Hyperparameter Optimization for Astronomy: Taking the Astronomer Out of the Loop

HPO for ASTCVS

Victor Gueorguiev
Supervisor: Prof. Deshen Moodley
CAIR, University of Cape Town - CSIR Meraka
GRGVIC001@myuct.ac.za

ABSTRACT

The task of phenomenon classification in astronomy provides a novel and challenging setting for the application of state-of-the-art techniques addressing the problem of combined algorithm selection and hyperparameter optimization (CASH) of machine learning algorithms, which find local applications such as at the data-intensive Square Kilometre Array (SKA). This work will use various algorithms for CASH to explore the possibility and efficacy of hyperparameter optimization on improving performance of machine learning techniques for astronomy. Then, with focus on the Galaxy Zoo project, these algorithms will be used to conduct an in-depth comparison of state-of-the-art in hyperparameter optimization (HPO) along with techniques that aim to improve performance on large datasets and expensive function evaluations. Finally, the likelihood for an integration with a cognitive vision system for astronomy will be examined by conducting a brief exploration into different feature extraction and selection methods.

CCS CONCEPTS

• **Computing methodologies** → **Search methodologies**; **Machine learning**; **Supervised learning**; *Kernel methods*; *Learning in probabilistic graphical models*; • **Applied computing** → **Astronomy**;

KEYWORDS

machine learning, bayesian optimization, model selection, hyperparameter optimization, astronomy, galaxy morphology

1 INTRODUCTION

Astronomy is a field that actively accumulates and processes vast amounts of data with structure and information that may only be extracted using computational means [18]. For instance, large sky surveys such as the Sloan Digital Sky Survey (SDSS) [34] produce datasets many terabytes in size with tens of millions of data points [21] that require intense computer pre-processing and analysis [18]. Moreover, South Africa will soon be one of the leading contributors to the field of radio astronomy with the construction of the Square Kilometre Array (SKA)[8], which will produce data on the order of exabytes of data in a single year of operation, at a rate of 2 terabytes per second during operation [8]. Therefore, analysis of this data will require the use of automated techniques such as machine learning in order to reduce data into consumable and useful knowledge, and to perform tasks such as astronomical phenomenon classification [1] [8, 32]. Indeed, astronomy has seen focused applications of many statistical machine learning approaches [18] such as decision trees, k-means clustering, and simple multi-layer perceptrons for galaxy morphology classification [18] and, most recently, deep-learning approaches for image analysis on problems including galaxy lensing identification to assist in dark matter sky surveys [13].

However, while there is a desire for the application of machine learning techniques in astronomy, there is also a lack of expert knowledge of machine learning within the astronomy community, with experts in astronomy relying either on libraries of easy to use tools or on experts in machine learning to develop and implement machine learning solutions for their domain [18]. This is not ideal, since a major component of machine learning modelling involves selecting which machine learning model to use and identifying which set of hyperparameters- those model parameters that are tuned by an experimenter, rather than those model parameters learnt/trained by the machine learning model- achieve the optimal level of performance for a given model on a specified task or dataset [22, 23]. These two tasks of optimizing model hyperparameters and algorithm selection are optimization procedures [10], and while they may be considered separately, the problem of combined algorithm selection and hyper-parameter optimization (CASH) is of interest for its ability to select models and optimize their hyperparameter configurations without the need for a machine learning expert's intervention [15, 31]. Consequently, solving this problem manually is not something that is immediately accessible to those who are not experts in any field of machine learning [22], and one that is not amenable to an exhaustive search when expert domain knowledge is not available since machine learning algorithms are often computationally expensive to evaluate [26]. Hence, there is a need for more streamlined tools and algorithms that can perform hyperparameter optimization and algorithm selection for a given task in fields such as astronomy. [22, 26].

In addition to the problem of CASH, the issue of performance of these techniques on computationally expensive functions and on large datasets is also a very active field of study [5, 7, 10, 11, 17, 29, 33]. Works such as dataset sub-sampling [1], evaluation time and cost/error function modelling [17], using expert knowledge to accelerate searches [10, 13], and neural networks to approximate

[1] Phenomenon classification involves classifying astronomical objects and events into various classes. Examples include galaxy classification [21], X-ray source identification and classification [24], and supernovae classification [18, 32].

distributions of the evaluation function [28] are some such approaches to this problem. Evidently, astronomy will benefit greatly from using techniques that will both bolster the efficiency of machine learning models and automate the process of model selection and hyperparameter optimization for astronomers entirely. This work will aim to examine the viability of using CASH tools for astronomy domain problems, and following this will examine the performance gains that can be achieved from using various data-efficient extensions to the CASH tools examined in this work.

This paper will be structured as follows: An initial set of preliminaries defining the problem of CASH will be discussed, and then the context in which CASH will be useful for astronomy will be shown using prominent examples of machine learning in astronomy. Then, in the following section, various approaches to the separate problems of automatic algorithm selection and hyperparameter optimization will be explored for the Galaxy Zoo dataset, with a discussion into the results. In all examinations of previous work, strengths and weaknesses of said works will be examined. Finally, the efficacy and potential of hyperparameter optimization to contribute to a cognitive system for astronomy will be examined by looking at integrating the technique with methods such as feature extraction and selection. In doing so, some of the questions this paper hopes to answer are: Can state-of-the-art algorithms for the problem of combined algorithm selection and hyperparameter selection be used to choose and optimize among a set of machine learning algorithms to perform galaxy morphological classification from optical images? Out of these methods, is Bayesian Optimization a more viable approach and what are the possible extensions of Bayesian Optimization variants for dealing with large astronomical datasets? Are these methods more efficient at constructing models than machine learning experts and do they yield better quality hyperparameter configurations?

2 BACKGROUND

In this section, a short overview of the required material on hyperparameter optimization and Bayesian optimization in general will be given as primer for related works and experimental details.

2.1 Hyperparameter Optimization

The problem of hyperparameter optimization for some algorithm A is stipulated in many works such as A. Klein et al. [17], B. Shahriari et al. [26] and C. Thornton et al. [31], and the reader is directed to those resources for a more in-depth introduction. In this work, combined model selection and hyperparameter optimization is considered for the class of classification functions f that take some input set X and maps it to some output set Y, $f : X \rightarrow Y$. Hence, a model within this context is one that takes as input a set of data points $D = \{d_1, d_2, ..., d_n\}$- which is a set of ordered pairs (x_i, y_i) for $i = 1, ..., n$ where each x_i is an element of some input set X and each y_i is an element of some output set Y- and optimizes matching a function to this dataset D by minimizing some specified metric that determines a loss/error function for the classification function f. As C. Thornton et al. [31] mention, the form of f under this learning algorithm A may be understood to be what is known as model parameters of the learning algorithm. For example, G. Luo et al. [22] mention that a function f of, say, simple neural networks has the following model parameters: The weight matrix $\mathbf{W}$ and any regularization functions chosen for neuron outputs. For decision trees, these model parameters would be the thresholds of each node in the tree and the maximum depth of the tree if stipulated [22].

