AutoML Loss Landscapes

YASHA PUSHAK, Department of Computer Science, The University of British Columbia, Canada and Oracle Labs, Canada
HOLGER H. HOOS, RWTH Aachen University, Germany, LIACS, Universiteit Leiden, The Netherlands, and Department of Computer Science, The University of British Columbia, Canada

As interest in machine learning and its applications continues to increase, how to choose the best models and hyper-parameter settings becomes more important. This problem is known to be challenging for human experts, and consequently, a growing number of methods have been proposed for solving it, giving rise to the area of automated machine learning (*AutoML*). Many of the most popular AutoML methods are based on Bayesian optimization, which makes only weak assumptions about how modifying hyper-parameters effects the loss of a model. This is a safe assumption that yields robust methods, as the *AutoML loss landscapes* that relate hyper-parameter settings to loss are poorly understood. We build on recent work on the study of one-dimensional slices of algorithm configuration landscapes by introducing new methods that test n-dimensional landscapes for statistical deviations from uni-modality and convexity, and we use them to show that a diverse set of AutoML loss landscapes are highly structured. We introduce a method for assessing the significance of hyper-parameter partial derivatives, which reveals that most (but not all) AutoML loss landscapes have only a small number of hyper-parameters that interact strongly. To further assess hyper-parameter interactions, we introduce a simplistic optimization procedure that assumes each hyper-parameter can be optimized independently, a single time in sequence, and we show that it obtains configurations that are statistically tied with optimal in all of the n-dimensional AutoML loss landcsapes that we studied. Our results suggest many possible new directions for substantially improving the state of the art in AutoML.

CCS Concepts: • **Computing methodologies** → **Machine learning**; *Search methodologies*; Artificial intelligence.

Additional Key Words and Phrases: landscape analysis, AutoML, hyper-parameter optimization

ACM Reference Format:
Yasha Pushak and Holger H. Hoos. 2022. AutoML Loss Landscapes. *ACM Trans. Evol. Learn.* xx, x, Article xxx (x 2022), 29 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Machine learning has made many impressive contributions to a broad range of applications through the development of a diverse set of models and training algorithms (see, *e.g.*, Breiman [2001]; Cortes and Vapnik [1995]; Krizhevsky et al. [2012]; LeCun et al. [2015]; Silver et al. [2016] or Angermueller et al. [2016]). However, no single technique dominates for all applications; therefore, for many applications, there are benefits in evaluating and comparing many approaches. Furthermore, it is well known that hyper-parameter settings can strongly impact model quality [Bergstra et al.

Authors' addresses: Yasha Pushak, ypushak@cs.ubc.ca, Department of Computer Science, The University of British Columbia, 2366 Main Mall, Vancouver, British Columbia, Canada, V6T 1Z4 and Oracle Labs, Vancouver, British Columbia, Canada; Holger H. Hoos, RWTH Aachen University, Theaterstraße 35-39, 52062, Aachen, Germany and LIACS, Universiteit Leiden, Leiden, The Netherlands and Department of Computer Science, The University of British Columbia, 2366 Main Mall, Vancouver, British Columbia, Canada, V6T 1Z4, hh@cs.rwth-aachen.de.

2688-3007/2022/x-ARTxxx $15.00
https://doi.org/10.1145/1122445.1122456

2011]. Recently, significant attention has been devoted to developing automated machine learning (*AutoML*) methods to address this problem (see, *e.g.*, Bergstra et al. [2011]; Bergstra and Bengio [2012]; Falkner et al. [2018]; Feurer et al. [2015]; Fusi et al. [2018]; Kandasamy et al. [2017, 2015]; Li et al. [2017]; Snoek et al. [2012]; Springenberg et al. [2016] or Yang et al. [2019]). AutoML often casts the combined algorithm selection and hyper-parameter optimization problem as a stochastic, black-box optimization task that aims to optimize model quality in terms of generalization loss by finding suitable values for discrete, continuous and conditional hyper-parameters. However, little is known about the *AutoML loss landscapes* that relate hyper-parameter settings to generalization loss [Bergstra et al. 2011; Pimenta et al. 2020], hence most AutoML methods make very few assumptions about these landscapes. The most common methods are based on Bayesian optimization (see, *e.g.*, Bergstra et al. [2011]; Feurer et al. [2015]; Kandasamy et al. [2017]; Snoek et al. [2012] or Falkner et al. [2018]). These methods typically only assume that the solution quality of configurations are locally correlated, and that this correlation can be learned and exploited. Other prominent methods make even fewer assumptions, see *e.g.*, Li et al. [2017, 2020] or Bergstra and Bengio [2012], which rely on random search. It has even been proposed that AutoML methods should perform well on nasty, multi-model synthetic benchmarks, such as rastrigin and hartmann-6 [Eggensperger et al. 2013]. Because these assumptions are very conservative, they can be expected to yield robust methods.

However, all of these methods may be unnecessarily inefficient if AutoML loss landscapes turned out to be highly structured in a way that makes them easier to optimize. For example, if they tended to be globally uni-modal or even convex (for numerical hyper-parameters), this would have significant ramifications on the kinds of search procedures used for AutoML. This motivates our first research question: **RQ 1. Is the global structure of typical AutoML loss landscapes relatively benign; in particular, are they (approximately) uni-modal or convex?** It is already well-known that most AutoML loss landscapes depend most strongly on a small number of hyper-parameters [Bergstra et al. 2011]; if these hyper-parameters tend to interact weakly or not at all, they could be optimized independently. This would also have substantial ramifications for many existing Gaussian-process-based methods, for which a primary bottleneck is fitting a Gaussian-process model to high-dimensional landscapes [Kandasamy et al. 2015]. This yields our second research question: **RQ 2. Do most hyper-parameters interact strongly; if not, to what extent do they interact and where?**

We are by no means the first to study the structure of optimization problems by way of an analogy to physical landscapes. The analysis of the landscapes induced by optimization problems is a well established topic, that can be traced back to the seminal work by Wright [1932] on the study of evolutionary biology. Since then, a vast number of techniques have been proposed for what is typically referred to as *fitness landscape analysis* (see, *e.g.*, Hoos and Stützle [2005]; Malan [2021]; Malan and Engelbrecht [2013]; Pitzer and Affenzeller [2012] or Watson [2010]).

The application of fitness landscape analysis has recently received considerable attention in machine learning. For example, to study the properties of the error landscapes induced by the weight parameters of neural networks (see *e.g.*, Rakitianskaia et al. [2016] or van Aardt et al. [2017]). However, this work is orthogonal to ours; instead, for neural networks we analyze the landscapes induced by architectural parameters and hyper-parameters.

Recently, neural architecture search landscapes have also been studied with fitness landscape analysis techniques. For example, Rodrigues et al. [2020] studied the landscapes of a neuroevolution procedure for optimizing the performance of convolutional neural networks (CNNs). In particular, they compared the performance responses induced by three different types of neuroevolution mutation operations. The first modifies the hyper-parameters of individual neurons, *e.g.*, the choice of the activation function. The second modifies the hyper-parameters of the optimization procedure

used to train the CNNs, *e.g.*, the learning rate. Finally, the third modifies the topology of the network itself. From their analysis, they concluded that it is easiest to configure the hyper-parameters of the optimization procedure, because they yield the smoothest responses, and hardest to configure the topology-related hyper-parameters, because they yield the most rugged responses.

Nunes et al. [2021] applied fitness landscape analysis methods to the validation accuracy obtained by various graph neural network (GNN) architectures. They estimate the validation accuracy using a single run for a given configuration with a given train/validation split. Unlike Rodrigues et al. [2020], they do not consider the hyper-parameters of the optimization procedure in the landscape, instead they focus only on parameters that relate specifically to the architecture of the GNN, *e.g.*, the activation functions used. They conclude from a fitness distance correlation (FDC) analysis [Jones and Forrest 1995] that the landscapes that arise for neural architecture search on three different datasets should be relatively simple to optimize, with most high-quality configurations grouped together in the search space. They also speculate from their results that the landscapes do not contain a substantial amount of neutrality [Reidys and Stadler 2001], but call for further study on the matter.

Clearly, the landscape analysis of neural architecture search is closely related to that on developing AutoML pipelines; however, NAS provides the unique additional challenge that neural networks have architectures that are graphical in nature, thereby imposing additional constraints on the parameterization of the target algorithms. Consequently, the landscapes that arise in NAS scenarios may (or may not) share similar properties with those of traditional AutoML pipelines.

