# Mixed-Variable Black-Box Optimisation Using Value Proposal Trees

**Yan Zuo**[1*], **Vu Nguyen**[1], **Amir Dezfouli**[2], **David Alexander**[2], **Benjamin Ward Muir**[2], **Iadine Chadès**[2]

[1] Amazon
[2] CSIRO
{ yanzuo@amazon.com, vutngn@amazon.com,
amir.dezfouli@data61.csiro.au, david.alexander@data61.csiro.au, ben.muir@csiro.au, iadine.chades@csiro.au }

## Abstract

Many real-world optimisation problems are defined over both categorical and continuous variables, yet efficient optimisation methods such as Bayesian Optimisation (BO) are ill-equipped to handle such mixed-variable search spaces. The optimisation breadth introduced by categorical variables in the mixed-input setting has seen recent approaches operating on local trust regions, but these methods can be greedy in suboptimal regions of the search space. In this paper, we adopt a holistic view and aim to consolidate optimisation of the categorical and continuous sub-spaces under a single acquisition metric. We develop a tree-based method which retains a global view of the optimisation spaces by identifying regions in the search space with high potential candidates which we call value proposals. Our method uses these proposals to make selections on both the categorical and continuous components of the input. We show that this approach significantly outperforms existing mixed-variable optimisation approaches across several mixed-variable black-box optimisation tasks.

## Introduction

Bayesian optimisation (BO) has established itself as an efficient method for optimising black-box functions that are costly to evaluate (Jones, Schonlau, and Welch 1998; Shahriari et al. 2015; Sazanovich et al. 2021; Cowen-Rivers et al. 2022). Typical BO methods model the black-box function of interest using a surrogate statistical model (usually a Gaussian Process (GP) (Dudley 2010)), seeking out the next point to evaluate by optimising a more tractable (typically differentiable) function called an acquisition function. The role of the acquisition function is to balance two conflicting requirements: exploitation of the current knowledge about the objective function and exploration to gain more knowledge about the objective function. BO has been applied effectively to tasks that range from experimental design (Griffiths and Hernández-Lobato 2017; Li et al. 2018; Shields et al. 2021; Deane et al. 2022) to hyperparameter search (Snoek, Larochelle, and Adams 2012; Gardner et al. 2014; Nguyen et al. 2020b) in machine learning models. Notably, it is often observed that BO is particularly well-suited for applications where the number of allowable evaluations on the objective function is limited (Bull 2011).

[*]Work done while at CSIRO



| Algorithms | Trust region | Categories $h$ selection | Continuous $x \mid h$ across categories |
|---|---|---|---|
| EXP3BO (Gopakumar et al. 2018) | ✓global | ✗independent | ✗non-shared |
| BanditBO (Nguyen et al. 2020a) | ✓global | ✗independent | ✗non-shared |
| CoCaBO (Ru et al. 2020) | ✓global | ✗independent | ✓shared |
| CASM... (Wan et al. 2021) | ✗local | ✓jointly | ✓shared |
| **VPT** (Ours) | ✓global | ✓jointly | ✓shared |

Table 1: Comparison with the mixed categorical-continuous BO approaches in terms of properties and relative trade-offs.

However, many real world optimisation problems involve a mixture of continuous and categorical variables in the input space. For example, in automated machine learning applications (Hutter, Kotthoff, and Vanschoren 2019; Parker-Holder et al. 2022) where the aim is to automatically select a machine learning model along with its corresponding optimal hyperparameters, each model can be seen as a categorical choice while the hyperparameters of the model can be viewed as category-specific continuous variables. Another example is in the chemical reaction space, where often the function we are interested in optimising is represented by both categorical (compositional) variables and continuous (process) variables (Zhou, Li, and Zare 2017). These scenarios pose challenges for current BO models (particularly those using GPs as their underlying surrogate models) which lack the ability to deal effectively with such problems containing multi-layered and complicated search spaces.

The mixed-input setting presents several difficulties; the most apparent of which is that the assumption the acquisition function is differentiable over the input space no longer holds. This has been addressed recently in various ways ranging from one-hot encoding of the categorical components (Snoek, Larochelle, and Adams 2012; Golovin et al. 2017), hierarchical models (Hutter, Hoos, and Leyton-Brown 2011; Bergstra et al. 2011) and most recently, hybrid approaches utilising Multi-Armed Bandits (MABs) (Auer, Cesa-Bianchi, and Fischer 2002) and BO for handling the respective categorical and continuous subspaces. More importantly, the complexity of the optimisation space in mixed-input functions increases dramatically due to the introduction of categorical variables (Ru et al. 2020; Wan et al. 2021). In the BO setting, this is further compounded by a limited budget for evaluating the objective. As such, recent approaches (Eriksson et al. 2019; Wan et al. 2021) have employed local trust regions to reduce the size of the search space for making optimisation more tractable.

However, this approach can result in the model converging to suboptimal solutions in local regions of the search space during optimisation.

In this paper, we present a new approach for optimising black-box functions with multiple continuous and categorical inputs. For carrying out a decision on the categorical part of the input, our method uses a tree-based approach to identify high potential candidates sampled from the surrogate model, which we call *value proposals*. This enables a global, unified approach for optimising on the discrete and continuous sub-spaces of the input, where the decision-making process for both categorical and continuous variables is based on a common metric obtained from the underlying surrogate. We show that this approach significantly improves over existing baselines when applied to a variety of mixed input synthetic and real-world problems.

## Background

Generally, we can organise the literature related to our work into three categories:

**One-hot Encoding** was used prior to the introduction of hierarchical and categorical-specific models for mixed input tasks. These methods (Snoek, Larochelle, and Adams 2012; González et al. 2016; Golovin et al. 2017) transform categorical variables into a one-hot encoded representation, which were used by Bayesian Optimisation frameworks to deal with inputs of mixed nature. In this scenario, the categorical variable with $N$ choices is transformed into a vector of length $N$ with a single non-zero element. Since categories are mutually exclusive, this type of approach treats each extra variable as continuous in $[0, 1]$ using a standard Bayesian Optimisation algorithm for optimisation. However, this type of approach places an equal measure of covariance between all category pairs (despite some or all pairs having different or no correlations), resulting in an acquisition function that is difficult to optimise with large areas of flatness (Rana et al. 2017). To address this issue, (Garrido-Merchán and Hernández-Lobato 2020) restricted the objective function to change only at designated points of 0 and 1, using a kernel function which computed covariances after rounding off the input. However, with this approach, the resulting acquisition function becomes step-wise, making it difficult to optimise.