On top of these model parameters, i.e. those that are found by the algorithm when fitting to data, algorithms may also have model hyperparameters encompassed in a vector $\boldsymbol{\lambda}$ from a hyperparameter space $\boldsymbol{\Lambda}$. These hyperparameters are those parameters that are set by the modeller/researcher or one of the hyperparameter optimizers, i.e. meta-processes (to be discussed); additionally, it is more often than not that these hyperparameters remain static during a single evaluation/training epoch of the algorithm A in question [26]. In particular, the i-th hyperparameter in this hyperparameter vector, λ_i, takes on values from a respective domain of values Λ_i, with the vector domain being a subset of the power set of these various domains of hyperparameters. Another useful definition from C. Thornton et al. is that of a conditional hyperparameter: That is, a hyperparameter λ_i is conditional on another such hyperparameter λ_j if λ_i is only active (that is, only affects the algorithm function and performance) if λ_j takes on certain values in its domain (strict subset), denoted by $V_i(j) \subset \Lambda_j$. To denote the dependency of function and performance on the vector of hyperparameters $\boldsymbol{\lambda}$ of the algorithm A, the algorithm will be denoted $A_{\boldsymbol{\lambda}}$ following notation from [31] to capture this dependency. In the example of a neural network, some of these hyperparameters may be the number of neurons per layer, the number of layers, the dropout rate, or the learning rate of the network [22, 31].

Finally then, a formula explaining succinctly the problem statement of hyperparameter optimization is given by

$$\boldsymbol{\lambda}^* = \underset{\boldsymbol{\lambda} \in \boldsymbol{\Lambda}}{\operatorname{argmin}} \frac{1}{k} \sum_{i=1}^{k} \mathcal{L}(A_\lambda, \mathcal{D}_{\text{train}}^{(i)}, \mathcal{D}_{\text{valid}}^{(i)}) \tag{1}$$

which can be understood as finding the set of hyperparameter values such that the loss of the algorithm- evaluated on some cross-validation method of partitioning the dataset- is minimized.

2.2 Model Selection and CASH

A similar definition holds true for the algorithm or model selection step which is defined as in B. Shahriari [26] and C. Thornton et al [31] as follows: Suppose one has a set of algorithms as described above, call it $\mathcal{A}$, and a set of data points as mentioned above, call it $\mathcal{D}$, then an algorithm $A^* \in \mathcal{A}$ is selected such that it has an optimal generalization performance- that is, on some cross-validation scheme, the performance on some partition $\mathcal{D}_{\text{valid}}^{(i)}$ is optimal after training on a test set of data point, $\mathcal{D}_{\text{test}}^{(i)}$- which may be written similarly to equation (1),

$$A^* = \underset{A \in \mathcal{A}}{\operatorname{argmin}} \frac{1}{k} \sum_{i=1}^{k} \mathcal{L}(A, \mathcal{D}_{\text{train}}^{(i)}, \mathcal{D}_{\text{valid}}^{(i)}) \tag{2}$$

which may be understood as minimizing the loss function of this algorithm over the data points partitioned into test and validation

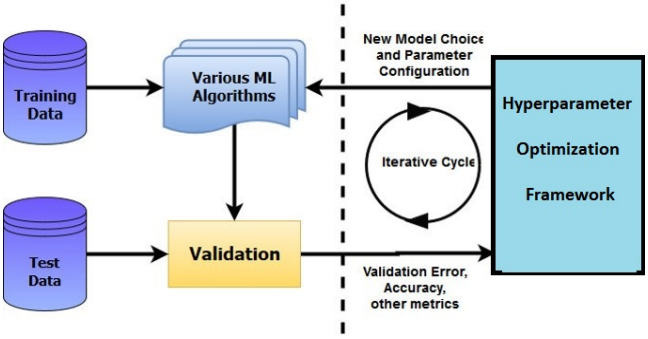

Figure 1: Iterative process of selecting model and model hyperparameter configurations using some hyperparameter optimization tool to perform CASH

sets.

Finally then, the problem of CASH can be stipulated as finding

$$A^*_{\boldsymbol{\lambda}^*} = \underset{\boldsymbol{\lambda}^* \in \boldsymbol{\Lambda}^{(j)}, A^{(j)} \in \mathcal{A}}{\operatorname{argmin}} \frac{1}{k} \sum_{i=1}^{k} \mathcal{L}(A^{(j)}_{\boldsymbol{\lambda}}, \mathcal{D}^{(i)}_{\text{train}}, \mathcal{D}^{(i)}_{\text{valid}}) \tag{3}$$

where the j-th algorithm $A^{(j)}$ has hyperparameter space $\mathcal{A}^{(j)}$. The overall component described which addresses CASH can be visualized in Figure 1 below. This figure illustrates the process of training machine learning models and model configurations on different k-fold cross validation training-validation dataset pairs in an iterative style process until some stopping criterion is met. In general, this stopping criterion is either the number of evaluations, the runtime, the quality of solution found (given by some error metric), or number of models trained.

2.3 Bayesian Optimization

Bayesian Optimization (BO), as with the previous algorithms mentioned above, aims to minimize an algorithm $A_{\boldsymbol{\lambda}}$ loss function by choosing an optimal set of hyperparameters $\boldsymbol{\lambda}^* \in \boldsymbol{\Lambda}$. The difference between this paradigm for CASH is that this method aims to "learn"or to reconstruct the function to be optimized by using a probability distribution over the possible values for this function. It then takes advantage of this probability distribution over hyperparameter configuration space to explore new hyperparameter configurations to query on the function, which in this context is the algorithm executed with the hyperparameters on the chosen data with return signal in the form of some error or loss. The observed set pairs of loss and hyperparameter configurations $\mathcal{D} = \{(\boldsymbol{\lambda}, y)\}$ form the data/observations for the BO method to update its distribution or model of the function to evaluate. This is an iterative process which is followed until some termination criterio is reached. As stated by B. Sharhiari et al., Bayesian Optimization is best-suited to applications where the function is computationally expensive to evaluate and typically has no closed form. Moreover, since BO is able to operate in data sparse application domains, non-differentiable, or multi-modal (many global optima) and non-convex (many local optima) functions may also be optimized by this approach. The details of this will be discussed below in what will be a short introduction to the theory behind Bayesian Optimization.

Definitions and notation that follow are as they have been used in B. Sharhiari et al. [26] and J. Snoek et al. [27]. For starters, items mentioned under the CASH problem description are renamed: the hyperparameter configuration $\boldsymbol{\lambda}$ is denoted as $\boldsymbol{x}$, and the algorithm using these hyperparameters is now denoted as an arbitrary function $f(\boldsymbol{x})$ (as a function that is evaluated using some hyperparameter configuration $\boldsymbol{x}$, on some implicit data, producing some error/loss to be minimized) with the hyperparameter optimization problem becoming the problem of finding $\boldsymbol{x}^*$ as follows

$$\boldsymbol{x}^* = \underset{\boldsymbol{x} \in \mathcal{X}}{\operatorname{argmin}}\, f(x) \tag{4}$$

The problem of CASH can also restated in this notation as

$$\boldsymbol{f}^*(\boldsymbol{x}^*) = \underset{\boldsymbol{x}_j \in \mathcal{X}_i, f_i \in \mathcal{F}}{\operatorname{argmin}}\, f_i(\boldsymbol{x}_j) \tag{5}$$

where $\mathcal{F}$ being the set of all such possible functions under consideration for model selection under the BO method.