We are only aware of two recent papers that specifically address AutoML loss landscapes. In the first, Garciarena et al. [2018] make several dubious claims, for example: That AutoML methods typically optimize the *training loss* of a model, which is trivially false (see *e.g.*, Bergstra et al. [2011]; Eggensperger et al. [2013] or Falkner et al. [2018]). They also claim that multiple distinct local optima exist in the landscapes they studied and substantiate this claim by the fact that multiple distinct machine learning models each contain hyper-parameter configurations that yield similarly-good solutions. However, given the neighbourhood operator *that they define*, each of these "local" minima may be directly connected.

Independently and in parallel to a major part of our work, Pimenta et al. [2020] apply fitness distance correlation and a measure of neutrality to the landscapes induced by modifying AutoML pipelines. They concluded that AutoML pipeline landscapes tend to be flatter close to high-quality solutions and that FDC is a poor metric for characterizing AutoML loss landscapes, necessitating further study. We observed similar results for FDC (see Section 4) and, partly prompted by this, introduced two novel methods for analyzing the global shape of the landscapes (see Section 2.2).

We note that both of these existing studies are somewhat orthogonal to ours, as they focus on the landscapes of AutoML pipelines, whereas we focus on the landscapes induced by hyper-parameters of pre-specified models. This is a more interesting first question, as each pipeline must contain several hyper-parameters that need instantiating, and therefore, our work is likely to apply to automated pipeline generation procedures as well. However, all of our methods (see Section 2) could be easily adapted to study the landscapes of AutoML pipeline generation through the appropriate definition of a neighbourhood graph. We leave this as future work.

More closely related to our current work on AutoML loss landscapes, was our recent analysis of algorithm configuration landscapes [Pushak and Hoos 2018]. We proposed methods to test for uni-modal and convex responses of parameters; however, our original work was limited to studying the effects of individual parameters while all other parameters are fixed. In contrast, in this work we propose novel improvements to these methods for analyzing the uni-modality of the *full* AutoML loss landscapes and the joint convexity of multiple numerical hyper-parameters. Nevertheless, one major advantage of analyzing lower-dimensional slices of a landscape is the ability to more

easily visualize and interpret the results. Therefore, we apply our methods to study the structure present in the full landscapes as well as in one- and two-dimensional slices. This also enables direct comparisons against our earlier results, revealing several similarities and some differences between the structures encountered in AutoML loss landscapes and running time minimization landscapes.

We also introduce a simplistic optimization procedure that naïvely assumes all hyper-parameters can be optimized independently a single time in a random sequence. We stress that the goal in considering this method is not to introduce a novel, state-of-the-art hyper-parameter optimization procedure. Instead, this optimization procedure should be viewed as a *landscape analysis tool* that explores the extent to which hyper-parameter interactions must be taken into account by a state-of-the-art optimizer that exploits, *e.g.*, the parallel evaluation of hyper-parameter configurations and multi-fidelity estimates of configuration performance.

Many methods exist to quantify the importance of hyper-parameters and, in some cases, their interactions. Most of these methods are local, see *e.g.*, forward selection [Hutter et al. 2013], ablation analysis [Biedenkapp et al. 2017; Fawcett and Hoos 2016] and local parameter importance [Biedenkapp et al. 2018]. While useful, local methods cannot determine if hyper-parameters can safely be optimized independently. Functional ANOVA [Hutter et al. 2014a] is the most relevant global importance technique; however, it is particularly sensitive to hyper-parameters for which a small fraction of their values yield extremely bad performance (see Section 4). Furthermore, while it returns technically sound results, they are nevertheless un-intuitive, which can obscure a lay-person's understanding when hyper-parameters interact strongly. To address this gap, we introduce a method to assess the significance of hyper-parameter partial derivatives (see Section 2.7).

Somewhat related to our work is the recent discovery of the so-called "double descent curve" that appears for many neural networks (see *e.g.*, Belkin et al. [2019] and references therein), which shows empirical evidence that there are two modes in the AutoML loss landscapes of neural networks. As model complexity increases, you pass from an "under-parameterized" regime to an "over-parameterized" regime – that is, in the under-parameterized regime there is insufficient flexibility in the model to perfectly memorize the training data, hence modifying model complexity corresponds to trading off between under- and over-fitting. However, somewhat surprisingly, it appears that a second mode of even-better models exists in the over-parameterized regime. Belkin et al. [2019] speculate that this is an example of Occam's razor, *i.e.*, that a large number of *smooth* models exist in the over-parameterized regime that can perfectly learn the training data and generalize well to unseen data. They show evidence of double descent curves for random forests and xgboost, and hypothesize that this phenomenon is ubiquitous among all sufficiently expressive models. Our results do not contradict theirs; most (if not all) of the landscapes we study (see Section 3) are likely restricted to being entirely within one of the two regimes. Furthermore, Belkin et al. [2019] conjecture that the peak between the two modes is very narrow and may be easily missed by discretization of hyper-parameters (as must be done for empirical landscape analysis).

We demonstrate that FDC [Jones and Forrest 1995] is poorly suited for characterizing the difficulty of AutoML loss landscapes; however, our novel methods reveal that the full AutoML loss landscapes appear to be uni-modal or very close to uni-modal (see Section 4.3). We also show that, even though none of the full landscapes we study are convex (see Section 4.3), most of the hyper-parameters studied yielded response slices that appear convex (see Section 4.1) when the other hyper-parameters are fixed to their optimal values. Furthermore, we show that, while a few hyper-parameters interact strongly, most do not, particularly in the vicinity of high-quality configurations, which may in part explain why our simplistic optimization procedure turns out to be highly effective. We present the (novel and existing) methods we used for this analysis (see Section 2) and summarize the machine learning methods that give rise to the landscapes we study

(see Section 3). Finally, we speculate about the intuitive shape of AutoML loss landscapes in light of our results and comment on future work (see Section 5).

2 LANDSCAPE ANALYSIS METHODS

The properties of any search landscape depend upon the definition of the underlying neighbourhood – in our case, on the relation that specifies which hyper-parameter configurations are neighbours and on the metric used to quantify distance between configurations. All of the scenarios we study (see Section 3) contain pre-evaluated grids of hyper-parameter configurations with a mix of numerical and categorical hyper-parameters. We define all categorical values of a hyper-parameter to be distance 1 from each other. For a numerical hyper-parameter, we define the distance between two values to be the number of steps between them on the grid used for discretization. We then define the distance between two configurations c_a and c_b to be the sum of the distances between the respective pairs of hyper-parameter values. We define a graph $G = (C, E)$, where the vertices C are the hyper-parameter configurations and edge $e = (c_a, c_b)$ is in E if and only if c_a and c_b are distance 1 from each other. Many of the landscape analysis methods we use require a confidence interval $[\mathcal{L}_{\text{low}}(c), \mathcal{L}_{\text{up}}(c)]$ for each $c \in C$ that captures the best-known estimate for the performance of configuration c measured in terms of some validation loss, $\mathcal{L}$.

2.1 Hyper-Parameter Response Slices

In earlier work, we formally defined one-dimensional *parameter response slices* [Pushak and Hoos 2018], which we generalize here for higher dimensions. Let A be a learning algorithm for a given model that has n hyper-parameters $P = \{p_1, p_2, ..., p_n\}$. We define a K-dimensional *hyper-parameter response slice* for hyper-parameters $P_s = \{p_a, p_b, ..., p_k\} \subseteq P$ to be obtained by fixing all other parameters $P \backslash P_s$ to their respective values in some configuration $c \in C$ and measuring the performance, $\mathcal{L}$, of A as a function of the hyper-parameters in P_s; formally,

$$r(v_a, v_b, ..., v_k) = \mathcal{L}(c|_{p_a=v_a, p_b=v_b, ..., p_k=v_k}), \tag{1}$$

where $c|_{p=v}$ denotes configuration c with hyper-parameter p modified to v. Intuitively, a one-dimensional hyper-parameter response slice corresponds to a single, axis-aligned slice through the configuration landscape of A. Technically, it can be seen as a conditional response, subject to all other hyper-parameters being held to fixed values. Even though all of the following methods can be applied to full, n-dimensional AutoML loss landscapes, we still study lower-dimensional hyper-parameter response slices of the landscapes, as they can be more easily visualized, and thus the corresponding results can be more easily interpreted.