**Hierarchical** methods were another approach for dealing with mixed inputs. Here, alternative surrogate models which can more naturally handle both continuous and categorical variables were used instead of a Gaussian process. Sequential Model-based Algorithm Configuration (SMAC) (Hutter, Hoos, and Leyton-Brown 2011) uses Random Forests (RFs) (Breiman 2001) as a surrogate model to handle both categorical and continuous components in the input. However, the random nature of RFs (through a reliance on bootstrapping samples and randomly choosing subsets of variables to be tested at each node) weakens the reliability of the derived acquisition function. Adding to this, RFs have a tendency to overfit to training data, requiring careful selection of the number of trees in the model to avoid overfitting. Other tree-based approaches include the Tree-Parzen Estimator (Bergstra et al. 2011), which uses tree-structured Parzen density estimators.

**Category-specific** approaches handle each component of the input separately. (Gopakumar et al. 2018) developed EXP3BO, an approach to deal with mixed categorical and continuous input spaces by utilising a Multi-Armed Bandit to make categorical choices and training a separate surrogate model specific for each choice of category. As a result, the observed data is divided into smaller subsets (one for each category), resulting in a sample-inefficient optimisation procedure that cannot handle problems with a large number of categorical choices. (Nguyen et al. 2020a) introduced a batched setting to the optimisation framework of (Gopakumar et al. 2018) and replaced the respective EXP3 and Upper Confidence Bound (UCB) algorithms of the framework with Thompson Sampling (Chapelle and Li 2011). A key limitation of these frameworks is that they only allow for optimisation of a single categorical variable; the work of (Ru et al. 2020) extended this type of approach by allocating a MAB per categorical variable, enabling the optimisation of functions with multiple categorical variables. However, each MAB is individually updated using the EXP3 algorithm and this may lead to estimates which are disjoint from the BO back-end. CASMOPOLITAN (Wan et al. 2021) utilised local trust regions in both the categorical and continuous parts of the input to reduce the overall size of the search space for optimisation.

## Preliminaries

### Problem Setup

We consider the problem of optimising a black-box function $f(\boldsymbol{z})$ where the input $\boldsymbol{z}$ is comprised of categorical and continuous parts *i.e.* $\boldsymbol{z} = [\boldsymbol{h}, \boldsymbol{x}]$. Here, $\boldsymbol{h} = [h_1, ..., h_k]$ is a vector of categorical variables from the discrete topological space $\mathcal{C}$, with each categorical variable $h_i \in \{1, 2, ..., N_j\}$ taking one of $N_j$ different values. The continuous component of the input, $\boldsymbol{x}$, is drawn from a $d_x$-dimensional hypercube $\mathcal{X}$. Formally, the optimisation of the black-box function $f$ is expressed as:

$$\boldsymbol{z}^* = [\boldsymbol{h}^*, \boldsymbol{x}^*] = \underset{\boldsymbol{z}}{\operatorname{argmax}}\, f(\boldsymbol{z}) \tag{1}$$

which is performed sequentially by making a series of evaluations on $\boldsymbol{z}_1, ...\boldsymbol{z}_T$. The goal is to find the best configuration $\boldsymbol{z}^*$ that maximises $y$, which is the value returned from our objective function $f$. For convenience, we use $c$ to denote a possible combination of categorical choices out of $C = \prod_{j=1}^{k} N_j$ available combinations, such that $c \in \{1, ..., C\}$.

### Bayesian Optimisation

Given a black-box objective function $f : \mathcal{X} \to \mathbb{R}$, the goal of BO is to find the optimal value $\boldsymbol{x}^*$ under a setting with limited evaluations on $f$ (Snoek, Larochelle, and Adams 2012; Nguyen and Osborne 2020). This optimal value maximises the objective $f$, and is defined as $\boldsymbol{x}^* = \operatorname{argmax}_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x})$. The process of BO involves using a surrogate to model the objective $f$; typically, $f$ is assumed to be a smooth function and commonly a Gaussian Process (GP) (Rasmussen 2003) is used for the surrogate. The GP models an underlying probability distribution over functions $f$ and is represented by

mean and covariance functions (or kernel) $\mu(\boldsymbol{x})$ and $\kappa(\boldsymbol{x}, \boldsymbol{x}')$ respectively, where $f(\boldsymbol{x}) \sim \mathtt{GP}(\mu(\boldsymbol{x}), \kappa(\boldsymbol{x}, \boldsymbol{x}'))$. In this work, our choice of kernel $\kappa$ is the mixed-kernel (Ru et al. 2020; Parker-Holder et al. 2021; Wan et al. 2021; Zuo et al. 2022) which is suited towards modelling mixed-input type problems (for additional details on the mixed-kernel, please refer to the supplementary material).

The surrogate encodes our prior beliefs about the objective $f$ and we can build a posterior through further observations when we evaluate $f(\boldsymbol{x})$. Using this posterior, at a given optimisation iteration $t$, an acquisition function $\alpha_t(\boldsymbol{x})$ can be built, which can then be optimised to identify the next point to be sampled such that $\boldsymbol{x}_t = \text{argmax}_{\boldsymbol{x} \in \mathcal{X}}\, \alpha_t(\boldsymbol{x})$. Since $\alpha_t(\boldsymbol{x})$ is derived from our surrogate model, it is comparatively cheaper to compute and can be optimised using standard optimisation techniques.

## Decision Trees & Forests

**Decision Trees** consist of a set internal nodes and terminal nodes. The internal nodes (referred to as decision nodes) dictate the routing path of data through the tree, via split functions which drive data samples to left or right child decision nodes, whilst terminal leaf nodes contain values which contribute to the prediction made by the tree (Safavian and Landgrebe 1991). The set of decision nodes is defined as $D = \{d_0, \cdots, d_{N-1}\}$, where each node holds a decision function $d(\boldsymbol{x}; \theta)$ parameterised by $\theta$. A terminal leaf node is defined as $\ell = \delta(\boldsymbol{x}; \Theta)$ where $\delta$ represents the routing function which directs data $\boldsymbol{x}$ to the terminal leaf node $\ell$, given the parameters of the tree $\Theta$.

The stored value in leaf node $\ell$ is defined as $q = Q(\ell)$, where $Q$ is a mapping of $\ell$ to its stored value (for more details, please see the supplementary material).

