is then $E = \lfloor S/M \rfloor$. For example, if $S = 800$ and $B = 64$, the number of epochs for iteration $N = 512$ would be $E = 100$. As another example, for all $0 < N \leq B$ (i. e. we have yet to observe enough data to fill a batch), we have $E = S = 800$. See Figure 5.E.1 for a plot of the effective number of epochs against iterations for different settings of batch size B and number of steps per epoch S. Across all our experiments, we fix $S = 100$.

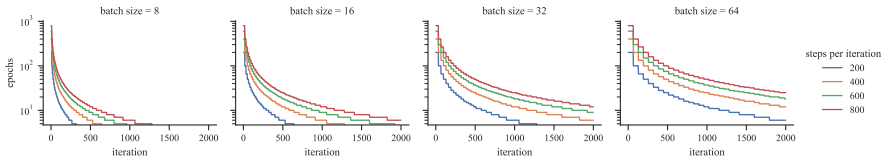


Figure 5.E.1: Effective number of epochs on the nth iteration for different settings of batch size B and number of steps per epoch S.

5.F DETAILS OF BENCHMARKS

5.F.1 *HPOBench*

The hyperparameters for the HPOBench problem and their ranges are summarised in Table 5.F.1. All hyperparameters are discrete – either ordered or unordered. All told, there are $6 \times 2 \times 4 \times 6 \times 2 \times 3 \times 6 \times 2 \times 3 = 66,208$ possible combinations. Further details on this problem can be found in [125].

5.F.2 *NASBench201*

The hyperparameters for the HPOBench problem and their ranges are summarised in Table 5.F.2. The operation associated with each of

Table 5.F.1: Configuration space for HPOBench.

Hyperparameter		Range
Initial learning rate (LR)		$\{5\times10^{-4}, 1\times10^{-3}, 5\times10^{-3}, 1\times10^{-2}, 5\times10^{-2}, 1\times10^{-1}\}$
LR schedule		{`cosine`, `fixed`}
Batch size		$\{2^3, 2^4, 2^5, 2^6\}$
Layer 1	Width	$\{2^4, 2^5, 2^6, 2^7, 2^8, 2^9\}$
	Activation	{`relu`, `tanh`}
	Dropout rate	$\{0.0, 0.3, 0.6\}$
Layer 2	Width	$\{2^4, 2^5, 2^6, 2^7, 2^8, 2^9\}$
	Activation	{`relu`, `tanh`}
	Dropout rate	$\{0.0, 0.3, 0.6\}$

Table 5.F.2: Configuration space for NASBench-201.

Hyperparameter	Range
Arc 0	{`none`, `skip-connect`, `conv-1` $\times$ `1`, `conv-3` $\times$ `3`, `avg-pool-3` $\times$ `3`}
Arc 1	{`none`, `skip-connect`, `conv-1` $\times$ `1`, `conv-3` $\times$ `3`, `avg-pool-3` $\times$ `3`}
Arc 2	{`none`, `skip-connect`, `conv-1` $\times$ `1`, `conv-3` $\times$ `3`, `avg-pool-3` $\times$ `3`}
Arc 3	{`none`, `skip-connect`, `conv-1` $\times$ `1`, `conv-3` $\times$ `3`, `avg-pool-3` $\times$ `3`}
Arc 4	{`none`, `skip-connect`, `conv-1` $\times$ `1`, `conv-3` $\times$ `3`, `avg-pool-3` $\times$ `3`}
Arc 5	{`none`, `skip-connect`, `conv-1` $\times$ `1`, `conv-3` $\times$ `3`, `avg-pool-3` $\times$ `3`}

the $\binom{4}{2} = 6$ arcs can belong to one of five categories. Hence, there are $5^6 = 15,625$ possible combinations of hyperparameter configurations. Further details on this problem can be found in [59].

5.F.3 *Robot pushing control*

This problem is concerned with tuning the controllers of two robot hands, with the goal of each pushing an object to some prescribed goal location $\mathbf{p}_g^{(1)}$ and $\mathbf{p}_g^{(2)}$, respectively. Let $\mathbf{p}_s^{(1)}$ and $\mathbf{p}_s^{(2)}$ denote the specified starting positions, and $\mathbf{p}_f^{(1)}$ and $\mathbf{p}_f^{(2)}$ the final positions (the latter of which are functions of the control parameters $\mathbf{x}$). The reward is defined as

$$R(\mathbf{x}) \triangleq \underbrace{\|\mathbf{p}_g^{(1)} - \mathbf{p}_s^{(1)}\| + \|\mathbf{p}_g^{(2)} - \mathbf{p}_s^{(2)}\|}_{\text{initial distances}} - \underbrace{(\|\mathbf{p}_g^{(1)} - \mathbf{p}_f^{(1)}\| + \|\mathbf{p}_g^{(2)} - \mathbf{p}_f^{(2)}\|)}_{\text{final distances}},$$

which effectively quantifies the amount of progress made toward pushing the objects to the desired goal. For each robot, the control parameters include the location and orientation of its hands, the pushing speed, moving direction and push duration. These parameters and their ranges are summarised in Table 5.F.3.