In short, Bayesian Optimization models the function under evaluation in an iterative fashion in what is known as a Sequential Model-Based Optimization (SMBO) process: Initially a model- in this case a prior distribution- $p(\boldsymbol{x})$ is specified over the hyperparameters; for the example of a single hyperparameter (unidimensional case) this prior distribution yields an average and variance for values of the hyperparameter. Then, in an iterative fashion, given data points on the nth iteration, the model is updated in a Bayesian fashion,

$$p(\boldsymbol{x}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{x})p(\boldsymbol{x})}{p(\mathcal{D})} \tag{6}$$

with $p(\boldsymbol{x}|\mathcal{D})$ known as the posterior distribution, encapsulating the probability of a hyperparameter configuration $\boldsymbol{x}$ given the nth set of observations $\mathcal{D}_n$. A new query point, i.e. a new hyperparameter configuration, $\boldsymbol{x}'$ is then chosen using an acquisition or utility function $\alpha_n : \mathcal{X} \mapsto \mathbb{R}$ which is a function that gauges the 'usefulness' of evaluating a certain hyperparameter configuration (note that this may be dependent on the nth set of data points and hence this appears in the formula). The next query point then determined by maximizing this acquisition function with respect to hyperparameter configurations. One can think of the acquisition function as introducing the ability for exploration versus exploitation: A greedy utility function can be chosen to select points with the least variance as specified by the posterior distribution, or an explorative utility function that takes the point with largest such variance. Of course, there are more complex examples of utility function that vary from one BO method to the next (such as Expected Improvement or Maximum Improvement [27]), and while these functions may- and often are- very expensive to compute, they are generally not as hard to compute as the function f one is aiming to optimize in configuration space [27]. Once the new point is chosen and evaluated using f, a new data point is created which may then be used to update the posterior distribution as above to enter the $n+1$th iteration. This is repeated until some convergence criterion is achieved. The process is illustrated as algorithm 1 below.

Algorithm 1 Bayesian optimization

1: **for** $n = 1, 2, \ldots$ **do**
2: select new $\mathbf{x}_{n+1}$ by optimizing acquisition function α

$$\mathbf{x}_{n+1} = \arg\max_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{D}_n)$$

3: query objective function to obtain y_{n+1}
4: augment data $\mathcal{D}_{n+1} = \{\mathcal{D}_n, (\mathbf{x}_{n+1}, y_{n+1})\}$
5: update statistical model
6: **end for**

Figure 2: General algorithm for a Bayesian Optimization strategy [26]

One way of realizing the search through a configuration space in a Bayesian optimization space, is by using a Gaussian process. A Gaussian process (GP) is a process where any subset of a set of observations is assumed to follow some underlying Gaussian distribution with mean μ and variance σ^2, and is a way of generalizing regression in the above algorithm, among others. More rigorously, a GP is completely specified by a mean function $\mu_0 : \mathcal{X} \mapsto \mathbb{R}$ and a covariance function (known as a kernel) $k : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$, and where data points are such that for an collection of data points $\boldsymbol{x}_{1:n}$[2], the function values $\boldsymbol{f} = f_{1:n}$ (with $f_i = f(\boldsymbol{x_i})$) are jointly Gaussian. These two functions are are parameters to the GP and hence to the BO using a GP. This is a justified addition of hyperparameters however, since most problems requiring BO span tens or hundreds of hyperparameters and hence are far less tractable than dealing with a handful of hyperparameters.

Research in accelerating BO that uses expensive to evaluate GPs is certainly an active area [11], with various approximate techniques such as Sparse Spectrum Gaussian Processes (SSGP) and Random Forests [26, 27] being examples of prominent alternatives.

In the illustrative example in Figure 1 below, the BO is shown is shown for a Gaussian Process with an arbitrary kernel, mean and utility function. Steps 5 and 6 are shown. In step 5 one can see that the probability distribution is completely certain around data points, becoming less so further away from data. The acquisition function chosen here is a maximum around the yellow star, and hence the function determines this as the next query point. The following step shows the updated posterior distribution given the data. The solid blue line represents the true function values.

A caveat to note for Gaussian processes is the following: Handling of discrete and conditional hyperparameter choices can be done if the GP is allowed to act on a continuous space, with the evaluations/black-box function merely translating hyperparameters into appropriate integer values using a floor or ceiling function.

3 REVIEW OF ALGORITHMS FOR CASH

There are various algorithms that attempt to tackle the problem of combined algorithm selection and hyperparameter optimization.

[2]Notation is as in B. Sharhiari et al. [26], with $\boldsymbol{x_{1:n}} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ representing a set of values

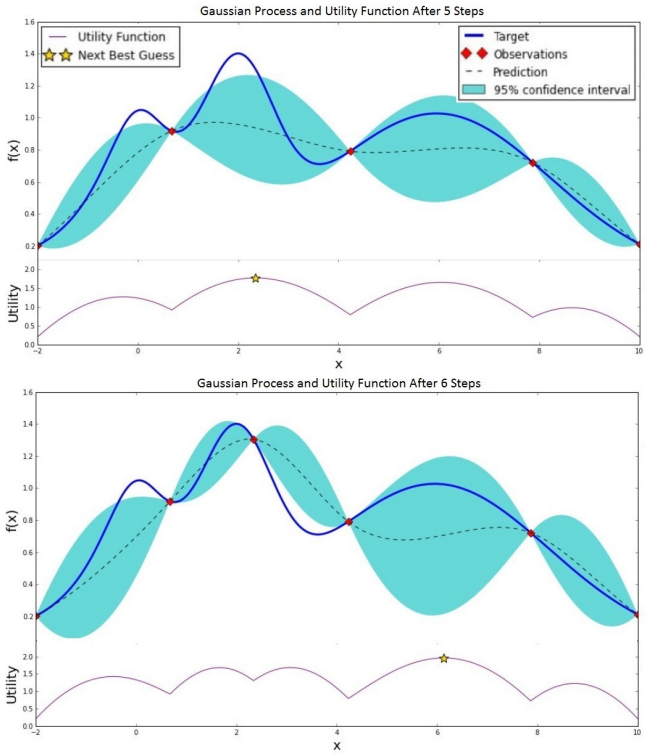

Figure 3: Bayesian optimization with Gaussian Process and Utility function at two steps, as in B. Sharhriari et al. [26]

These are: Grid searches; random searches; evolutionary algorithms; Hypergradients; Hyperband (which builds on successive halving); and Bayesian optimization (with different variants of this). Evolutionary algorithms and Hypergradients are not considered in this review of algorithms for CASH because evolutionary algorithm approaches- as described in Z. Young et al. [35]- have large number of additional hyperparameters to optimize, while Hypergradients [12] require numerically differentiable hyperparameters and therefore cannot deal with categorical and conditional hyperparameters in any form. Therefore, out of the algorithm mentioned above Bayesian optimization, random searches, Hyperband, and extensions to Bayesian optimization for large datasets will be considered and discussed below as candidates for CASH in astronomy.