**Decision Forests** are the ensemble setting of decision trees. Here, the ensemble of $\mathcal{F}$ trees is combined to give a single output, usually by averaging the predictions delivered by each tree (Ho 1995). Hence, the prediction made by the ensemble for the input $\boldsymbol{x}$ is given as:

$$V = \frac{1}{\mathcal{F}} \sum_{j=1}^{\mathcal{F}} Q(\ell) = \frac{1}{\mathcal{F}} \sum_{j=1}^{\mathcal{F}} Q^j(\delta^j(\boldsymbol{x}; \Theta^j)) \tag{2}$$

where $Q^j$, $\delta^j$ and $\Theta^j$ are the respective leaf node mapping function, routing function and parameters of tree $j$ in the ensemble.

## Local Trust Regions in BO

Under the BO setting, local trust regions (TRs) have been effectively applied to help with scaling up the optimisation process of higher dimensional problems. The key idea behind this approach is to break down the function landscape into several smaller local regions (called trust regions) which should be easier to optimise when compared to the entire function landscape. TuRBO (Eriksson et al. 2019) first established TRs in the continuous input setting, as hyper-rectangles centred around the best solution found so far during optimisation ($\boldsymbol{x}^*$). These hyper-rectangles are initialised to a base side length $L$, and re-scaled on each of the $d$ dimensions by the GP model's corresponding lengthscale parameter $l_i$, where the re-scaled side length on the $i^{th}$ dimension is given as:

$$L_i = \frac{l_i L}{(\prod_{j=1}^{d} l_j)^{1/d}}. \tag{3}$$

To maintain these TRs, (Eriksson et al. 2019) adopts the simplex approach of (Nelder and Mead 1965) using a shrinking and expanding heuristic (for additional details, please refer to the supplementary material). CASMOPOLITAN (Wan et al. 2021) adapted the approach of (Eriksson et al. 2019) for the categorical and mixed-input setting, using a Hamming distance (Hamming 1950) based TR to partition the categorical component of the input space. Here the TR is defined in terms of a radius $L^h$ from the best observed location $\boldsymbol{h}^*$:

$$\text{TR}_h(\boldsymbol{h}^*)_{L^h} = \left\{ \boldsymbol{h} | \sum_{i=1}^{d_h} \delta(h_i, h_i^*) \leq L^h \right\} \tag{4}$$

where the same shrinking and expanding heuristic from (Nelder and Mead 1965) is used to maintain the size of the TR.

# Method

There are a few issues with the local TR-based approach. The first is sensitivity to the size of the local TR: maintaining the correct side length $L$ is critical as too large of a TR would mean losing local accuracy and too small would mean the TR may not contain good solutions. The second is in maintaining a correct TR size during optimisation; this is an additional consideration that must be made on top of balancing between exploration and exploitation. Finally, a local TR-based model needs to be able to identify if it has converged to a suboptimal solution and make a decision on when to employ its restart strategy (Eriksson et al. 2019; Wan et al. 2021), which is yet another heuristic that needs to be tuned for the problem at hand.

Our proposed method addresses each of these aforementioned issues; instead of restricting the input space to *local* trust regions, we can instead utilise an auxiliary model alongside the surrogate model to identify *non-local* sub-regions of the input search space earlier in the optimisation pipeline. The auxiliary model maintains a global view of the input search space and effectively establishes trust regions that are not necessarily restricted to their locality (see Table 1). For the auxiliary model that is used to identify non-local trust regions, we utilise a tree-based approach which have been shown to perform well across both continuous (Bartz-Beielstein and Markon 2004; Zuo, Avraham, and Drummond 2018, 2021) and categorical data (Breiman 2001; Zuo and Drummond 2017) and have an innate ability to handle mixed-input type data efficiently (Breiman 2001; Chen et al. 2015; Zuo and Drummond 2020), making them particularly suited for optimising mixed-input objectives under the BO setting (Hutter, Hoos, and Leyton-Brown 2011).

## Value Proposal Trees (VPT)

The VPT process is detailed in Algorithm 1 and can be summarised by three main components: (i) a candidate generation

process queries the statistical surrogate model to obtain a set of potential input configurations to evaluate; (ii) this set of input configurations and their corresponding acquisition values are fit to a tree-based regression model and clustered by a similarity measure; (iii) a top cluster is chosen as the value proposal set - a set of candidates that share similarities and possess categorical variable configurations which are predicted to have high potential in maximising the objective. The BO back-end is used to optimise the continuous variables for the candidates within this top cluster given their respective categorical variable configuration and the candidate with the highest acquisition value from within the value proposal set is selected as the next configuration for evaluating the black-box objective function.

Algorithm 1: Value Proposal Trees

**Input:** Black-box function $f$, Initial data $\mathcal{D}_0$, Max iterations $T$, Number of possible categorical combinations $C$
**Output:** The best recommendation $\boldsymbol{z}_T^* = [\boldsymbol{h}_T^*, \boldsymbol{x}_T^*]$
**for** $t = 1, ..., T$ **do**
  Fit a GP model using $\mathcal{D}_{t-1}$ Generate $N$ candidates $\mathcal{T}_t$ from a GP surrogate Train a tree-based regression model using $\mathcal{T}_t$ Generate proximity matrix $\mathcal{P}_t$ Use $\mathcal{P}_t$ to cluster $\mathcal{T}_t$ to generate the value proposal set $\mathcal{V}_t$
  **for** $c \in \{1, ..., \mathcal{C}_t\}$ **do**
    $\boldsymbol{x}_{t,c}^* = \underset{\boldsymbol{x}}{\operatorname{argmax}}\, \alpha_{t,c}(\boldsymbol{x}|\mathcal{D}_{t-1}, \boldsymbol{h}_{t,c})$
    $v_{t,c} = \alpha_{t,c}(\boldsymbol{x}_{t,c}^*|\mathcal{D}_{t-1}, \boldsymbol{h}_{t,c})$
    $\mathcal{V}_t[c] = [\boldsymbol{x}_{t,c}^*, v_{t,c}]$
  $c^*$ = $\underset{c}{\operatorname{argmax}}\, \mathcal{V}_t$
  Set $\boldsymbol{z}_t^* = [\boldsymbol{h}_t^*, \boldsymbol{x}_t^*]$ = $[\boldsymbol{h}_{t,c^*}, \boldsymbol{x}_{t,c^*}^*]$ and obtain $f_t^* = f(\boldsymbol{z}_t^*)$
  Augment the data: $\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \bigcup (\boldsymbol{z}_t^*, f_t^*)$