Further details on this problem can be found in [272]. This simulation is implemented with the Box2D library, and the associated code repository can be found at `https://github.com/zi-w/Ensemble-Bayesian-Optimisation`.

Table 5.F.3: Configuration space for the robot pushing control problem.

Hyperparameter		Range
Robot 1	Position x	$[-5, 5]$
	Position y	$[-5, 5]$
	Angle θ	$[0, 2\pi]$
	Velocity v_x	$[-10, 10]$
	Velocity v_y	$[-10, 10]$
	Push Duration	$[2, 30]$
	Torque	$[-5, 5]$
Robot 2	Position x	$[-5, 5]$
	Position y	$[-5, 5]$
	Angle θ	$[0, 2\pi]$
	Velocity v_x	$[-10, 10]$
	Velocity v_y	$[-10, 10]$
	Push Duration	$[2, 30]$
	Torque	$[-5, 5]$

5.F.4 *Racing Line Optimisation*

This problem is concerned with finding the optimal racing line. Namely, given a racetrack and a vehicle with known dynamics, the task is to determine a trajectory around the track for which the minimum time required to traverse it is minimal. We adopt the set-up of Jain and Morari [110], who consider 1:10 and 1:43 scale miniature remote-controlled cars traversing tracks at UC Berkeley [144] and ETH Zürich [214], respectively.

The trajectory is represented by a cubic spline parameterised by the 2D coordinates of D waypoints, each placed at locations along the length of the track, where the ith waypoint deviates from the centerline of the track by $x_i \in \left[-\frac{W}{2}, \frac{W}{2}\right]$, for some track width W. Hence, the parameters are the distances by which each waypoint deviates from the centerline, $\mathbf{x} = [x_1 \cdots x_D]^\top$.

Our blackbox function of interest, namely, the minimum time to traverse a given trajectory, is determined by the solution to a convex optimisation problem involving partial differential equations (PDEs) [145]. Further details on this problem can be found in [110], and the associated code repository can be found at `https://github.com/jainachin/bayesrace`.

5.G PARAMETERS, HYPERPARAMETERS, AND META-HYPERPARAMETERS

We explicitly identify the parameters $\boldsymbol{\omega}$, hyperparameters $\boldsymbol{\theta}$, and meta-hyperparameters $\boldsymbol{\lambda}$ in our approach, making clear their distinction, examining their roles in comparison with other methods and discuss their treatment.

Table 5.G.1: A taxonomy of parameters, hyperparameters, and meta-hyperparameters.

	BO with Gaussian processes (GPs)	BO with Bayesian neural networks (BNNs)	BORE with neural networks (NNs)
Meta-hyperparameters $\boldsymbol{\lambda}$	kernel family, kernel isotropy (ARD), etc..	layer depth, widths, activations, etc..	prior precision α, likelihood precision β, layer depth, widths, activations, etc..
Hyperparameters $\boldsymbol{\theta}$	kernel lengthscale and amplitude, ℓ and σ, likelihood precision β	prior precision α, likelihood precision β	weights $\mathbf{W}$, biases $\mathbf{b}$
Parameters $\boldsymbol{\omega}$	None $\varnothing$ (nonparametric)	weights $\mathbf{W}$, biases $\mathbf{b}$	None $\varnothing$ (by design)

5.G.1 *Parameters*

Since we seek to *directly* approximate the acquisition function, our method is by design free of parameters $\boldsymbol{\omega}$. By contrast, in classical BO, the acquisition function is derived from the analytical properties of the posterior predictive $p(y \mid \mathbf{x}, \boldsymbol{\theta}, \mathcal{D}_N)$. To compute this, the uncertainty about parameters $\boldsymbol{\omega}$ must be marginalised out

$$p(y \mid \mathbf{x}, \boldsymbol{\theta}, \mathcal{D}_N) = \int p(y \mid \mathbf{x}, \boldsymbol{\omega}, \boldsymbol{\theta}) p(\boldsymbol{\omega} \mid \boldsymbol{\theta}, \mathcal{D}_N) \, \mathrm{d}\boldsymbol{\omega}, \tag{5.14}$$

where

$$p(\boldsymbol{\omega} \mid \boldsymbol{\theta}, \mathcal{D}_N) = \frac{p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\omega}, \boldsymbol{\theta}) p(\boldsymbol{\omega} \mid \boldsymbol{\theta})}{p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta})}.$$