3.1 Manual, Grid, and Random Searches

Two simple approaches for CASH are those of a grid search or exhaustive search (one that searches all possible hyperparameter configurations) and a random search (random selection of possible hyperparameter configurations), both of which are described the work J. Bergstra et al.[2]. As J. Bergstra et al. point out, these methods are the most commonly used in practice for hyperparameter tuning, along with guided manual searches using machine learning expert knowledge, since they are easy to implement and generally work well when guided with expert knowledge of hyperparameter configuration spaces. While offering good performance on small data sets, these methods suffer from various drawbacks for novice

machine learning tool users: Firstly, the random search method described in J. Bergstra et al. [2] is shown by G. Luo [22] to not outperform domain experts in selecting hyperparameters. Furthermore, the exhaustive nature of grid searches and manual searches leave them as intractable methods for much larger problem domains where an evaluation function must be evaluated on very large datasets, where the evaluation function is itself very computationally expensive- taking days, weeks or even months to execute- or where the dimensionality of the hyperparameter search space is very high. The variant of random search introduced by J. Bergstra et al. [3] was shown to outperform grid search in performance and quality of final configurations and therefore remains a ubiquitous method of performing HPO [6]. For this reason, it is often beneficial to use a random search, rather than a grid search, as a baseline when evaluating other algorithms for hyperparameter optimization. While many extensions to random search exist, such as annealing search [10, 11] and entropy search [14], the simplicity- and hence extra speed- that one achieves on a random search makes random search a better baseline method to use.

3.2 GP/TPE/SMAC-based BO

There are some considerable drawbacks of current SMBO techniques grounded in Gaussian processes that prevent its use for general CASH tasks: The first of these is that GPs only support continuous numeric parameters; secondly, they only work on single instances, making combined algorithm and hyperparameter selection intractable for such methods; and lastly, they lack mechanisms for early-stopping, that is, terminating expensive and/or poor runs early. To combat this, various approximate techniques are used for SMBO such as TPE [4] and SMAC [16]. These methods both implement BO using tree-based methods rather than Gaussian processes, where random forests coupled with parzen estimators are used in the case of TPE, and regression trees are used in the case of SMAC to model the evaluation function response surfaces. The benefits of using tree-based methods is that conditional and categorical (discrete) hyperparameter choices become more natural to express in these algorithms. As a result, not only do these outperform random searches, but they also outperform Gaussian processes in spaces that contain many discrete or categorical choices, since Gaussian processes tend to explore "continuous gaps"between the discrete values of interest. These qualities place TPE and SMAC among the favoured algorithms for CASH in astronomy, where machine learning model choices may be modelled as a discrete hyperparameter choice. Despite this however, these two techniques still require training on the full dataset, and do not provide a means to allow early stopping of algorithm training and evaluation.

3.3 Hyperband and Successive Halving

L. Li et al. [20] and B. Zoph et al. [36] both implement basic reinforcement learning approaches to tackling algorithm selection and hyperparameter optimization. L. Li et al. develop HyperBand, and consider the problem akin to that of an n-armed bandit for each algorithm choice available (n choices) and the reward signal being the loss of the algorithm chosen for evaluation. The various populations of configurations samples are then culled using a successive halving algorithm, described in the paper, and the configurations adjusted slightly based on a simple Bayesian inference of the nearby configuration space. The advantages of this method of HPO is that it is very effective at finding good configurations very quickly, outperforming even tried and tested methods of GPs, TPE, and even SMAC on some problems. Some disadvantages is that methods such as these are still very data-hungry, and rely on maintaining a large pool of candidate configurations that have been preselected in some way, either randomly or with some specified prior distribution. However, while this may seem like a drawback, this does lend itself to a simple parallelization task. Work by A. Klein et al. [17] also shows that Hyperband is very effective when an additional dataset size hyperparameter is introduced into each model. Its speed of finding superior configurations and the possibility of introducing a form of data sub-sampling into model evaluations makes Hyperband a promising candidate for use in CASH for astronomy.

3.4 Extensions of Bayesian Optimization

One method of extending Bayesian optimization is to make it more robust and efficient on large datasets. To achieve this, many previous authors [17, 30**?**]Multi-Task Bayesian Optimization (MTBO) by K. Swersky et al. [30] emerges as one technique that aims to achieve this by introducing dataset size as an additional task to optimize in their list of multi-objective optimization tasks, hence treating it as a free parameter to tune, and by concurrently training many different configurations and models in parallel and incorporating results into a multi-object expectation maximization step. In a similar fashion, one can introduce dataset size as a free parameter in the Hyperband algorithm above, as has been done by A. Klein et al. [17] in their experiments. Both Hyperband and MTBO allow parallel execution of similar models to be executed, while using information to improve searches for single model configurations. Both Hyperband and MTBO show promise in being robust methods for CASH and efficient methods for astronomy.

Another robust technique to consider is FABOLAS (FAst Bayesian Optimization for LArge DatasetS), hereto referred to as Fabolas, by A. Klein et al [17] that treats dataset size as a free parameter open to tuning and extrapolates function loss/error rates for future function evaluations by modelling both dataset size and function loss as latent variables using past data points and experiments in order to further minimize the number of function evaluations performed by their BO procedure. Overall, reducing the number of function evaluations and introducing a tunable data set size parameter, makes Fabolas efficient on large data set sizes and expensive algorithms/functions. As a result, this method is amicable to CASH for astronomy, allowing one to use large data sets such as the Galaxy Zoo and SDSS datasets, as well as computationally heavy/expensive learning algorithms such as convolutional neural networks. Both these techniques show results on benchmark datasets that achieves similar or better performance than previous methods such as TPE, SMAC, and Hyperband.

As a possible way of making BO more robust, Z. Wang et al. [33] developed a method called Random Embedding Bayesian Optimization (REMBO) that can operate on data with on the order of billions of dimensions, that is, more formally, data sets $\mathcal{X} \subset \mathbb{R}^D$ where

the number D can be in the billions. The method achieves this by finding a linear subspace of the original space $\mathbb{R}^D$ over which the function is invariant over a subset of the dimensions on all data points in this subspace. Within the context of astronomy, however, hyperparameter configuration spaces for machine learning models used in the domain are not usually very high in dimension, and so are not amicable to benefit from REMBO, which as the authors comment works best in high dimensional spaces.

A neat summary of the various techniques for CASH is displayed in Table 1 below. Some implementations and links to implementations of the tools referenced above are also included.

4 EXPERIMENTAL DESIGN

The experiments conducted in this work serve two purposes. To first explore the efficacy and viability of hyperparameter optimization approaches such as Bayesian optimization for the domain area of astronomy and in particular for galaxy morphology classification, and to explore the effects of different hyperparameter optimization algorithms, some with data-efficient extensions, on performance and quality of machine learning configurations found for the problem of galaxy morphology classification. Two experiments to accomplish this are devised below, with further details on hardware and software discussed afterwards.