**Sampling the Posterior** occurs at each optimisation $t$ iteration where VPT first queries the surrogate model to produce a set of $N$ candidates $\mathcal{T}_t = \{(\boldsymbol{z}_{t,0}, \alpha_{t,0}), (\boldsymbol{z}_{t,1}, \alpha_{t,1}), (\boldsymbol{z}_{t,2}, \alpha_{t,2}), ..., (\boldsymbol{z}_{t,N}, \alpha_{t,N})\}$. Each sample $i$ in the set $\mathcal{T}_t$ is a pair consisting of a candidate input configuration $\boldsymbol{z}_{t,i}$ and its corresponding acquisition value $\alpha_{t,i}$. For convenience and the sake of brevity, we drop the $t$ subscript notation from $\boldsymbol{z}_{t,i}$ and its acquisition value $\alpha_{t,i}$ when referring to individual samples from the candidate set $\mathcal{T}_t$ from this point onward. Before optimisation begins, the objective function $f$ is first evaluated a number of times using input configurations which are uniformly sampled from the input search space. The results of these evaluations are added to the initial observation set $\mathcal{D}_0$ and a posterior is fitted to $\mathcal{D}_0$ by maximising the log marginal likelihood. From the initial observation set $\mathcal{D}_0$, we obtain our initial incumbent $\mathcal{E}_0$ by selecting the observation with maximises the objective function $f$ (*i.e.* $\mathcal{E}_0 = \operatorname{argmax}_{\boldsymbol{z}} f(\boldsymbol{z})$). We can then sample the required $N$ candidates from the fitted surrogate model.

For sampling, we employ a trust region based technique similar to (Wan et al. 2021). Here, we develop a general strategy for sampling the categorical component of the input $\boldsymbol{h}_t$ using the current incumbent $\mathcal{E}_{t-1}$ for optimisation iteration $t$. Selecting candidates at optimisation iteration $t$ on the categorical choices of the input involves searching in a local trust region around the current incumbent $\mathcal{E}_{t-1}$ via making perturbations to the categorical component of the input. Motivated by (Wan et al. 2021), we use Hamming distance (Hamming 1950) to determine the range of permissible perturbations to the categorical variables from the current incumbent.

**Regression & Clustering with Trees** is performed on the set of candidates $\mathcal{T}_t$ for fitting a tree-based regression model. For the $i^{th}$ sample in $\mathcal{T}_t$, we use the candidate input configuration $\boldsymbol{z}_i$ as the input to the regression model and its corresponding acquisition value $\alpha_i$ as the regression target. Here, the regression tree model is fit on $\mathcal{T}_t$ via induction (Quinlan 1986) and minimises mean squared loss between the prediction of the model and the target acquisition value:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\alpha_i - y_i)^2 \tag{5}$$

where $y_i$ denotes the predicted acquisition value for sample $i$ in the set $\mathcal{T}_t$. The fitted regression model is used to generate a proximity matrix $\mathcal{P}_t$, a square matrix which is used to measure the similarity between samples in $\mathcal{T}_t$. For each sample in the set $\mathcal{T}_t$, we compute its pairwise distance to other samples in the set using a count of common leaf nodes the pairwise samples share (normalised across the $\mathcal{F}$ trees in the ensemble). That is, for sample $i$, we compute its proximity $p$ to sample $j$ as:

$$p_{i,j} = \frac{1}{\mathcal{F}}\sum_{k=1}^{\mathcal{F}}\mathbb{I}_{\ell_i=\ell_j}. \tag{6}$$

Here, $\mathbb{I}_{\mathbb{C}}$ is an indicator function which equals 1 when its condition $\mathbb{C}$ is met and 0 otherwise. The proximity matrix $\mathcal{P}_t$ is thus composed of normalised values in the range of $[0, 1]$. Using the proximity matrix $\mathcal{P}_t$, samples from $\mathcal{T}_t$ can be clustered according to their similarity, generating the set of clusters $\{\mathcal{K}_0, ..., \mathcal{K}_K\}$. Each cluster is scored using its acquisition value $\alpha_k$, by averaging over the individual acquisition values corresponding to samples from within the given cluster $k$:

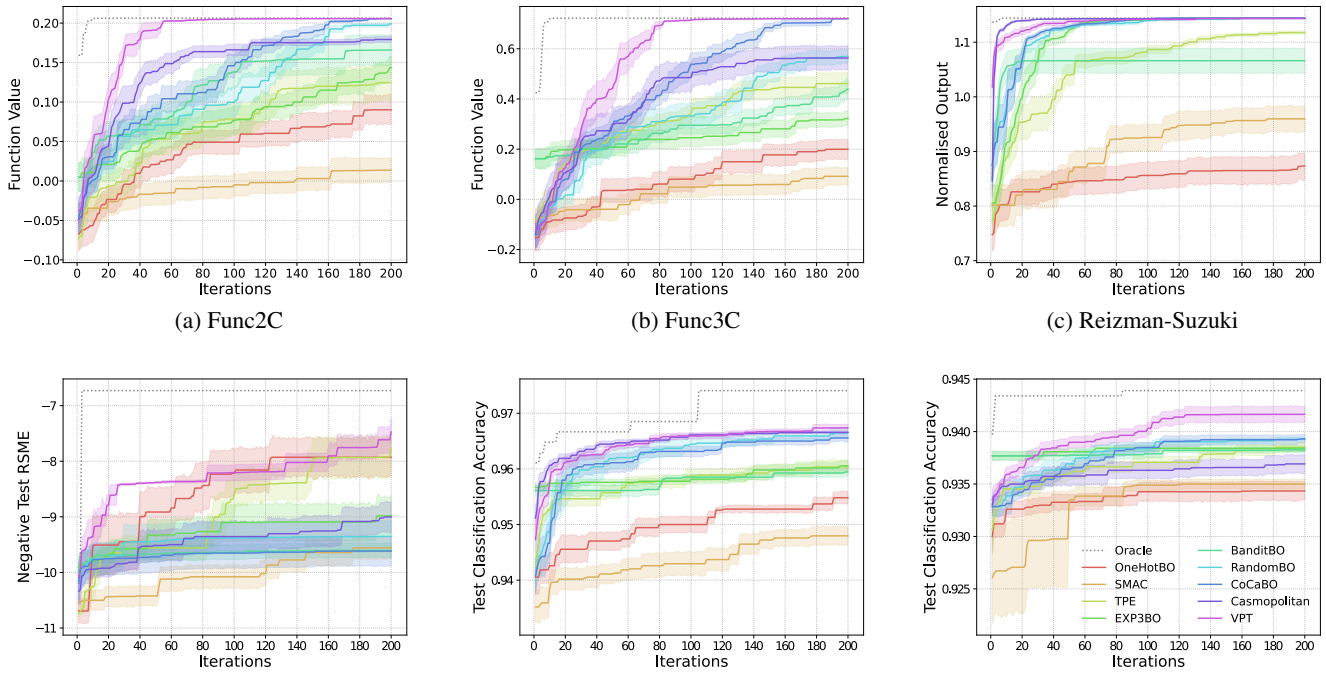
$$\alpha_k = \frac{1}{N_k}\sum_{i=1}^{N_k}\alpha_{k,i} \tag{7}$$