While GPs are free of parameters, the latent function values $\mathbf{f}$ must be marginalised out

$$p(y \mid \mathbf{x}, \boldsymbol{\theta}, \mathcal{D}_N) = \int p(y \mid \mathbf{x}, \mathbf{f}, \boldsymbol{\theta}) p(\mathbf{f} \mid \boldsymbol{\theta}, \mathcal{D}_N) \, \mathrm{d}\mathbf{f}.$$

In the case of GP regression, this is easily computed by applying straightforward rules of Gaussian conditioning. Unfortunately, few other models enjoy this convenience.

CASE STUDY: As a concrete example, consider BNNs. The parameters $\boldsymbol{\omega}$ consist of the weights $\mathbf{W}$ and biases $\mathbf{b}$ in the NN, while the hyperparameters $\boldsymbol{\theta}$ consist of the prior and likelihood precisions, α and β, respectively. In general, $p(\boldsymbol{\omega} \mid \mathcal{D}_N, \boldsymbol{\theta})$ is not analytically tractable.

- To work around this, DNGO [231] and ABLR [194] both constrain the parameters $\boldsymbol{\omega}$ to include the weights and biases of only the *final* layer, $\mathbf{W}_L$ and $\mathbf{b}_L$, and relegate those of all preceding layers, $\mathbf{W}_{1:L-1}$ and $\mathbf{b}_{1:L-1}$, to the hyperparameters $\boldsymbol{\theta}$. This yields an exact (Gaussian) expression for $p(\boldsymbol{\omega} \mid \mathcal{D}_N, \boldsymbol{\theta})$ and $p(y \mid \mathbf{x}, \boldsymbol{\theta}, \mathcal{D}_N)$. To treat the hyperparameters, Perrone et al. [194] estimate $\mathbf{W}_{1:L-1}$,

$\mathbf{b}_{1:L-1}$, α and β using type-II MLE, while Snoek et al. [231] use a combination of type-II MLE and slice sampling [181].

- In contrast, BOHAMIANN [237] makes no such simplifying distinctions regarding the layer weights and biases. Consequently, they must resort to sampling-based approximations of $p(\boldsymbol{\omega} \mid \boldsymbol{\theta}, \mathcal{D}_N)$, in their case by adopting SGHMC [33].

In both approaches, compromises needed to be made in order to negotiate the computation of $p(\boldsymbol{\omega} \mid \boldsymbol{\theta}, \mathcal{D}_N)$. This is not to mention the problem of computing the posterior over the *hyperparameters* $p(\boldsymbol{\theta} \mid \mathcal{D}_N)$, which we discuss next. In contrast, BORE avoids the problems associated with computing the posterior predictive $p(y \mid \mathbf{x}, \boldsymbol{\theta}, \mathcal{D}_N)$, and, by extension, the posterior $p(\boldsymbol{\omega} \mid \boldsymbol{\theta}, \mathcal{D}_N)$ of Equation (5.14). Therefore, such compromises are simply unnecessary.

5.G.2 *Hyperparameters*

For the sake of notational simplicity, we have thus far not been explicit about how the acquisition function depends on the hyperparameters $\boldsymbol{\theta}$ and how they are handled. We first discuss generically how hyperparameters $\boldsymbol{\theta}$ are treated in BO. Refer to [222] for a full discussion. In particular, we rewrite the acquisition function, expressed in Equation (2.45), to explicitly include $\boldsymbol{\theta}$

$$\alpha(\mathbf{x}; \boldsymbol{\theta}, \mathcal{D}_N, \tau) \triangleq \mathbb{E}_{p(y \mid \boldsymbol{\theta}, \mathbf{x}, \mathcal{D}_N)}[U(y; \tau)].$$

MARGINAL ACQUISITION FUNCTION. Ultimately, one wishes to maximise the *marginal* acquisition function $A(\mathbf{x}; \mathcal{D}_N, \tau)$, which marginalises out the uncertainty about the hyperparameters,

$$A(\mathbf{x}; \mathcal{D}_N, \tau) = \int \alpha(\mathbf{x}; \boldsymbol{\theta}, \mathcal{D}_N, \tau) p(\boldsymbol{\theta} \mid \mathcal{D}_N) \, \mathrm{d}\boldsymbol{\theta},$$

where

$$p(\boldsymbol{\theta} \mid \mathcal{D}_N) = \frac{p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathbf{y} \mid \mathbf{X})}.$$