2.2 Test for Uni-Modality

The test for uni-modality attempts to construct a piece-wise affine landscape that is both uni-modal and contained within the confidence intervals. If no such landscape exists, it rejects uni-modality. To do this, we define an augmented graph $G' = (C', E')$, where $(c, \mathcal{L}) \in C'$ if and only if $c \in C$ and $\mathcal{L}_{\text{low}}(c) \leq \mathcal{L} \leq \mathcal{L}_{\text{up}}(c)$; and where a directional edge $e' = ((c_a, \mathcal{L}_a), (c_b, \mathcal{L}_b))$ is in E' if and only if $e = (c_a, c_b) \in E$ and $\mathcal{L}_a \leq \mathcal{L}_b$. The method begins by finding a vertex $(c^*, \mathcal{L}_{\text{up}}(c^*))$ such that $\mathcal{L}_{\text{up}}(c^*) \leq \mathcal{L}_{\text{up}}(c)$ for all $c \in C$. We define c_k to be *reachable from* c^*, if and only if there exists a *certifying path* $p = (c^*, \mathcal{L}^*), (c_1, \mathcal{L}_1), ..., (c_k, \mathcal{L}_k)$ in G' – *i.e.*, c_k can be reached via a path p from c^* with a sequence of non-decreasing objective function values that stay within the confidence

intervals for each c on p. Technically, a certifying path p is a chain of tuples $(c_i, \mathcal{L}_i)$ such that

$$\begin{aligned} &\mathcal{L}_i \leq \mathcal{L}_{i+1}, \\ &distance(c_i, c_{i+1}) = 1, \\ &\mathcal{L}_i \in [\mathcal{L}_{\text{low}}(c_i), \mathcal{L}_{\text{up}}(c_i)] \text{ and} \\ &c_0 = c^*. \end{aligned} \tag{2}$$

Clearly, if each $c \in C$ is reachable from c^*, we cannot reject uni-modality for the landscape that induced G'. In Theorem 1 we show that if there exists some $c_0 \in C$ that is not reachable from c^* then no piece-wise affine, uni-modal landscape exists within the confidence intervals of C, hence we can safely reject uni-modality for the landscape that induced G'.

Theorem 1 (Correctness of Test for Uni-Modality). *Let $G' = (V', E')$ be a neighbourhood relation graph defined for an AutoML loss landscape that contains a set of pre-evaluated configurations C, such that each configuration $c \in C$ has a corresponding confidence interval $[\mathcal{L}_{low}(c), \mathcal{L}_{up}(c)]$ for the loss of the machine learning method. If $\mathcal{L}_{up}(c^*) \leq \mathcal{L}_{up}(c)$ for all $c \in C$, and there exists $c_0 \in C$ that is not reachable from c^*, then no uni-modal, piece-wise affine function exists that is contained within the confidence intervals, and hence uni-modality can be rejected for the landscape.*

A proof of Theorem 1 is in Appendix A.

We test if each configuration is reachable from c^* by running Dijkstra's algorithm on the modified graph G' starting at the vertex $(c^*, \mathcal{L}_{\text{low}}(c^*))$. Clearly, if c_k is reachable from c^*, Dijkstra's algorithm will find a certifying path from $(c^*, \mathcal{L}_{\text{low}}(c^*))$ to $(c_k, \mathcal{L}_k)$. Furthermore, Dijsktra's algorithm finds the shortest path to each vertex and visits them in order. Hence, for each c, the first vertex $(c, \mathcal{L})$ visited by Dijkstra's algorithm must contain the smallest value $\mathcal{L}$ out of all vertices $(c, \mathcal{L}')$ for which a path exists from $(c^*, \mathcal{L}_{\text{low}}(c^*))$. Trivially, if an edge $e' = ((c, \mathcal{L}'), \cdot)$ is in E' and $\mathcal{L}_{\text{low}}(c) \leq \mathcal{L} < \mathcal{L}'$, then $e = ((c, \mathcal{L}), \cdot)$ must also be in E'. We can therefore keep the time complexity of the test log-linear, by pruning all paths to vertices $(c, \mathcal{L}')$ such that $\mathcal{L}' > \mathcal{L}$.

2.3 Test for Convexity

Convexity does not apply to categorical hyper-parameters, so we fix these to their respective values in the optimal configuration prior to applying the test. Convexity is also not defined for discrete, numeric hyper-parameters. However, we can still include them in our test by checking if the piece-wise affine function that linearly interpolates between their discrete values is convex. The test converts the n numeric hyper-parameters of each configuration into a set of points with $n + 1$ dimensions, where the last dimension is the upper bound of the performance metric confidence intervals. The lower half of the convex hull of these upper bounds then corresponds to a convex, piece-wise linear function that has the largest possible performance value for all configurations, without exceeding any of the upper bounds. If this function is contained within all of the confidence intervals for the configurations, then there exists at least one convex function within the confidence intervals, and thus we cannot reject the hypothesis of the landscape being convex. We say that a configuration is *interior* to the convex hull if the lower bound of its confidence interval is contained within the convex hull. If any of the configurations are interior, then our test rejects convexity.

Determining whether or not the hypothesis of convexity should be rejected therefore amounts to checking whether any of the lower bounds for a given configuration are contained within the convex hull of the upper bounds. Let $X \in \mathbb{R}^{m \times n+1}$ be a matrix containing all of the configurations of the landscape for which the first n columns correspond to the hyper-parameter values, and the $n+1^{\text{th}}$ column represents the corresponding upper bounds. Let $x \in \mathbb{R}^{n+1}$ be an $n+1$-dimensional point that corresponds to a particular configuration and the corresponding lower bound. A computationally

efficient way to determine whether or not x is in the convex hull of X is to check if x can be represented as a convex combination of X – that is, if there exists $\alpha \in \mathbb{R}^m$ such that

$$\sum_{i=1}^{m} \alpha_i \cdot X_i = x, \tag{3}$$

where

$$\sum_{i=1}^{m} \alpha_i = 1 \text{ and } \alpha_i \geq 0 \text{ for all } \alpha_i. \tag{4}$$

One edge case that needs to be considered when applying this method is how lower bounds that are co-planar with the convex hull of the upper bounds are treated. Technically, if a single lower bound is co-planar with the convex hull of the upper bounds, then there exists only a single value that a convex function contained within the confidence intervals can take on for the corresponding configuration, and therefore convexity should not be ruled out. However, testing whether or not x is a convex combination of X does not allow us to easily distinguish between whether x is co-planar to the convex hull of X or strictly interior to it. This problem is further complicated by the numerical imprecision of floating point operations, which may introduce possible rounding errors. Therefore, for any configuration and lower bound x that is found to be a convex combination of X, we propose to perform a second test with x', where x' contains the same set of hyper-parameter values as x, but where its lower bound has been slightly decreased by a small absolute and relative tolerance. If x' can also be represented as a convex combination of X, then we say that the lower bound for the configuration is interior and reject the hypothesis of convexity, otherwise we say that it is co-planar within machine precision. In our experiments, we used a relative and absolute tolerance of 10^{-5} and 10^{-8}, respectively.

2.4 Validation of Sensitivity

If the confidence intervals are all very large or similar (*e.g.*, the intersection of every confidence interval is non-empty), then our tests will be trivially unable to reject their null hypotheses due to a lack of sensitivity. The situation is further complicated in high dimensions by the connectedness of the graph (*e.g.*, a fully connected graph is uni-modal for any set of confidence intervals). We would not expect a landscape with completely random performance values to be either uni-modal or convex. We therefore randomly permute the confidence intervals for each landscape, thereby breaking the association between the original configurations and performance estimates, but maintaining the distribution of confidence intervals and the neighbourhood graph. If a test for convexity or uni-modality fails to reject their null hypotheses for a permuted landscape, then a similar result on the original landscape should be considered meaningless. We would not expect any random landscapes to be uni-modal or convex. However, to avoid falsely assuming sensitivity, in our experiments, we heuristically repeated this procedure three times for each landscape. We then considered the test "sensitive", only if convexity or uni-modality was rejected for all three of the permuted landscapes.