where $N_k$ is the number of samples in cluster $k$ and the $\alpha_{k,i}$ is the acquisition value corresponding to the $i^{th}$ sample in cluster $k$.

**Creating Value Proposals** occurs via selecting the cluster $k^*$ with the largest acquisition value, which creates the value proposal set $\mathcal{V}_t$:

$$k^* = \underset{k\in\{1,...,K\}}{\operatorname{argmax}}\ \alpha_k. \tag{8}$$

Following this, the categorical part of the input $\boldsymbol{h}_t$ is fixed and the BO back-end is then used to maximise the acquisition function and optimise the continuous part of the input $\boldsymbol{x}_t$. This step is done in parallel for each candidate $\boldsymbol{z}_c$ in cluster $k^*$, where we find the corresponding optimal value for $\boldsymbol{x}$ given the selected categorical values which is denoted as $\boldsymbol{x}_{t,c}^*$. Following optimisation of the continuous part of the




(a) Func2C (b) Func3C (c) Reizman-Suzuki

(d) SVM-Boston (e) XG-MNIST (f) NASBench-101

Figure 1: Performance of VPT against existing methods on various synthetic and real-world tasks. The shaded regions represent the standard error from the mean value (solid line) over random trials.

input $\boldsymbol{x}_t$ for each sample $\mathcal{V}_t$, we obtain a cluster of candidates which represents high potential samples to evaluate on the objective function $f$. We refer to these candidates as *value proposals*, and this is collected in the set of value proposals $\mathcal{V}_t$ (containing $\mathcal{C}_t$ proposals) for the optimisation iteration $t$.

The value proposal with the maximum acquisition value within the value proposal set $\mathcal{V}_t$ is selected and its corresponding input configuration $[\boldsymbol{h}_t^*, \boldsymbol{x}_t^*]$ is chosen as the next query point of the objective function $f$. Following this, the newly observed function value $f_t^*$ and corresponding input $\boldsymbol{z}_t^*$ is added to the observation set $\mathcal{D}_t$.

## Experiments

We compared VPT against several competing baselines which can handle mixed-variable inputs: SMAC (Hutter, Hoos, and Leyton-Brown 2011), TPE (Bergstra et al. 2011), GP-based BO with one-hot encoding (One-Hot BO), EXP3BO (Gopakumar et al. 2018), Bandit-BO (Nguyen et al. 2020a), CoCaBO (Ru et al. 2020) and CASMOPOLITAN (Wan et al. 2021). All baseline methods are implemented according to their publicly available Python repositories (please see the supplementary material for detailed settings used in these baselines). Additionally, we implemented RandomBO, where the EXP3 agent in CoCaBO is replaced by a random selection agent.

**VPT Settings** We utilise the same mixed-kernel GP surrogate as (Wan et al. 2021), which uses a fixed kernel mixture hyperparameter of $\lambda = 0.5$. We follow the optimisation approach of (Wan et al. 2021), optimising GP hyperparameters by maximising the log marginal likelihood using variational inference (VI) (Ranganath, Gerrish, and Blei 2014), with a learning rate of $\alpha = 0.03$. At each optimisation iteration, we generate $N = 1000$ randomly selected candidates, sampling from a local trust region of the current incumbent. For the tree-based regression model, we use the Extremely Randomised Trees (ERT) approach (Geurts, Ernst, and Wehenkel 2006), constructing an ensemble of 100 trees, each with a maximum depth equal to the number of categorical variables in the black-box function of interest. Clustering is performed using DBSCAN (Ester et al. 1996). We use the proximity matrix generated from the ERT model to specify $\epsilon$, where we use the $20^{th}$ percentile of distances for the value of $\epsilon$. We set the minimum number of points clustered around a region for it to be considered dense as 5.

**Datasets** For all benchmarks, the continuous inputs were normalised to $\boldsymbol{x} \in [0, 1]^{d_x}$ and all experiments were conducted on an 8-core 3.4Ghz Intel Xeon processor with 64GB RAM. Our benchmarks include a variety of synthetic and real, single and multi-objective problems (for additional details, please see the supplementary material).

### Performance of VPT

We evaluated the optimisation performance of our proposed VPT method against existing methods. Following (Ru et al. 2020), we set each optimisation trial to consist of $T = 200$