4.1 Hyperparameter Optimization for Astronomy

The first goal of this work is to determine the feasibility of hyperparameter optimization for astronomy. To determine this, hyperparameter optimization will be tested against human experts for tuning hyperparameter configurations for machine learning algorithms and/or outperform given methods for hyperparameter optimization for astronomy domain problems. Considering the latter has not been done before, the former option is explored in these experiments. Rather than collect data by using expert users to tune algorithms, this task of testing against human expert performance will be accomplished by using results on problems quoted SKYNET [13]. SKYNET uses warm-starting with expert knowledge of configurations, together with a Newton's method to make adjustments to hyperparameters such as the learning rate. In SKYNET, various results of machine learning algorithms on both toy and real data are employed for both classification and regression tasks. These datasets consisted of

- **The Sinc Function** regression dataset given by $y = \sin(x)/x + 0.04x$ with Gaussian noise of mean 0 and deviation 0.05 added. From this, 200 points were taken for training and 100 for validation.
- **The Radford Neal** classification dataset is a toy dataset developed to test the ability to handle simple non-linearity.
- **The Dark Matter Sky Survey** dataset from Kaggle [3]. Here, users needed to use a coupled galaxy and star pair of images to denoise the galaxy image and to recover the 2 ellipticity parameters of the galaxy. Consequently, this is a regression problem.
- **The Galaxy Zoo** dataset from Kaggle [4]. This dataset consists of close to 100 000 optical images of galaxies. The goal of this challenge is to replicate the 37 feature probabilities that crowd-sourced users produced. This means that this lends itself to a regression problem.
- **A Supernovae** classification dataset using data provided by the SDSS via the Python AstroML package. This dataset presented 1D time-series light-curves of various known supernovae events. The task is to classify the supernovae into one of three classes, type IA, IB or IIA.
- **The MNIST** dataset. This is the standard machine learning classification task using 60 000 training images and 10 000 validation images of handwritten digits ranging from 0 to 9.

Experiments were run to try and replicate and potentially improve on the quoted RMSE and classification accuracy scores for these problems that were solved by the SKYNET system and for the top solution on the Galaxy Zoo leaderboard published by S. Dieleman [9]. The hyperparameter optimization tools chosen for these experiments were HyperOpt and Fabolas. HyperOpt was selected for its ability to explicitly deal with categorical hyperparameters via its TPE implementation, and for its ease of use with tools such as scikit-learn in Python. On top of its ease of implementation, HyperOpt was chosen to negate any additional effects that data-efficient algorithms such as Fabolas would create, which are subsequently explored in the Fabolas experiments. Fabolas was chosen for its reported state-of-the-art performance on large dataset sizes. HyperOpt was run for each of the problems listed above, while Fabolas was only run for the MNIST, Dark Matter, and Galaxy Zoo datasets, since performance gains were negligible with the smaller datasets.

The next set of experiments were focused on the Galaxy Zoo dataset and aimed to examine performance of different hyperparameter optimizers by examining the rate of convergence of these to high quality solutions. To perform this experiment, various software implementations of algorithms discussed from literature were used on the Galaxy Zoo dataset and a performance evaluation was conducted. Furthermore, this was done using 4 different sets of features extracted from galaxies: The WND-CHARM feature extraction method [25], custom shape descriptors, and a VGG16 Convoutional neural net coupled with tSNE feature extraction approach discussed below. All quoted results stem from evaluations on unseen test data, and all training is done using k-fold cross validation, where k is equal to the number of hyperparameter choices. This is to prevent over-fitting the training data and moreover to prevent fitting to the validation data when experimenting with different model hyperparameter configurations.

4.2 Performance on Large Datasets

In the second round of experiments, various HPO algorithm implementations listed in Table 1 are used to evaluate the time it takes to find well-performing solutions on large datasets. For this, the largest dataset available, the Galaxy Zoo, is used. Preprocessing for features is done as described in the section below. To measure the efficiency of the various HPO techniques, the best solution

[3] .kaggle.com/c/mdm

[4] www.kaggle.com/c/galaxy-zoo-the-galaxy-challenge

Table 1: A list of tools for implementing algorithms for CASH

Name	Tool	Link	Description
SMAC [16]	SMAC3	https://github.com/automl/SMAC3	A hyperparameter optimizer and model selector implementing SMAC
Fabolas [17]	RoBO	https://github.com/automl/RoBO	A Gaussian process based implementation of Fabolas
Hyperband [20]	RoBO	https://github.com/zygmuntz/hyperband	A hyperparameter optimizer and model selector using Successive Halving and Armed Bandit selection
TPE and Random [4]	HyperOpt	https://github.com/hyperopt/hyperopt	A Python framework for bayesian optimization with TPE
MTBO [30]	RoBO	https://github.com/automl/RoBO	A Python implementation of multi-task Bayesian optimization

(measured by RMSE) is captured as a function of runtime, up to a maximum runtime of 12 hours.

4.3 Data Preprocessing

For all experiments conducted on the MNIST, sinc function, Dark Matter, and supernovae datasets, no preprocessing was conducted other than flattening any images into the corresponding row vector (e.g. an image of dimension 28×28 becomes a row vector of length 784). For the Galaxy Zoo dataset, the following preprocessing techniques were done: Images are cropped to size 214 in each of the 2 dimensions and after centering around the galaxy, after which downsampling $3x$ and grayscaling is performed. Then, as described in S. Dieleman et al. [9], affine transformation (rotations, scaling, translation, and reflections) are applied at random to images to generate more training examples and to make trained networks more invariant to such transformations. This benefited their algorithm and is hence also done here as a preprocessing step before extracting features.

Finally, to extract feature vectors, rather than flattening out the images into row vectors, the VGG16 network with ImageNet weights was used, as described by A Krizhevsky et al. [19], was used to extract image features by using the final convolutional layer output as the feature vector. Following this, t-SNE (or other dimensionality reduction techniques such as MDS, PCA, etc.) is used to reduce the set of 150 features down to just 4. Hyperparameter optimization of the VGG16, and of the t-SNE hyperparameters used in this step is not explored and is left to future work.

4.4 Hardware and Software

Hardware used for experiments was both the CHPC Lengau cluster (chpc.ac.za) and the UCT HPC Hex cluster (hex.uct.ac.za). Each node on these clusters contains 4 Intel Xenon CPUs with 16 cores each, with 2Gb of RAM per core. The UCT HPC also provides access to NVIDIA Quadro GPUs. GPUs used were both the aforementioned NVIDIA Quadro and NVIDIA GTX 670MX cards for training neural network models in Keras. Software used was Python 3.5, with Scikit-learn and Keras (Tensorflow backend) packages (with their dependencies) for implementing the machine learning component of the project. Tools used in the conducted experiments were HyperOpt, SMAC3, RoBO (which contains implementations of Fabolas, Hyperband, and MTBO). These hyperparameter optimization packages, with references to descriptive papers and links to Github pages, are displayed in Table 1.

4.5 Machine Learning Models, Execution Details, and Evaluations

Machine learning models used for the experiments include the following: Neural Networks (with possible convolutions included), Support Vector Machines (SVMs), Random Forests, Extra Trees, Gradient Boosted Trees, Gaussian Mixture Models, K Nearest Neighbours, and Lasso. Hyperparameters exposed to the various tools listed above for configuration options are listed in the appendix available on the ASTCVS website.