2.5 Identifying "Interesting" Hyper-Parameters

Hyper-parameters with almost flat responses (*i.e.*, robust ones, whose settings have little or no effect on the performance of the algorithm) are of limited interest to our investigation. We therefore used the same simple heuristic procedure from our previous work [Pushak and Hoos 2018] to identify hyper-parameters with interesting (*i.e.*, non-flat or sensitive) one-dimensional response slices, based on the sizes of and overlap between the confidence intervals for each value of the hyper-parameter. Intuitively, this measure is somewhat related to how we validate the sensitivity

of the statistical tests as described in Section 2.4, since if the response of a hyper-parameter is flat, then we should not expect the permuted landscape to be non-flat. To be precise, we say a hyper-parameter's response slice is *interesting*, if the size of the overlap between the two confidence intervals with the least amount of overlap is at most half of the average size of the confidence intervals.

2.6 Fitness Distance Correlation

Fitness-distance correlation (FDC) [Jones and Forrest 1995] measures the degree to which the "fittness" of an objective function is correlated with the distance from an optimal configuration. Technically, let $d(c)$ be the distance of configuration c from the nearest optimal solution c^*. Then, given the fitness-distance pairs $(\mathcal{L}(c), d(c))$ for all candidate configurations $c \in C' \subseteq C$, FDC is defined as

$$\rho_{FDC}(\mathcal{L}, d) := \frac{Cov(\mathcal{L}, d)}{\sigma(\mathcal{L}) \cdot \sigma(d)}, \tag{5}$$

where Cov and σ represent the covariance and standard deviations over all fitness-distance pairs, respectively.

High FDC corresponds to globally funnel-shaped landscapes that are easy to optimize and low FDC corresponds to random or deceptive landscapes. FDC has since been prominently applied to study many problems (see *e.g.*, Hoos and Stützle [2005] or Pimenta et al. [2020]).

2.7 Locally Significant Partial Derivatives

For each combination of hyper-parameters, we calculate finite difference approximations for the partial derivatives of the performance measure with respect to the hyper-parameters. We use the grid of hyper-parameter configurations to obtain multiple local estimates for each partial derivative. For scenarios with k estimates for the performance measure, we calculate k paired estimates for each local partial derivative. Then, for each one, we check to see if the partial derivative is significantly different from zero. Finally, we count the percentage of local partial derivatives that are different from zero at a 5% significance level.

Counter-intuitively, this method can even be generalized for use with categorical hyper-parameters. Let $\mathcal{L}_k(c|_{p=v_l})$ be the estimate for the performance measure of a response slice evaluated at the l^{th} value v_l of hyper-parameter p on the k^{th} validation set. We calculate the l^{th} local partial derivative for the k^{th} validation set as

$$\frac{\partial \mathcal{L}_k}{\partial p} = \frac{\mathcal{L}_k(c|_{p=v_l}) - \mathcal{L}_k(c|_{p=v_{l-1}})}{v_l - v_{l-1}}. \tag{6}$$

However, since the distance $v_l - v_{l-1}$ is a constant for each value of k in $\frac{\partial \mathcal{L}_k}{\partial p}$, it can be ignored without changing the outcome of the statistical test. Therefore, we can generalize our notion of local hyper-parameter significance and apply it to each pair of values for a categorical hyper-parameter by simply omitting the distance normalization term.

In some scenarios, only a single point estimate for the performance measurement was available; however, confidence intervals for the performance could still be obtained by making additional assumptions, *e.g.*, we assume the binary classification test errors for the latent structured SVM scenario (see Section 3) are binomially distributed. In these cases, we applied a pessimistic version of the test that operates directly on the confidence intervals. In particular, to calculate the upper bound of a derivative for two points v_l and v_{l-1}, we take the difference between the upper bound of v_l and the lower bound of v_{l-1}, *i.e.*,

$$\frac{\partial \mathcal{L}_{\text{up}}}{\partial p} = \mathcal{L}_{\text{up}}(c|_{p=v_l}) - \mathcal{L}_{\text{low}}(c|_{p=v_{l-1}}). \tag{7}$$

Similarly, to calculate a lower bound, we use

$$\frac{\partial \mathcal{L}_{\text{low}}}{\partial p} = \mathcal{L}_{\text{low}}(c|_{p=v_l}) - \mathcal{L}_{\text{up}}(c|_{p=v_{l-1}}). \tag{8}$$

We then say that the derivative is significantly different from zero, if the confidence interval does not contain zero. For first-order derivatives, this is equivalent to checking if two neighbouring configurations have overlapping confidence intervals for their performance measurements.

2.8 fANOVA

Functional analysis of variance (fANOVA) has been prominently applied to study the importance of hyper-parameters and their interactions [Hutter et al. 2014a]. It decomposes the variance observed in the performance into components associated with each hyper-parameter individually and with their interactions. However, instead of using the implementation by Hutter et al. [2014a], which operates on a landscape approximated using a random forest, we re-implemented the analysis to compute the exact results for a grid of hyper-parameter configurations. Since the complexity of fANOVA grows exponentially with the order of the interaction effects computed, previous work has been restricted to studying the importance of low-order interaction effects (*e.g.* 2^{nd}- or 3^{rd}-order effects) [Hutter et al. 2014a; Klein and Hutter 2019]. However, our re-implementation came with an unexpected bonus: we made heavy use of highly-optimized libraries (*i.e.*, `numpy`), which allowed us to compute the importance of the 11^{th} order hyper-parameter interactions for our largest scenario within a few minutes. While purely due to software engineering, this is a substantial improvement, since calculating only low-order importance scores for landscapes with strong interactions could fail to reveal the important hyper-parameters due to marginalization: *e.g.*, the function defined by $f(x, y) = |x - y|$ for $x, y \in [-1, 1]$ would be attributed with no importance to the first order effects and 100% importance to the interaction effect.

2.9 Optimizing Hyper-Parameters Independently

Locally significant partial derivatives and fANOVA both quantify the degree to which the interactions of two or more hyper-parameters impact the response in terms of loss. The first speaks to the percentage of the landscape for which the impact of interactions are statistically significant, and the second speaks to the magnitude of those interactions. However, a much more natural and practical question to ask is: Can I optimize each of the hyper-parameters of my algorithm independently and still obtain a high-quality loss value?

In practice, this is quite likely how some algorithm designers choose the default values of each of the hyper-parameters of their algorithms, *i.e.*, through manual exploration of the response of varying each hyper-parameter independently, in sequence. After selecting a good value for a hyper-parameter, it would then typically be held fixed for the remainder of the manual optimization process.

We therefore propose to use the grids of the pre-evaluated hyper-parameter configurations to emulate a simplistic optimization procedure, for which we can quantify how frequently it can be expected to produce high-quality configurations. In particular, the procedure assumes that the default configuration (*i.e.*, the initial incumbent) is the one in the grid that is estimated to obtain the worst loss. Then, for a random ordering of the hyper-parameters, the simplistic optimization procedure looks at the point-estimates for the loss of all of the configurations in the one-dimensional slice centered around the current incumbent and updates the incumbent to be the one with the best loss. This process is carried out for each of the hyper-parameters, until each of them has been optimized once. At that point, we say that its final incumbent configuration is 'tied with optimal', if the lower bound for its loss is at least as good as the upper bound for the loss

Table 1. The machine learning scenarios for which AutoML loss landscape analysis was performed.

Model	Dataset	# HP		# Instances			# Loss Samples	
		Num	Cat	Train	Val	Test	Val	Test
FCNet	SL	6	3	32K	11K	11K	4	4
	PS	6	3	27K	9K	9K	4	4
	NP	6	3	7K	2K	2K	4	4
	PT	6	3	4K	2K	2K	4	4
XGBoost	CT	11	–	47K	5K	–	5	–
LogReg	MNIST	4	–	471K	52K	10K	1	1
LSSVM	UP	3	–	20K	–	20K	–	1
OLDA	Wiki	3	–	200K	25K	25K	–	1

of the configuration with the best point-estimate for the loss. For a given grid of pre-evaluated hyper-parameter configurations, this procedure can be run for all possible permutations of the hyper-parameters (which determine the sequence in which these are optimized), in order to obtain the probability that the simplistic optimization procedure will obtain a final incumbent that is tied with optimal.

Notice that we do not expect this simplistic optimization procedure to be competitive with state-of-the-art hyper-parameter optimizers. Instead, our goal is to use it as a tool for exploring the extent to which hyper-parameters can be naïvely optimized independently, and thus the extent to which a state-of-the-art hyper-parameter optimizer (which can also make use of parallel resources and multi-fidelity estimates of the loss) can rely upon an assumption of independence between its hyper-parameters.