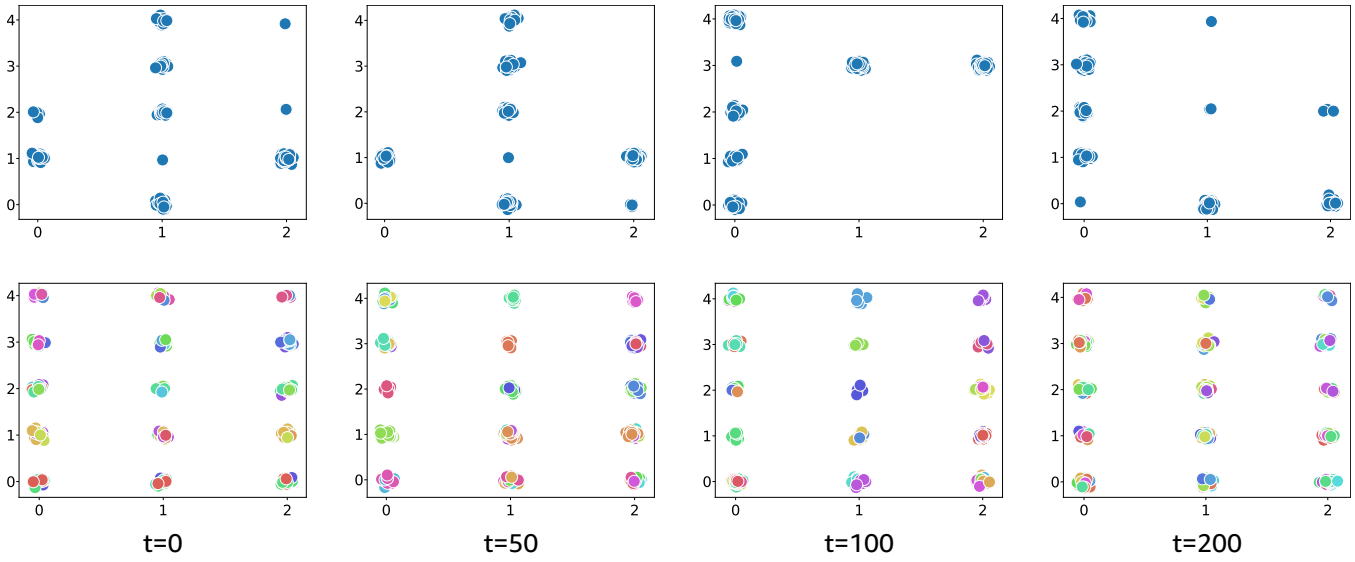



Figure 2: Clustering of categorical candidates generated by CASMOPOLITAN (Wan et al. 2021) (top) and VPT (bottom) of the Func2C dataset. For each method, $N = 1000$ candidates were generated at each optimisation iteration and the optimal categorical configuration is $[1, 1]$. The x-axis and y-axis show the different possible choices for each categorical variable ($k = 2$), with a small amount of noise added to help with visualisation. For VPT, the different colours represent different clusters as indicated by the regression tree model. When compared to the local trust region approach of (Wan et al. 2021), our method establishes trust regions that *non-local* and span across multiple categorical configurations. In contrast, CASMOPOLITAN is restricted in selecting specific categorical configurations which may not be optimal.

iterations. We performed 20 random trials for the Func2C, Func3C and Reizman-Suzuki datasets. For the SVM-Boston, XG-MNIST and NASBench-101 datasets, we performed 10 random trials. Means and standard errors over all trials are presented.

The gray dotted line in Fig. 1 represents an Oracle agent that we trained (similar to (Gopakumar et al. 2018)). Here, we run BO for each of the $C$ choices of possible categorical combinations, with each choice allocated its own separate GP surrogate model $\text{GP}_c$. We allocate each of the $C$ choices the full $T = 200$ iterations as well as all 24 initial sample points and optimise the hyperparameters for $\text{GP}_c$ every iteration by maximising the log marginal likelihood. At each iteration $t$, the Oracle's performance is then taken as the best value for $f$ over all possible categorical choices $C$. Hence, we use the Oracle to represent a best possible outcome scenario at each iteration $t$.

Across all synthetic and real-world problems, our VPT method outperforms other competing approaches, where the general trend is that VPT demonstrates a significant improvement in initial performance and more quickly converges towards the performance of the Oracle agent. Across the competing baselines, we observe that there is large variance in performance between different problems, but the performance of our VPT approach remains consistent across different datasets. This is encouraging to see and highlights the beneficial properties of our method over competing mixed-input approaches (highlighted in Table 1).

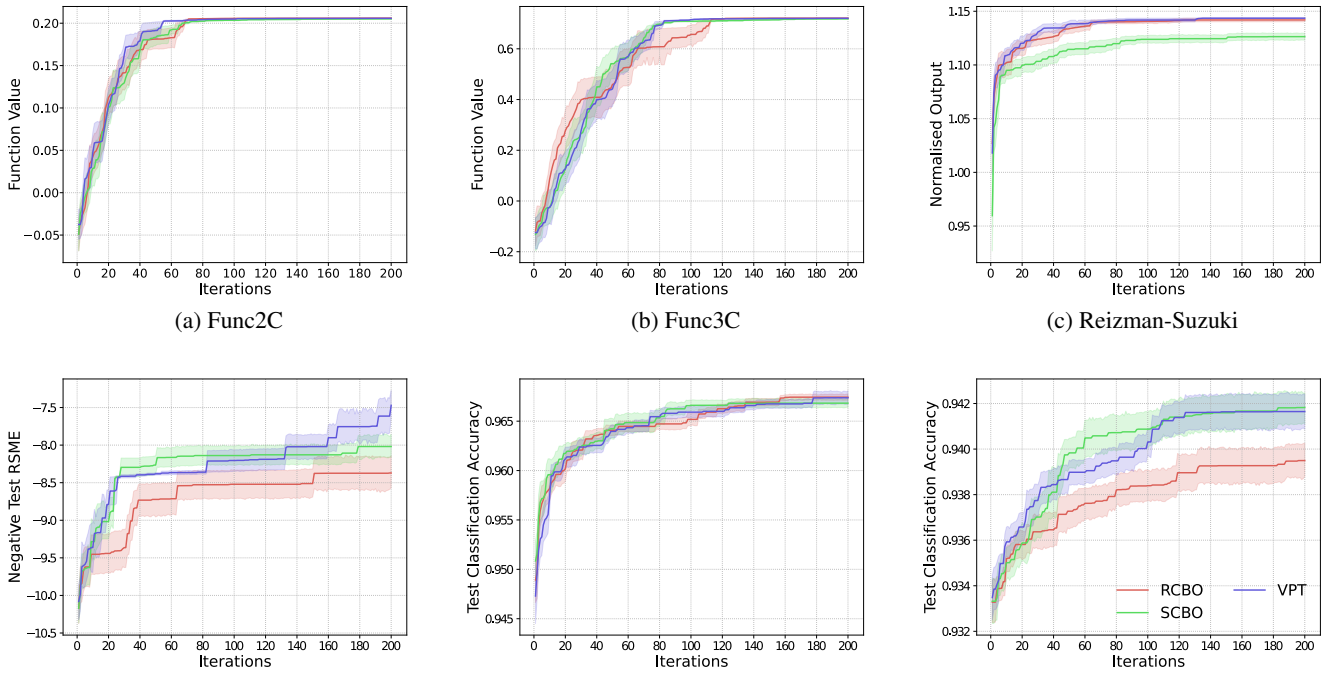
Note that due to the nature of the EXP3BO and BanditBO methods maintaining separate surrogate models for each of the possible $C$ categories and being allocated 3 initial samples per surrogate, for problems where $C$ is large (*e.g.* Func3C, XG-MNIST and NASBench-101), their initial performance at $t$ =1 is significantly higher due to the larger number of initial samples observed from $f$ (*e.g.* 180 samples for Func3C vs. 24 for all other methods including VPT).