Experiments aimed at evaluating HPO for Astronomy were conducted by allowing HyperOpt to optimize the set of algorithms mentioned above in a CASH setting, for no longer than 12 hours on each setting. This hard cut-off was chosen due to time-constraints, and due to convergence of HyperOpt scores after 8-10 hours becoming apparent. Experiments aimed at measuring efficiency of various hyperparameter optimization tools on large datasets were conducted by allowing each hyperparameter tool to run for 12 hours on the Galaxy Zoo dataset features for the same reasons mentioned above.

Machine learning methods used were evaluated on regression problems using the Root Mean Squared Error (RMSE) and on the accuracy score (together with confusion matrices) for classification problems. These two metrics were chosen for two reasons. Firstly, RMSE and accuracy scores are used for comparison purposes with SKYNET for the sinc function, Dark Matter survey, Neal dataset, MNIST, and supernovae. Secondly, the Dark Matter and Galaxy Zoo Kaggle competitions used RMSE as an evaluation metric, so this was selected for comparison to leaderboard scores.

5 RESULTS AND DISCUSSION

In this section, results for the sets of experiments described are presented and discussed. First, the viability of HPO for astronomy will be examined, followed by performance of HPO algorithms

on large datasets, and then a brief exploration into other feature extraction techniques will be conducted.

Results from the first set of experiments using HPO methods against SKYNET are shown in Table 2. These results show the best HyperOpt and Fabolas runs with both toy and real dataset problems. Fabolas found the best configurations (hyperparameters for which may be found on the ASTCVS website) for the Galaxy Zoo and Dark Matter surveys, and in all other cases except for the Neal dataset, HyperOpt found hyperparameter configurations that performed better than those found by SKYNET. With more trial runs of HyperOpt, the discrepancy between the SKYNET and HPO results for Neal is likely to become statistically insignificant. Confusion matrices for the Neal and MNIST problems are shown in Figures 4 and 5 respectively. For the Neal dataset, the misclassification in Region 2 and 3 relates to the strong non-linearity and sharing of data points between these two regions. For the MNIST confusion matrix, deviations in digits 7 and 9 can be attributed to similar features present in badly drawn examples. This may later be remedied by adding convolutions to the networks to attempt to perform better feature extraction.

These results are indicators that statistical and probabilistic approaches are more conducive for hyperparameter optimization than human-expert led searches, even when supplemented by analytical methods as in the case of SKYNET. Even more interesting is that these results were found even after limiting hyperparameter searches to only neural networks, which vastly culls the diversity in solutions that could be found if other models such as SVMs or GMMs were added to the configuration space. This leaves room for further experimentation and exploration.

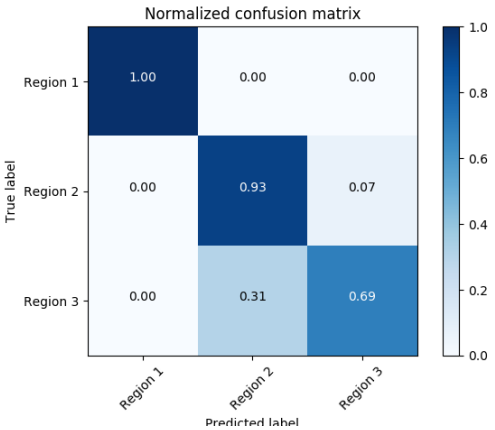

Figure 4: Confusion matrix for the Neal dataset. The misclassification in Region 2 and 3 relates to the strong non-linearity and sharing of data points between these two regions (see SKYNET [13] for an example plot)

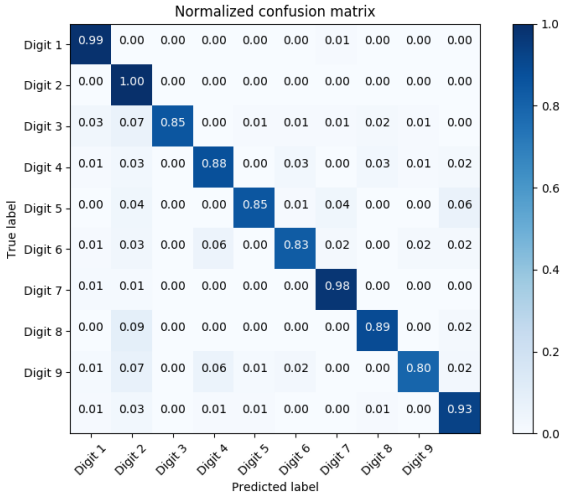

Figure 5: Confusion matrix for the MNIST problem. Here we see that most digits are accurately classified, with misclassification occurring in digits 2 and 8, and between 4 and 6, where these two digits might easily be mistaken for one another. Addition of convolutions to the network may improve scores in the confusion matrix.

Table 2: RMSE and accuracy comparisons of HyperOpt vs. SkyNet

Problem	RMSE	
	SKYNET	HPO
Dark Matter	0,0171	0,01596
Galaxy Zoo	0,0786	0,0643
Toy	0,4562	0,3326
	Accuracy %	
Radford Neal	0,24	0,25
Supernovae	85,6	89,4
MNIST	97,2	98,8

In the second set of experiments, best configuration scores were recorded as functions of runtime for the various HPO methods in Table 1. In the figure below, the best configuration RMSE loss is shown as a function of runtime using the VGG16/ImageNet features on the Galaxy Zoo dataset. Random search is shown to converge to similar quality solutions eventually after about 600 minutes, but only after a long halt in improvements. This is a signature of the randomness of the annealing procedure implemented by HyperOpt's random configuration suggestion algorithm from J. Bergstra [4]. Among the random, TPE, and SMAC algorithms for BO, SMAC finds the overall best quality solutions but only after a short time exploring poor configuration settings, while both TPE and SMAC outperform a pure random search. These results are attributed to the Bayesian inference and of the exploration vs exploitation nature of the EI step used by both algorithms to choose unseen configuration spaces for the set of machine learning algorithms available. Specifically, while TPE models this using random forests, SMAC chooses regression forests. The difference being that TPE has a

greater deal of randomness and thus is less likely to become stuck in local optima for as long periods of time as SMAC. Finally, and very promisingly, methods such as MTBO and Fabolas show very fast convergence to very low RMSE scoring models and model configurations. The best models found were extra trees with an RMSE score of 0.069. and a neural network with RMSE of 0.064. This also outperforms the highest quality solution on the Kaggle Galaxy Zoo challenge leaderboards, further indicating the efficacy of HPO for CASH and for astronomy problem domains. The parameters for the best known models can be found as an appendix on the website of the ASTCVS project.

Results using Fabolas show that the entire dataset need not be considered when selecting algorithms, choosing algorithm configurations, and training algorithms. Fabolas shows that the cost of general and expensive black-box functions can indeed be modelled over the entire dataset when considering only subsamples of the data. The reason for such stark differences between the MTBO and Fabolas runtime curves and those of traditional BO methods is that Fabolas and MTBO do not need to evaluate a given model configuration on the entire dataset in order to infer a suitable value for the expected function loss. This modelling of function loss on small subsamples of the full dataset means that Fabolas and MTBO can both evaluate many more models and hyperparameter configurations in a given amount of time than other methods of BO that do not perform dataset subsampling and do not support early stopping. While Hyperband can support dataset subsampling, the tools does not allow for early stopping of function evaluations. The ability for Fabolas and MTBO to perform a larger number of function evaluations per unit time allows for these techniques to explore a large region of the configuration space in a shorter amount of time than other methods, and hence the large difference in the runtime curves becomes less surprising as a result.