3 EXPERIMENTAL SETUP

We applied the methods in Section 2 to landscapes from eight machine learning scenarios summarized in Table 1. Ideally, each model should be trained and evaluated on separate, randomly sampled training and validation sets (*e.g.*, using k-fold cross validation), as this allows for generalization loss confidence intervals that capture three sources of variance: independent training runs, and the instances included in the particular training and validation datasets. However, we found several existing scenarios with grids of pre-evaluated hyper-parameter configurations that provided sufficient data to calculate confidence intervals that captured subsets of these sources of variance.

Klein and Hutter [2019] pre-evaluated a grid of 62 208 joint hyper-parameter and neural architecture configurations for a feed-forward neural network (FCNet)[1] applied to four different UCI datasets [Dua and Graff 2017]: slice localization (SL) [Graf et al. 2011], protein structure (PS) [Rana 2013], naval propulsion (NP) [Coraddu et al. 2016] and Parkinsons telemonitoring (PT) [Tsanas et al. 2010]. They performed 4 training runs for each configuration using different random seeds, yielding a total of 4 validation and 4 test loss scores (for the final, 100-epoch training budget that we analyze here). To obtain the best-possible estimate for generalization error, we took the mean of all 8 loss samples and calculated 95% Student-T-based confidence intervals by assuming that all 8 loss samples were independently and identically distributed. When looking at the validation and test scores separately, 11.92% of the configurations between all of the scenarios have normality rejected for their losses at a 5% significance level using a Shapiro-Wilk test[Shapiro and Wilk 1965]. However, by naïvely combining the validation and test scores in this way, these samples are not in fact independently and identically distributed, and hence 32.89% of the sets of samples have normality rejected. This indicates that these confidence intervals are likely slightly over-confident,

[1] Available at https://github.com/automl/nas_benchmarks.

and thus some of the statistical tests that we perform may incorrectly reject their null hypothesis more frequently than expected for the given significance level.

Our second scenario comes from ACLib [Hutter et al. 2014b], where Xgboost [Chen and Guestrin 2016] is applied to the covertype dataset [Dua and Graff 2017]. We applied fANOVA [Hutter et al. 2014a] to a random sample of configurations and found that `eta` was by far the most important hyper-parameter. We therefore evaluated a uniform grid of hyper-parameter values with 7 values for `eta` and 3 values for each of the remaining 10 hyper-parameters (see Appendix B for the table of values).

ACLib provides 10 splits for performing 10-fold cross validation. To keep within our computational budget, we trained and evaluated the model on the first 5 splits. ACLib also specifies a 1 000 second running time cut-off for training. In the scripts provided by ACLib, this time includes reading the data from disk and splitting it. However, since this required about 40 seconds per run, and we performed just over 2 million runs of Xgboost, we reduced this cutoff to 960 seconds and used our own scripts that pre-loaded the instances and held them in memory. Nevertheless, collecting all of the performance data for xgboost took 14.7 CPU years. We calculated 95% confidence intervals for the generalization loss of Xgboost using Student-T-based confidence intervals (normality was rejected for only 5.37% of the samples at a 5% significance level).

Our remaining three scenarios came from pre-computed grids available in HPOLib [Eggensperger et al. 2013]. The first scenario [Snoek et al. 2012], logistic regression on the MNIST digits dataset [LeCun et al. 1998], contained one validation and one test loss, so we again calculated Student-T-based confidence intervals[2]. The hyper-parameter configuration grids for the latent structured SVM [Miller et al. 2012] applied to a DNA motif-finding dataset (UniProbe or UP) and the online LDA model [Hoffman et al. 2010] applied to Wikipedia articles were evaluated by Snoek et al. [2012]. However, each only had a single estimate of generalization loss available. Since the SVM scenario was binary classification, we assumed that the errors were binomially distributed[3] and calculated 95% confidence intervals using the Wald method [de Laplace 1820]. We were unable to make a similar assumption for the perplexity loss from online LDA. We therefore assumed all confidence intervals could be expressed as a percentage of the loss and performed a binary search to find the smallest value for the size of the interval for which the tests failed to reject their null hypotheses (uni-modality or convexity). If the confidence intervals are small, we can still conclude that the landscape is close to uni-modal or convex.

4 RESULTS

We applied each applicable method of analysis from Section 2 to each of the 45 one-dimensional hyper-parameter response slices (see Section 4.1), each of the 211 two-dimensional hyper-parameter responses slices (see Section 4.2) as well as the full landscapes (see Section 4.3).

4.1 One-Dimensional Hyper-Parameter Response Slices

The results from our tests for uni-modality and convexity, as well as the median FDC for the 45 one-dimensional, numeric hyper-parameter response slices are summarized in Table 2. Confidence intervals of size ± 0.14% were large enough for all of the response slices for the hyper-parameters

[2]With only two samples per configuration we were unable to test for deviations from normality.

[3]Technically, the errors are distributed according to the convolution of two binomial distributions – one for each class; however, without the error rates for each class it is impossible to obtain more precise confidence intervals. Using this method yields conservatively large confidence intervals [Emil and Tamar 2013], thus our test may have failed to detect slight deviations from uni-modality; however, our permutation tests still indicated that the results should be considered sensitive (see Tables 3, and 4), hence it is unlikely that we missed observing any large deviations from uni-modality.

Table 2. The percentage of the one-dimensional hyper-parameter response slices for which uni-modality (Uni-M) and convexity (Cvx) could not be rejected, and the median fitness distance correlation (FDC). All hyper-parameter response slices are centered around the global optima of the landscapes. Note that this table assumes the online LDA scenario has intervals of size ± 0.14%.

Type	# Slices	Uni-M	Cvx	FDC
All	45	97.8%	82.2%	0.93
Interesting	44	97.7%	81.8%	0.93

of online LDA to appear both uni-modal and convex; since these are very small, we count them as being both uni-modal and convex, and we show them as such in the table.

Overall, the results are quite similar to running time minimization landscapes induced by SAT, TSP and MIP algorithms [Pushak and Hoos 2018]: Nearly all of the hyper-parameter response slices appear uni-modal (44 out of 45) and most of them appear convex (37 out of 45). The biggest difference is that all but one of the 45 hyper-parameter response slices are considered to be interesting, according to our heuristic criterion (compared to just 18 out of the 193 of the running time minimization slices). We believe this is caused by AutoML loss landscapes typically being much less noisy than those for algorithm configuration for running time minimization.[4]

Xgboost's `subsample` hyper-parameter was the only one for which the response slice was determined to be neither unimodal nor convex. Since this hyper-parameter was found to be relatively unimportant, according to fANOVA (its first order effect accounts for 2.5% of the variance in the loss), we originally only evaluated three different values for it. The middle hyper-parameter value had a loss substantially larger than both of the others, as well as a much wider confidence interval.

We investigated by increasing the number of hyper-parameter values from 3 to 21 for this individual hyper-parameter response slice (and we re-evaluated the original three values as well). We show the resulting hyper-parameter response slice in the top left pane of Figure 1. As can be seen, the large barrier in the response completely disappeared and our tests now failed to reject both uni-modality and convexity. Looking more closely at the original data, we see that 4 out of the 5 evaluations on different cross-validation folds exceeded the 960 second running time cutoff; therefore, their error rates were recorded as 1. For the expanded response slice, the mean running time for `subsample` = 0.50 was 863 seconds, with a standard deviation of 42. Therefore, we surmise that background processes or other environment noise likely caused several of the original runs to be censored.

Given that this is, effectively, a spurious result, it would be reasonable to replace the original loss values with the new ones for the remaining analysis. However, in practice, AutoML optimizers are likely to encounter similar challenges from time to time, therefore, we leave the data as is and continue to analyze the landscape with a spike in it.

[4]In fact, the difference in the variability of the performance in the two applications makes sense. In AutoML scenarios, each problem instance corresponds to a random training and validation split of a given data set. For most data sets, this will likely result in problem instances that are relatively similar. In contrast, for running time minimization scenarios the diversity in the problem instances is likely much larger. For example, consider the bounded model checking SAT instance sets, for which each problem instance corresponds to unrolling different (possibly unrelated) hardware circuits to different depths, thereby creating a much more heterogeneous collection of problem instances. Furthermore, running time is an inherently noisy metric, since even repeated calls to a deterministic algorithm for solving the same problem instance will most often result in different running times due to concurrently-running background processes on the machine. Conversely, some of the performance metrics in the AutoML loss landscapes we study (*e.g.*, the error rate of the latent-structured SVM) are bounded to the interval [0, 1], which further restricts the maximum size of the confidence intervals.