## Local vs. Non-local Trust Regions

Here, we further show the trade-offs associated with local TR-based approaches such as CASMOPOLITAN (Wan et al. 2021) as indicated in Table 1, when compared to our VPT method. In Fig. 2 we show the visualised clustering of candidates generated by (Wan et al. 2021) and our VPT method at $t = 0$, $t = 50$, $t = 100$ and $t = 200$ optimisation iterations for the Func2C synthetic problem. We plot the different possible choices for each of the $k = 2$ variables on the respective x-axis and y-axis. Each point represents a candidate generated by (Wan et al. 2021) and our VPT method at the specified optimisation iteration. At each of the shown optimisation iterations, CASMOPOLITAN appears to concentrate its categorical configurations around a few selections, and largely misses the optimal categorical configuration of $[1, 1]$. In contrast, VPT maintains a diverse set of candidates which covers all possible categorical configurations and the optimal categorical configuration contains candidates from several different clusters (represented by different colours).

## VPT Ablation Study

We perform an ablation study comparing our VPT method with its possible variations including Random Candidate




(a) Func2C (b) Func3C (c) Reizman-Suzuki

(d) SVM-Boston (e) XG-MNIST (f) NASBench-101

Figure 3: Ablation study between our proposed VPT method and its variations. The shaded regions represent the standard error from the mean value (solid line) over random trials.

Bayesian Optimisation (RCBO) and Selected Candidate Bayesian Optimisation (SCBO). RCBO omits the tree fitting and clustering step in our VPT approach and instead just optimises the continuous component of the input for the full set of generated candidates using the previously described trust region sampling method, selecting the candidate with the largest acquisition value. SCBO generates a candidate for each possible categorical combination and selects the arm which maximises the acquisition value after optimising for the continuous component of the input. All 3 methods share the same BO back-end of (Wan et al. 2021) which uses a mixed-kernel surrogate GP model where optimisation is performed using variational inference. The results of this ablation study across the 6 datasets are shown in Fig. 3.

For RCBO, we observe that the exclusion of tree-fitting and clustering from the VPT approach results in a reduction in performance over optimisation trials, particularly on the SVM-Boston and NASBench-101 datasets. This indicates that using the tree-based regression model and clustering does indeed help in identifying high potency candidates for further optimisation.

For SCBO, there are two main issues; the first is that making a selection through comprehensively considering all possible categorical variable configurations can result in SCBO getting stuck locally and repeatedly selecting the same sub-optimal arm. The second issue is that as the number of categorical variables increase, the possible configuration space exponetially increases and SCBO subsequently incurs exponential computation costs as shown in Table 2 for problems with a large number of possible categorical configurations (XG-MNIST and NASBench-101).

| | Method | | |
|---|---|---|---|
| | RCBO | SCBO | VPT |
| Func2C | 7.1587 | **2.3063** | 2.5666 |
| Func3C | 8.6316 | 2.5372 | **2.5346** |
| Reizman-Suzuki | 4.4041 | **2.2138** | 2.3854 |
| SVM-Boston | 8.9047 | 2.4609 | **2.5397** |
| XG-MNIST | 13.0525 | 5.2359 | **4.7051** |
| NASBench-101 | 11.7123 | 4.2158 | **4.1572** |
| Average | 8.7359 | 3.1616 | **3.1482** |

Table 2: The mean wall-clock time (in seconds) overheads for each BO iteration across 200 optimisation rounds.

## Conclusion

In this paper, we presented a mixed-variable black-box optimisation approach which adopts a global BO approach in the joint optimisation of categorical and continuous variables. Our approach adopts a tree-based method to establish *non-local* trust regions for identifying high potential candidates to be optimised in a BO setting. Empirically, we show that VPT gives significant improvements in performance over existing mixed-variable optimisation approaches on a wide range of problems.

## References


Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2): 235–256.

Bartz-Beielstein, T.; and Markon, S. 2004. Tuning search algorithms for real-world applications: A regression tree based approach. In *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No. 04TH8753)*, volume 1, 1111–1118. IEEE.

Bergstra, J.; Bardenet, R.; Bengio, Y.; and Kégl, B. 2011. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24.

Breiman, L. 2001. Random forests. *Machine learning*, 45(1): 5–32.

Bull, A. D. 2011. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12(10).

Chapelle, O.; and Li, L. 2011. An empirical evaluation of thompson sampling. *Advances in neural information processing systems*, 24: 2249–2257.

Chen, T.; He, T.; Benesty, M.; Khotilovich, V.; Tang, Y.; Cho, H.; et al. 2015. Xgboost: extreme gradient boosting. *R package version 0.4-2*, 1(4): 1–4.

Cowen-Rivers, A. I.; Lyu, W.; Tutunov, R.; Wang, Z.; Grosnit, A.; Griffiths, R. R.; Maraval, A. M.; Jianye, H.; Wang, J.; Peters, J.; et al. 2022. HEBO: Pushing The Limits of Sample-Efficient Hyper-parameter Optimisation. *Journal of Artificial Intelligence Research*, 74: 1269–1349.

Deane, K.; Yang, Y.; Licavoli, J. J.; Nguyen, V.; Rana, S.; Gupta, S.; Venkatesh, S.; and Sanders, P. G. 2022. Utilization of Bayesian Optimization and KWN Modeling for Increased Efficiency of Al-Sc Precipitation Strengthening. *Metals*, 12(6): 975.

Dudley, R. M. 2010. Sample functions of the Gaussian process. *Selected Works of RM Dudley*, 187–224.

Eriksson, D.; Pearce, M.; Gardner, J.; Turner, R. D.; and Poloczek, M. 2019. Scalable global optimization via local bayesian optimization. *Advances in neural information processing systems*, 32.

Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X.; et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, 226–231.

Gardner, J. R.; Kusner, M. J.; Xu, Z. E.; Weinberger, K. Q.; and Cunningham, J. P. 2014. Bayesian Optimization with Inequality Constraints. In *ICML*, volume 2014, 937–945.

Garrido-Merchán, E. C.; and Hernández-Lobato, D. 2020. Dealing with categorical and integer-valued variables in bayesian optimization with gaussian processes. *Neurocomputing*, 380: 20–35.