This consists of an expectation over the posterior $p(\boldsymbol{\theta} \mid \mathcal{D}_N)$ which is, generally speaking, analytically intractable. In practice, the most straightforward way to compute $A(\mathbf{x}; \mathcal{D}_N, \tau)$ is to approximate the posterior using a delta measure centered at some point estimate $\hat{\boldsymbol{\theta}}$, either the type-II MLE $\hat{\boldsymbol{\theta}}_{\text{MLE}}$ or the MAP estimate $\hat{\boldsymbol{\theta}}_{\text{MAP}}$. This leads to

$$A(\mathbf{x}; \mathcal{D}_N, \tau) \approx \alpha(\mathbf{x}; \hat{\boldsymbol{\theta}}, \mathcal{D}_N, \tau).$$

Suffice it to say, sound uncertainty quantification is paramount to guiding exploration. Since point estimates fail to capture uncertainty about hyperparameters $\boldsymbol{\theta}$, it is often beneficial to turn instead to MC estimation [230]

$$A(\mathbf{x}; \mathcal{D}_N, \tau) \approx \frac{1}{S} \sum_{s=1}^{S} \alpha(\mathbf{x}; \boldsymbol{\theta}^{(s)}, \mathcal{D}_N, \tau), \qquad \boldsymbol{\theta}^{(s)} \sim p(\boldsymbol{\theta} \mid \mathcal{D}_N).$$

MARGINAL CLASS-POSTERIOR PROBABILITIES. Recall that the likelihood of our model is

$$p(z \mid \mathbf{x}, \boldsymbol{\theta}) \triangleq \text{Bern}(z \mid \pi_{\boldsymbol{\theta}}(\mathbf{x})),$$

or more succinctly $\pi_{\boldsymbol{\theta}}(\mathbf{x}) = p(z = 1 \mid \mathbf{x}, \boldsymbol{\theta})$. We specify a prior $p(\boldsymbol{\theta})$ on hyperparameters $\boldsymbol{\theta}$ and marginalise out its uncertainty to produce our analog to the marginal acquisition function

$$\Pi(\mathbf{x}; \mathcal{D}_N) = \int \pi_{\boldsymbol{\theta}}(\mathbf{x}) p(\boldsymbol{\theta} \mid \mathcal{D}_N) \, \mathrm{d}\boldsymbol{\theta},$$

where

$$p(\boldsymbol{\theta} \mid \mathcal{D}_N) = \frac{p(\mathbf{z} \mid \mathbf{X}, \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathbf{z} \mid \mathbf{X})}.$$

As in the generic case, we are ultimately interested in maximizing the marginal class-posterior probabilities $\Pi(\mathbf{x}; \mathcal{D}_N)$. However, much like $A(\mathbf{x}; \mathcal{D}_N, \tau)$, the marginal $\Pi(\mathbf{x}; \mathcal{D}_N)$ is analytically intractable in turn due to the intractability of $p(\boldsymbol{\theta} \mid \mathcal{D}_N)$. In this work, we focus on minimizing the log loss of Equation (5.8), which is proportional to the negative log-likelihood

$$\mathcal{L}(\boldsymbol{\theta}) = -\frac{1}{N} \sum_{n=1}^{N} \log p(z_n \mid \mathbf{x}_n, \boldsymbol{\theta}) \propto -\log p(\mathbf{z} \mid \mathbf{X}, \boldsymbol{\theta}).$$

Therefore, we're effectively performing the equivalent of type-II MLE,

$$\hat{\boldsymbol{\theta}}_{\text{MLE}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \log p(\mathbf{z} \mid \mathbf{X}, \boldsymbol{\theta}).$$

In the interest of improving exploration and, of particular interest in our case, calibration of class-membership probabilities, it may be beneficial to consider MC and other approximate inference methods [18, 73, 132]. This remains fertile ground for future work.

5.G.3 *Meta-hyperparameters*

In the case of BORE-MLP, the meta-hyperparameters might consist of, e. g., layer depth, widths, activations, etc. – the tuning of which is often the reason one appeals to BO in the first place. For improvements in calibration, and therefore sample diversity, it may be beneficial to marginalise out the uncertainty about these, or considering some approximation thereof, such as hyper-deep ensembles [274].