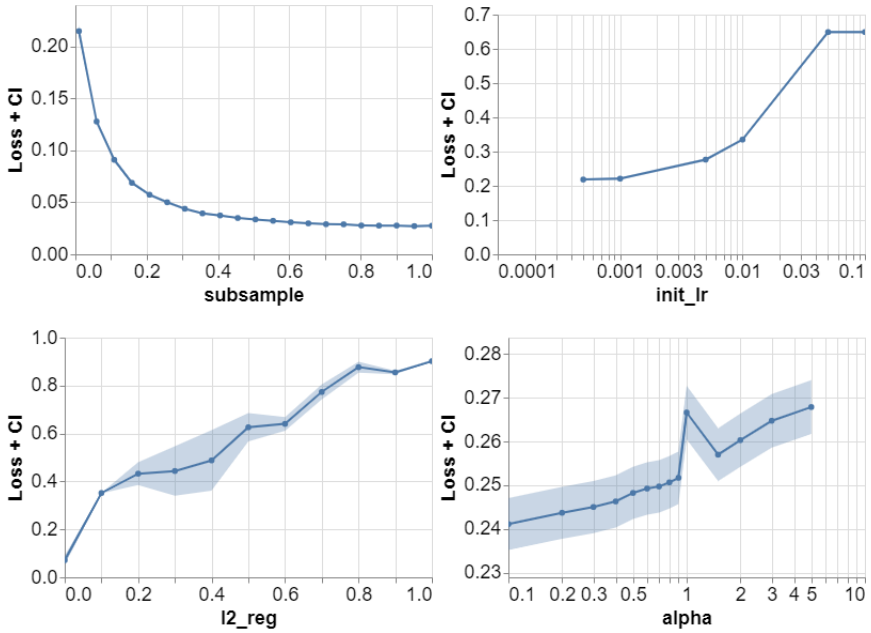

Fig. 1. Four example one-dimensional hyper-parameter response slices. From top to bottom, left to right: Xgboost's `subsample` hyper-parameter, FCNet's `init_lr` hyper-parameter on the protein structure dataset, Logistic Regression's `l2_reg` hyper-parameter, and LSSVM's `alpha` hyper-parameter.

Of the remaining seven hyper-parameters for which convexity was rejected, three were FCNet's `init_lr` hyper-parameter for three of the four datasets. For all four datasets, small values of the hyper-parameter provided the best loss, with a sometimes-abrupt transition to a sub-optimal, approximately plateau-shaped region. The response on the protein structure dataset (see the top right pane of Figure 1) is representative of the other responses; however, the response on the slice localization dataset has confidence intervals that are too large to reject convexity.

The `epsilon` hyper-parameter for latent-structured SVMs also yielded a somewhat similar response slice; however, the difference between the good and bad values for the hyper-parameter were substantially smaller, there were only 4 values evaluated for the hyper-parameter, and the size of the confidence intervals were nearly as large as the transition, thus it is unclear whether or not this was merely due to random fluctuations in the performance measurements.

At the bottom left of Figure 1, we show the response slice for logistic regression's `l2_reg` hyper-parameter, for which convexity was also rejected. The mean loss increases nearly monotonically with the value of the hyper-parameter from 0.07 to 0.90; however, the response is more rugged than for most of the other hyper-parameters, and appears to perhaps even be concave in shape. The only other hyper-parameter that shows some ruggedness in its one-dimensional response is latent-structured SVM's `alpha` hyper-parameter, which is shown in the bottom right pane of Figure 1. In this case, the response is very smooth, apart from a single hyper-parameter value that yields an abnormally large loss. Visually, this hyper-parameter response slice at first appears bi-modal; however, careful inspection reveals that the confidence intervals are just large enough to admit the possibility of a uni-modal response. Nevertheless, in light of the spurious bi-modal response that we observed for Xgboost's `subsample` hyper-parameter, it is possible that the behaviour present

Table 3. Summary of the test results for uni-modality and convexity, as well as the median FDC applied to the two-dimensional hyper-parameter response slices centred on the global optima of the AutoML loss landscapes. We also show the mean percentage of the landscape that was unreachable (UnR) from the global optimum or interior (Int) to the convex hull of the upper bounds for those slices for which uni-modality or convexity could be rejected, respectively; the mean percentage of the lower-bounds that were co-planar (Co-P) to the convex hull of the upper bounds; and the mean percentage of the landscapes for which the tests appeared sensitive (Sen) according to our permutation-based analysis.

Type	# Slices	Uni-M	UnR	Sen	Cvx	Int	Co-P	Sen	FDC
N×N	127	92.1%	22.3%	77.2%	58.3%	24.8%	22.2%	90.6%	0.71
N×C	72	100.0%	–	62.5%	–	–	–	–	0.77
C×C	12	100.0%	–	0.0%	–	–	–	–	0.72

in this response may have arisen in a similar fashion. However, without access to more detailed information or the original model and dataset, we were unable to verify this hypothesis.

Other than the seven exceptions discussed so far (and one more discussed in Section 4.2 and shown in the top left pane of Figure 2), the remaining 37 of the hyper-parameter response slices are quite smooth, with many of them being qualitatively similar in smoothness to that seen at the top left of Figure 1. The final non-convex hyper-parameter response, for Xgboost's `min_child_weight`, also appears smooth, but has a slope that is slightly negatively correlated with the loss.

4.2 Two-Dimensional Hyper-Parameter Response Slices

In Table 3, we show a summary of the results from our tests for uni-modality and convexity, as well as median FDC values for the two-dimensional hyper-parameter response slices centered around the global optima of each landscape. The results are quite similar to those from the one-dimensional analysis (see Section 4.1); however, all of the numbers are slightly lower – especially those from the tests for convexity. In total, of the 208 hyper-parameter response slices we tested (this excludes those for online LDA, for which true confidence intervals could not be obtained, see Section 3), 95.19% appear to be uni-modal (all but 10), and the median FDC is 0.73. However, the test for convexity rejected its null hypothesis more frequently, and only 58.87% of the response slices with two numeric hyper-parameters appear to be convex. In Table 3, we include the three response slices for online LDA as if it had confidence intervals of size ± 0.16%, which were small enough to fail to reject uni-modality, but not large enough to fail to reject convexity. Indeed, it appears that the landscape is most likely not convex, as it requires the intervals to be at least ± 8.70% before it fails to reject convexity.

The latent-structured SVM's `c` and `alpha` hyper-parameters yielded the most unusual two-dimensional response slice (see the top left pane of Figure 2). For many of the values of `alpha`, `c`'s response smoothly decreases monotonically prior to the optimal value near to $\texttt{c} = 10\,000$; however, there is a small "bump" around $\texttt{c} = 1\,000$ that causes the response to almost form a sub-optimal plateau. Most interestingly, for the four largest values of `alpha`, the smaller values of `c` yield saw-tooth-style responses instead of the smooth curves. Furthermore, each of the "teeth" in this part of the landscape align closely, which suggests that there exists a complex interaction between the two hyper-parameters. Despite this, only 2.5% of this response slice had locally significant partial derivatives, since with the exception of $\texttt{alpha} = 1$ and $\texttt{alpha} = 1.5$, all of the other neighbouring values of `alpha` yielded qualitatively similar hyper-parameter response slices in `c`. Note that this is the same `alpha` hyper-parameter that is pictured in the bottom right pane of Figure 1, hence the near-bi-modality in that one-dimensional response slice may in fact be due to a complex dependence of the loss upon the value of `alpha` rather than a spurious result.