Geurts, P.; Ernst, D.; and Wehenkel, L. 2006. Extremely randomized trees. *Machine learning*, 63(1): 3–42.

Golovin, D.; Solnik, B.; Moitra, S.; Kochanski, G.; Karro, J.; and Sculley, D. 2017. Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 1487–1495.

González, J.; Dai, Z.; Hennig, P.; and Lawrence, N. 2016. Batch Bayesian optimization via local penalization. In *Artificial intelligence and statistics*, 648–657. PMLR.

Gopakumar, S.; Gupta, S.; Rana, S.; Nguyen, V.; and Venkatesh, S. 2018. Algorithmic assurance: An active approach to algorithmic testing using bayesian optimisation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 5470–5478.

Griffiths, R.-R.; and Hernández-Lobato, J. M. 2017. Constrained bayesian optimization for automatic chemical design. *arXiv preprint arXiv:1709.05501*.

Hamming, R. W. 1950. Error detecting and error correcting codes. *The Bell system technical journal*, 29(2): 147–160.

Ho, T. K. 1995. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, 278–282. IEEE.

Hutter, F.; Hoos, H. H.; and Leyton-Brown, K. 2011. Sequential model-based optimization for general algorithm configuration. In *International conference on learning and intelligent optimization*, 507–523. Springer.

Hutter, F.; Kotthoff, L.; and Vanschoren, J. 2019. *Automated machine learning: methods, systems, challenges*. Springer Nature.

Jones, D. R.; Schonlau, M.; and Welch, W. J. 1998. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4): 455–492.

Li, C.; Santu, R.; Gupta, S.; Nguyen, V.; Venkatesh, S.; Sutti, A.; Rubin, D.; Slezak, T.; Height, M.; Mohammed, M.; et al. 2018. Accelerating Experimental Design by Incorporating Experimenter Hunches. In *2018 IEEE International Conference on Data Mining (ICDM)*, 257–266. IEEE.

Nelder, J. A.; and Mead, R. 1965. A simplex method for function minimization. *The computer journal*, 7(4): 308–313.

Nguyen, D.; Gupta, S.; Rana, S.; Shilton, A.; and Venkatesh, S. 2020a. Bayesian optimization for categorical and category-specific continuous inputs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 5256–5263.

Nguyen, V.; Masrani, V.; Brekelmans, R.; Osborne, M.; and Wood, F. 2020b. Gaussian process bandit optimization of the thermodynamic variational objective. *Advances in Neural Information Processing Systems*, 33: 5764–5775.

Nguyen, V.; and Osborne, M. A. 2020. Knowing the what but not the where in Bayesian optimization. In *International Conference on Machine Learning*, 7317–7326. PMLR.

Parker-Holder, J.; Nguyen, V.; Desai, S.; and Roberts, S. J. 2021. Tuning mixed input hyperparameters on the fly for efficient population based autorl. *Advances in Neural Information Processing Systems*, 34: 15513–15528.

Parker-Holder, J.; Rajan, R.; Song, X.; Biedenkapp, A.; Miao, Y.; Eimer, T.; Zhang, B.; Nguyen, V.; Calandra, R.; Faust, A.; et al. 2022. Automated reinforcement learning (autorl): A survey and open problems. *Journal of Artificial Intelligence Research*, 74: 517–568.

Quinlan, J. R. 1986. Induction of decision trees. *Machine learning*, 1(1): 81–106.

Rana, S.; Li, C.; Gupta, S.; Nguyen, V.; and Venkatesh, S. 2017. High dimensional Bayesian optimization with elastic Gaussian process. In *International conference on machine learning*, 2883–2891. PMLR.

Ranganath, R.; Gerrish, S.; and Blei, D. 2014. Black box variational inference. In *Artificial intelligence and statistics*, 814–822. PMLR.

Rasmussen, C. E. 2003. Gaussian processes in machine learning. In *Summer school on machine learning*, 63–71. Springer.

Ru, B.; Alvi, A.; Nguyen, V.; Osborne, M. A.; and Roberts, S. 2020. Bayesian optimisation over multiple continuous and categorical inputs. In *International Conference on Machine Learning*, 8276–8285. PMLR.

Safavian, S. R.; and Landgrebe, D. 1991. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3): 660–674.

Sazanovich, M.; Nikolskaya, A.; Belousov, Y.; and Shpilman, A. 2021. Solving black-box optimization challenge via learning search space partition for local bayesian optimization. In *NeurIPS 2020 Competition and Demonstration Track*, 77–85. PMLR.

Shahriari, B.; Swersky, K.; Wang, Z.; Adams, R. P.; and De Freitas, N. 2015. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175.

Shields, B. J.; Stevens, J.; Li, J.; Parasram, M.; Damani, F.; Alvarado, J. I. M.; Janey, J. M.; Adams, R. P.; and Doyle, A. G. 2021. Bayesian reaction optimization as a tool for chemical synthesis. *Nature*, 590(7844): 89–96.

Snoek, J.; Larochelle, H.; and Adams, R. P. 2012. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25.

Wan, X.; Nguyen, V.; Ha, H.; Ru, B.; Lu, C.; and Osborne, M. A. 2021. Think Global and Act Local: Bayesian Optimisation over High-Dimensional Categorical and Mixed Search Spaces. *arXiv preprint arXiv:2102.07188*.

Zhou, Z.; Li, X.; and Zare, R. N. 2017. Optimizing chemical reactions with deep reinforcement learning. *ACS central science*, 3(12): 1337–1344.

Zuo, Y.; Avraham, G.; and Drummond, T. 2018. Generative adversarial forests for better conditioned adversarial learning. *arXiv preprint arXiv:1805.05185*.

Zuo, Y.; Avraham, G.; and Drummond, T. 2021. Improved training of generative adversarial networks using decision forests. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 3492–3501.

Zuo, Y.; Dezfouli, A.; Chades, I.; Alexander, D.; and Muir, B. W. 2022. Bayesian Optimisation for Mixed-Variable Inputs using Value Proposals. *arXiv preprint arXiv:2202.04832*.

Zuo, Y.; and Drummond, T. 2017. Fast residual forests: Rapid ensemble learning for semantic segmentation. In *Conference on Robot Learning*, 27–36. PMLR.

Zuo, Y.; and Drummond, T. 2020. Residual Likelihood Forests. *arXiv preprint arXiv:2011.02086*.