Finally, the effect of the different feature extraction techniques (VGG16/ImageNet, WND-CHARM, and shape-descriptors) is shown in Table 3 by comparing different best-found RMSE values using the Fabolas and Hyperband methods. It is evident that RMSE values of configurations found using these techniques do not perform as well as using unsupervised feature extraction techniques such as ImageNet, coupled with tSNE. This does not come as much of a surprising result, since convolutions are still state-of-the-art in the field of image classification [18]. However, this may change when other datasets such as 1D time-series, and radio astronomy datasets are explored in further works, where convolutional approaches may begin to suffer and where other techniques for feature extraction such as time-series shapelets, and where LSTM networks may improve. In addition to this, the hand-crafted features used in this work relied strongly on various hyperparameters chosen by the researcher, which may lend itself to hyperparameter optimization approaches. This, however, is left to future work in exploring HPO for selecting feature extraction methods and hyperparameters for these methods, coupled with evaluations using machine learning algorithms.

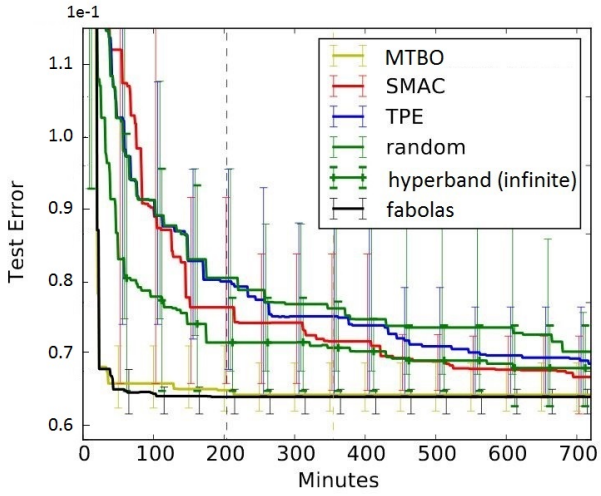

Figure 6: Best (minimum) RMSE as a function of runtime for various HPO methods. Out of traditional BO methods, SMAC performs best. Data efficient techniques such as Fabolas and MTBO find solutions at a faster rate than any other HPO algorithm.

6 CONCLUSIONS AND FUTURE WORK

The feasibility of using hyperparameter optimization for the problem domain area of astronomy certainly exists. In this work, HPO was shown to outperform existing methods of optimizing model configurations such as SKYNET [13]. This opens up doors into future work for finding and improving other existing models and algorithm configurations for astronomy, which could potentially reduce computational costs and decrease runtime on these problems where simpler and computationally cheaper configurations of models are selected by HPO models to solve these problems. Further, future work in developing machine learning models for automated data processing in astronomy may also be accelerated if adoption of these HPO tools becomes more wide-spread.

Along with this, various algorithms for HPO were evaluated on the Galaxy Zoo dataset, showing the effectiveness of such algorithms in outperforming state-of-the-art solutions such as those published from top-performing Kaggle solutions [9], and on increasing the efficiency of methods on large dataset sizes, by training configurations on smaller sub-samples of the full dataset.

In the last set of experiments, data-efficient extensions to Bayesian optimization were able to find better quality solutions in a shorter amount of time (up to 10 times as fast) than traditional methods of BO such as TPE and SMAC. Future studies for hyperparameter configuration will make great headway in pursuing the avenue of extending techniques such as Fabolas to deal with conditional and categorical hyperparameter choices explicitly, through the use of similar techniques as those in TPE and SMAC, rather than through Gaussian processes. Potential also exists for further incorporation of expert knowledge in “warm-starting ”the hyperparameter learning procedure with expert guesses for configurations in a similar

fashion to works such as SKYNET. Additional studies include performing HPO on feature extraction and selection techniques in order to optimize the best features for a given dataset or problem domain. By performing these extensions, one can hope to not only optimize the procedure of optimizing choices of machine learning model along with model hyperparameters, but also to choose appropriate preprocessing and feature extraction and selection techniques to use for a given problem.

In summary, applying state-of-the-art hyperparameter optimization methods to problems in an astronomy domain has shown that these methods are a viable and efficient approach for selecting and configuring machine learning solutions to a number of problems. Moreover, techniques for reducing expensive algorithm training overheads via early stopping and for dealing with large datasets via dataset sub-sampling are an effective way to learn hyperparameters and to train machine learning algorithms on large datasets such as the Galaxy Zoo.

REFERENCES

[1] Alnur Ali, Rich Caruana, and Ashish Kapoor. 2014. Active Learning with Model Selection. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI'14)*. AAAI Press, QuÃĪbec City, QuÃĪbec, Canada, 1673–1679. http://dl.acm.org/citation.cfm?id=2892753.2892785

[2] James Bergstra and Yoshua Bengio. 2012. Random Search for Hyper-parameter Optimization. *J. Mach. Learn. Res.* 13, 1 (Feb. 2012), 281–305. http://dl.acm.org/citation.cfm?id=2503308.2188395

[3] James Bergstra, Brent Komer, Chris Eliasmith, and David Warde-Farley. 2014. Preliminary evaluation of hyperopt algorithms on HPOLib. In *ICML workshop on AutoML*. http://ai2-s2-pdfs.s3.amazonaws.com/0da5/366fb41f66827c89144e5a949cc0e7e27d47.pdf

[4] James Bergstra, Brent Komer, Chris Eliasmith, Dan Yamins, and David D. Cox. 2015. Hyperopt: a Python library for model selection and hyperparameter optimization. *Comput. Sci. Disc.* 8, 1 (2015), 014008. https://doi.org/10.1088/1749-4699/8/1/014008

[5] J. Bergstra, D. Yamins, and D. D. Cox. 2013. Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28 (ICML'13)*. JMLR.org, Atlanta, GA, USA, I–115–I–123. http://dl.acm.org/citation.cfm?id=3042817.3042832

[6] James S. Bergstra, RÃĪmi Bardenet, Yoshua Bengio, and BalÃązs KÃľgl. 2011. Algorithms for Hyper-Parameter Optimization. In *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 2546–2554. http://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.pdf

[7] Yutian Chen, Matthew W. Hoffman, Sergio Gomez Colmenarejo, Misha Denil, Timothy P. Lillicrap, and Nando de Freitas. 2016. Learning to Learn for Global Optimization of Black Box Functions. *arXiv:1611.03824 [cs, stat]* (Nov. 2016). http://arxiv.org/abs/1611.03824 arXiv: 1611.03824.

[8] P. E. Dewdney, P. J. Hall, R. T. Schilizzi, and T. J. L. W. Lazio. 2009. The Square Kilometre Array. *Proc. IEEE* 97, 8 (Aug. 2009), 1482–1496. https://doi.org/10.1109/JPROC.2009.2021005

[9] Sander Dieleman, Kyle W Willett, and Joni Dambre. 2015. Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Monthly notices of the royal astronomical society* 450, 2 (2015), 1441–1459.