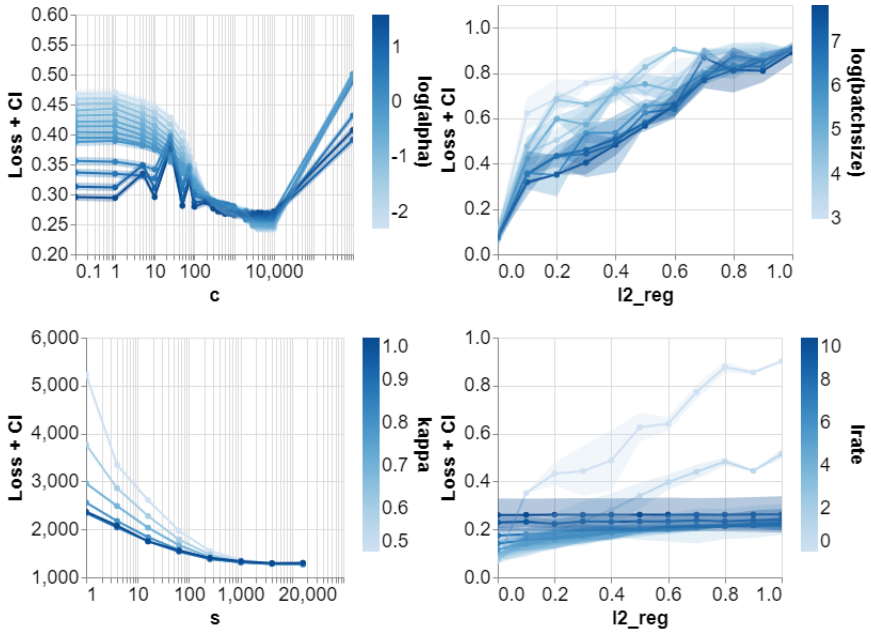

Fig. 2. Four examples of two-dimensional hyper-parameter response slices. From top to bottom, left to right: latent-structured SVM's `c` and `alpha` hyper-parameters, logistic regression's `l2_reg` and `batchsize`, online LDA's kappa and `s`, and logistic regression's `l2_reg` and `lrate`.

Another unusual, non-uni-modal hyper-parameter response slice was observed for logistic regression's `l2_reg` and `batchsize` hyper-parameters (see the top right pane of Figure 2). Similar to the previously discussed case, we see more evidence that `l2_reg` has a highly rugged response; however, unlike the rugged portion of the response between the latent-structured SVM's `c` and `alpha` hyper-parameters, the jagged peaks of `l2_reg`'s response slices for varying values of `batchsize` do not align. Instead, the shape of the landscape is suggestive of a highly rugged and jagged mountainscape that slowly levels off towards the top. However, recall that for this scenario only one validation and one test loss value were available. Since these losses are likely correlated, it is possible that the landscape only appears to have numerous local minima due to our assumption that the two loss values were *i.i.d.* when we calculated the confidence intervals. If so, then the landscape may actually be uni-modal, albeit with much larger variance than most other cases.

Nevertheless, many hyper-parameter optimizers never bother quantifying (accurately or at all) this source of variance in AutoML loss landscapes. Instead, they only perform a single run of the training algorithm on a single training and validation split (or they may take the mean over a few cross-validation folds). As a result, a robust hyper-parameter optimizer must still be prepared to occasionally deal with some local ruggedness that can cause small, sub-optimal local minima to speckle the landscape.

FCNet's `dropout_1` and `dropout_2` hyper-parameters on the protein structure dataset yield a third non-uni-modal response slice. In this case, there is only a single point in the landscape that is unreachable from the global optimum; however, if the confidence level of the test is just slightly increased from 95% to 96.52%, our test fails to reject its null hypothesis. It is also possible that this apparent sub-optimal local minima could be due to the discretization of the landscape, since its 3

by 3 grid may be too coarse to allow for diagonally-oriented basins to appear uni-modal. However, without access to the original model and dataset, we are unable to verify whether or not this is the case.

The remaining seven hyper-parameter response slices for which uni-modality was rejected all contain Xgboost's `subsample` hyper-parameter. In each case, we continue to see that one or more of the configurations with `subsample` = 0.5 caused the algorithm to exceed the running time cutoff, thus once again introducing spike-like barriers into the landscape. We speculate that re-running each of these instances with larger running time cutoffs (or perhaps even with the same running time cutoff, but on a machine with a smaller background load) would yield uni-modal responses. However, given that hyper-parameter optimizers would not have access to this kind of information while in use, we continue to analyze the landscape with these spikes present.

In the bottom right pane of Figure 2, we show the hyper-parameter response slice for online LDA's `kappa` and `s` hyper-parameters. Given the apparent smoothness in the response, it should come as no surprise that an interval size as small as ± 0.16% yields uni-modal responses for each of the three two-dimensional hyper-parameter response slices. Out of the four response slices shown in Figure 2, both the smoothness and simplicity of this response is the most representative of all of the other hyper-parameter response slices (not shown).

The hyper-parameter response in the bottom right pane of Figure 2 also exhibits an interesting property: If the hyper-parameters were to be optimized individually a single time in sequence or independently in parallel, one would still find a configuration very near to optimal. This remains true even though 74.29% of the local second-order partial derivatives are significant (when arbitrarily using an interval size of ± 0.16%). It was this observation that ultimately inspired the methodology described in Section 2.9, which seeks to answer the question: How often does this hold true in practice for other combinations of hyper-parameters? Clearly, it does not always hold (see, *e.g.*, the bottom right pane of Figure 2); nevertheless, 80.77% of the hyper-parameter pairs and permutations of hyper-parameter optimization order yielded landscapes so benign that our simplistic optimization process was able to find a configuration tied with optimal. This shows that, even though many of the hyper-parameters have statistically significant interactions within somewhat substantial fractions of their landscapes (see Table 7 in Appendix D), most of these interactions are relatively benign from the perspective of a hyper-parameter optimizer.

This trend held true quite often even for hyper-parameter response slices involving one or two categorical hyper-parameters. In fact, all of the hyper-parameter response slices with at least once categorical hyper-parameter also appear to be uni-modal. Unsurprisingly, in some cases, changing a categorical hyper-parameter shifts the loss up or down without substantially moving the optimum (see *e.g.*, the left pane of Figure 3); however, we would also expect to find cases where changing a categorical hyper-parameter would result in a completely different landscape with a new optimum for the other hyper-parameter. In fact, we did observe a small number of examples of the latter case (see, *e.g.*, the right pane of Figure 3); however, in each case, the optimum of the numeric hyper-parameter for a sub-optimal categorical value yielded an equal or greater loss for the configuration with the same numeric value and the optimal categorical value. Therefore, the overall hyper-parameter response slices were still uni-modal. Nevertheless, since our study included a relatively small number of categorical hyper-parameters, we remain skeptical that this behaviour will generalize to all – or perhaps even most – other scenarios. We leave the study of this question as future work.

4.3 Higher-Dimensional AutoML Loss Landscapes

We show the results for uni-modality, convexity and FDC applied to the full AutoML loss landscapes in Table 4. Contrary to most of the lower-dimensional hyper-parameter response slices of the

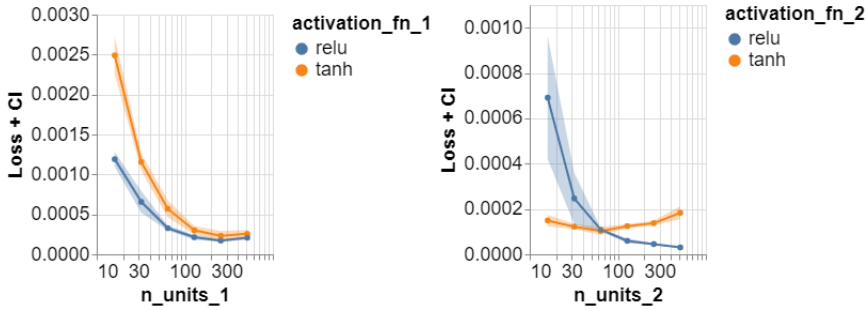

Fig. 3. Two examples of two-dimensional hyper-parameter response slices that include one numeric and one categorical hyper-parameter. Left: FCNet's `n_units_1` and `activation_fn_1` hyper-parameter on the slice localization dataset. Right: FCNet's `n_units_2` and `activation_fn_2` hyper-parameter on the naval propulsion dataset.

landscapes we studied, we see that all of the complete AutoML loss landscapes have relatively low FDC, which suggests that they should be quite challenging to optimize. However, the test for uni-modality tells a different story: It rejects uni-modality for only two of the seven landscapes we were able to test. For the remaining scenario, online LDA, only 3.12% of the landscape is unreachable – even without using confidence intervals at all. Furthermore, no deep (if small) sub-optimal modes exist in this landscape, since an interval size of only ±1.36% is sufficient to fail to reject uni-modality, which limits the height of a barrier between two modes to being no greater than 2.76% of the loss of the best configuration corresponding to either mode (the derivation for this is in Appendix C).