[10] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter. 2015. Efficient and Robust Automated Machine Learning. In *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS'15)*. MIT Press, Cambridge, MA, USA, 2755–2763. http://dl.acm.org/citation.cfm?id=2969442.2969547

[11] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost T Springenberg, Manuel Blum, and Frank Hutter. 2015. Methods for improving bayesian optimization for AutoML. In *AutoML Workshop, International Conference on Machine Learning*, Vol. 2015.

[12] Luca Franceschi, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. 2017. Forward and Reverse Gradient-Based Hyperparameter Optimization. *arXiv:1703.01785 [stat]* (March 2017). http://arxiv.org/abs/1703.01785 arXiv: 1703.01785.

[13] P. Graff, F. Feroz, M. P. Hobson, and A. Lasenby. 2014. SKYNET: an efficient and robust neural network training tool for machine learning in astronomy. *Monthly Notices of the Royal Astronomical Society* 441, 2 (May 2014), 1741–1759. https://doi.org/10.1093/mnras/stu642

[14] JosÃĹ Miguel HernÃąndez-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. 2014. Predictive Entropy Search for Efficient Global Optimization of Black-box Functions. In *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 918–926. http://papers.nips.cc/paper/5324-predictive-entropy-search-for-efficient-global-optimization-of-black-box-functions.pdf

[15] Matthew Hoffman, Bobak Shahriari, and Nando Freitas. 2014. On correlation and budget constraints in model-based bandit optimization with application to automatic machine learning. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research)*, Samuel Kaski and Jukka Corander (Eds.), Vol. 33. PMLR, Reykjavik, Iceland, 365–374. http://proceedings.mlr.press/v33/hoffman14.html

[16] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. 2011. Sequential Model-Based Optimization for General Algorithm Configuration. *LION* 5 (2011), 507–523.

[17] Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. 2016. Fast Bayesian Optimization of Machine Learning Hyperparameters on Large Datasets. *arXiv:1605.07079 [cs, stat]* (May 2016). http://arxiv.org/abs/1605.07079 arXiv: 1605.07079.

[18] Jan Kremer, Kristoffer Stensbo-Smidt, Fabian Gieseke, Kim Steenstrup Pedersen, and Christian Igel. 2017. Big Universe, Big Data: Machine Learning and Image Analysis for Astronomy. *IEEE Intelligent Systems* 32, 2 (March 2017), 16–22. https://doi.org/10.1109/MIS.2017.40

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[20] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. 2016. Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization. *arXiv:1603.06560 [cs, stat]* (March 2016). http://arxiv.org/abs/1603.06560 arXiv: 1603.06560.

[21] Chris J. Lintott, Kevin Schawinski, AnÅ¿e Slosar, Kate Land, Steven Bamford, Daniel Thomas, M. Jordan Raddick, Robert C. Nichol, Alex Szalay, Dan Andreescu, Phil Murray, and Jan Vandenberg. 2008. Galaxy Zoo: morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey âŸĚ. *Monthly Notices of the Royal Astronomical Society* 389, 3 (Sept. 2008), 1179–1189. https://doi.org/10.1111/j.1365-2966.2008.13689.x

[22] Gang Luo. 2016. A review of automatic selection methods for machine learning algorithms and hyper-parameter values. *Netw Model Anal Health Inform Bioinforma* 5, 1 (Dec. 2016), 18. https://doi.org/10.1007/s13721-016-0125-6

[23] Gustavo Malkomes, Charles Schaff, and Roman Garnett. 2016. Bayesian optimization for automated model selection. In *Advances in Neural Information Processing Systems*. 2900–2908.

[24] M. Olamaie, F. Feroz, K. J. B. Grainge, M. P. Hobson, J. S. Sanders, and R. D. E. Saunders. 2015. Bayes-X: a Bayesian inference tool for the analysis of X-ray observations of galaxy clusters. *Monthly Notices of the Royal Astronomical Society* 446, 2 (Jan. 2015), 1799–1819. https://doi.org/10.1093/mnras/stu2146 arXiv: 1310.1885.

[25] Nikita Orlov, Lior Shamir, Tomasz Macura, Josiah Johnston, D Mark Eckley, and Ilya G Goldberg. 2008. WND-CHARM: Multi-purpose image classification using compound image transforms. *Pattern recognition letters* 29, 11 (2008), 1684–1693.

[26] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. 2016. Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proc. IEEE* 104, 1 (Jan. 2016), 148–175. https://doi.org/10.1109/JPROC.2015.2494218

[27] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 2951–2959. http://papers.nips.cc/paper/4522-practical-bayesian-optimization-of-machine-learning-algorithms.pdf

[28] Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Md Mostofa Ali Patwary, Prabhat, and Ryan P. Adams. 2015. Scalable Bayesian Optimization Using Deep Neural Networks. *arXiv:1502.05700 [stat]* (Feb. 2015). http://arxiv.org/abs/1502.05700 arXiv: 1502.05700.

[29] Jaemin Son, Samarth Gupta, and Gary Tan. 2016. Bayesian Optimization in High Dimensional Input Space. In *Proceedings of the 9th EAI International Conference on Simulation Tools and Techniques (SIMUTOOLS'16)*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, 18–27. http://dl.acm.org/citation.cfm?id=3021426.3021430

[30] Kevin Swersky, Jasper Snoek, and Ryan P Adams. 2013. Multi-task bayesian optimization. In *Advances in neural information processing systems*. 2004–2012.

[31] Chris Thornton, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. 2013. Auto-WEKA: Combined Selection and Hyperparameter Optimization of Classification Algorithms. In *Proceedings of the 19th ACM SIGKDD International*

Conference on Knowledge Discovery and Data Mining (KDD '13). ACM, New York, NY, USA, 847–855. https://doi.org/10.1145/2487575.2487629

[32] Jake Vander Plas, A. J. Connolly, and Z. Ivezic. 2014. AstroML: Python-powered Machine Learning for Astronomy, Vol. 223. 253.01. http://adsabs.harvard.edu/abs/2014AAS...22325301V

[33] Ziyu Wang, Frank Hutter, Masrour Zoghi, David Matheson, and Nando de Freitas. 2016. Bayesian Optimization in a Billion Dimensions via Random Embeddings. *Journal of Artificial Intelligence Research* (2016). https://doi.org/10.1613/jair.4806

[34] D. G. York. 2000. The Sloan Digital Sky Survey: Technical Summary. *The Astronomical Journal* 120, 3 (Sept. 2000), 1579–1587. https://doi.org/10.1086/301513 arXiv: astro-ph/0006396.

[35] Steven R. Young, Derek C. Rose, Thomas P. Karnowski, Seung-Hwan Lim, and Robert M. Patton. 2015. Optimizing Deep Learning Hyper-parameters Through an Evolutionary Algorithm. In *Proceedings of the Workshop on Machine Learning in High-Performance Computing Environments (MLHPC '15)*. ACM, New York, NY, USA, 4:1–4:5. https://doi.org/10.1145/2834892.2834896

[36] Barret Zoph and Quoc V Le. 2016. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578* (2016).