Both scenarios for which uni-modality is rejected are for FCNet. In one case only three, and in the other only 41, out of 62 208 configurations are unreachable. Given that these numbers are so small, could this merely be due to random chance? To answer this question, we analyzed the distances between the unreachable configurations. If they were unreachable due to random chance, we might expect them to be spread out uniformly at random. For the protein structure instance set, the mean pairwise distance between unreachable configurations is 4.73. For comparison, we drew 1 000 sets of 41 random configurations from the landscape. The smallest mean pairwise distance out of the 1 000 samples was 9.61 and the mean was 10.36, hence the 41 unreachable configurations are clustered together. In fact, many of them are adjacent, thus forming 18 very small, sub-optimal modes. We calculated the depth of these modes as the amount the smallest upper bound of each would need to increase to become reachable from the global optimum. Since there are a small number of configurations with extremely bad loss, we measure the size of this increase as a percentage of the range of losses spanned by the optimal configuration and the 95^{th} percentile of the losses. The median increase in loss required is 1.92% and the largest is only 6.30%; hence, these modes are rather shallow.

On the other hand, convexity is rejected for all of the complete AutoML loss landscapes that we studied. In particular, the FCNet landscapes appear to be quite far from convex, with 38.76% to 50.05% of the lower bounds being interior to the convex hull of the upper bounds. One of the two scenarios that is closest to being convex is Xgboost on the covertype dataset. For this scenario, it would have been computationally expensive (requiring approximately 1.5 months on our machines) to exactly calculate the percentage of the lower bounds that were interior; however, given that we only needed to find a single such configuration to reject convexity, it sufficed to randomly sample and evaluate 5% of the lower bounds to estimate the percentage that are interior. Using this method we calculated a 95% confidence interval for the percentage of points that are interior as [9.56%,

Table 4. Summary of the tests for uni-modality and convexity, and FDC applied to the full AutoML loss landscapes. Includes the mean percentage of the landscape that was unreachable (UnR) from the global optimizer or interior (Int) to the convex hull of the upper bounds for those slices for which uni-modality or convexity could be rejected, respectively; the mean percentage of the lower-bounds that were co-planar (Co-P) to the convex hull of the upper bounds; and the mean percentage of the landscapes for which the tests appeared sensitive (Sen) according to our permutation-based analysis.

Model	Dataset	Interval	UnR	Sen	Int	Co-P	Sen	FDC
FCNet	SL	95%	0.00%	Yes	50.05%	0.00%	Yes	0.00
	PS	95%	0.03%	Yes	49.67%	0.00%	Yes	0.03
	NP	95%	0.00%	Yes	43.63%	0.03%	Yes	0.01
	PT	95%	< 0.01%	Yes	38.76%	0.00%	Yes	0.01
LSSVM	UniProbe	95%	0.00%	Yes	18.86%	0.36%	Yes	0.43
LogReg	MNIST	95%	0.00%	Yes	28.25%	0.10%	Yes	−0.03
XGB	CT	95%	0.00%	Yes	9.65%	0.21%	Yes	0.33
OLDA	Wiki	± 0.00%	3.12%	Yes	21.88%	61.46%	Yes	0.36
		± 1.36%	0.00%	Yes	6.25%	0.00%	Yes	0.36
		± 8.70%	0.00%	Yes	0.00%	0.00%	Yes	0.36

9.74%] – *i.e.*, just below 10% of the landscape would need to be altered for the entire landscape to appear convex. The other landscape that is somewhat close to being convex is online LDA, for which the loss intervals need to be ± 8.70% in size to fail to reject convexity. However, even the much smaller interval size of ± 1.36% leaves only 6.25% of the lower bounds interior to the convex hull.

While these results from the test for convexity on the full landscapes are quite different than those on the lower-dimensional hyper-parameter response slices, they should come as no surprise. In fact, since every one of the scenarios had at least one two-dimensional hyper-parameter response slice that was non-convex, it follows that the full landscapes must also all be non-convex.

In Table 5, we show the mean percentage of each landscape that is dependent on the 1^{st}- through 6^{th}-order partial derivatives, *i.e.*, for each n^{th}-order interaction we calculated the percentage of significant partial derivatives and then report the means for each order n. For landscapes that had them, the higher order interaction results remained relatively similar to the 6^{th} order interaction results (see Appendix D). Most of these scenarios behave roughly as we would expect: The first-order derivatives for most hyper-parameters affect relatively large portions of the landscapes. However, there are a few exceptions: For example, logistic regression's `n_epochs` hyper-parameter only affects 10.56% of the landscape. As the order of interactions increases, fewer tuples of hyper-parameters interact, and those that do affect smaller portions of the landscape. For example, 50% of the 2^{nd}-order interactions for logistic regression affect between 20.90% – 26.61% of the landscape, and the rest affect less than 7.07%; one of the 3^{rd}-order interactions affects 18.10% of the landscape, and the remaining four affect less than 5.21%.

However, the results are rather different for the FCNet landscapes. The higher-order parameter interactions are substantially more influential than expected. For the four scenarios between 19.90% – 24.27% of the landscapes are dependent upon interactions between all 9 hyper-parameters. The results from fANOVA (see Table 9) show additional support for the fact that the FCNet landscapes depend to a much larger degree on higher-order interactions than the other landscapes. In particular, fANOVA attributes between 17.01%–28.58% of the total variance in the FCNet landscapes to the 6^{th}-order interactions, compared to only 0.15%–4.50% for the 1^{st}-order effects. In comparison, for XGBoost, fANOVA attributes only 0.24% to the 6^{th}-order interactions and 58.94% to 1^{st}-order effects. One possible explanation for this is that the FCNet scenarios are the only ones we study that fall into the over-parameterized regime [Belkin et al. 2019], which may exhibit qualitatively different

AutoML loss landscapes. In particular, because there may be a large number of models which all obtain (nearly) perfect training scores, the hyper-parameters may interact in complex ways to produce models that interpolate differently between the training instances.

There is a qualitative difference between the fANOVA results and those regarding partial derivatives. For most scenarios, fANOVA reports that a small number of hyper-parameters are responsible for a large percentage of the variance in the objective function. The XGBoost landscape is a representative example: fANOVA reports that the 1st-order effect of the hyper-parameter `eta` is responsible for 52.6% of the variance, and the rest are individually responsible for less than 2.54%. In comparison, `eta` has the third largest percentage of non-zero partial derivatives (67.65%) and all but one of the rest have between 17.06% – 75.16% (the one remaining has 3.01%). However, these results are not contradictory, since the statistical test can detect small but significant partial derivatives. Indeed, all that can be interpreted from fANOVA's result for `eta` is that at least one value of `eta` yields very different performance from the rest.

To determine where hyper-parameters are most important, we excluded from the analysis any partial derivative that contained a configuration in the worst X% of the landscape (see Tables 10–13 for $X = 50\%$ and $X = 75\%$). The outcome of this analysis was mixed. For latent-structured SVMs, logistic regression and online LDA, the partial derivatives for most hyper-parameters were significant less often; *e.g.*, first-order partial derivatives of the hyper-parameters of latent-structured SVMs were found to be significant for 19.10% of the entire landscape on average, but only for 6.42% of the landscape when excluding the worst 50% of the configurations. However, we found the opposite to be true for FCnet and Xgboost; for example, the first-order partial derivatives for the FCNet landscape were significant 9.75% more often on average when the worst 50% of the configurations were excluded from the analysis.

This result for FCNet is consistent with the sub-optimal, plateau-shaped regions that we observed in FCNet's `init_lr` hyper-parameter response, suggesting that this behaviour likely also occurs for more of the slices of AutoML loss landscapes that were excluded from our one-dimensional analysis. For Xgboost, it is possible that if there are regions of the configuration space for which the running time of the algorithm tends to be high, then entire portions of the landscape may have been artificially censored due to a running time cutoff. (We observed one example of this bevahiour for `subample` = 0.5, which tended to have many censored runs reported as a loss of 1.) If this were to happen sufficiently often in clustered areas, then it could be the case that the worst configurations form regions of sub-optimal plateaus where the loss is 1.

We also observed one hyper-parameter with a particularly notable increase in the significance of its partial derivatives: logistic regression's `l2_reg` hyper-parameter. For example, when excluding the worst 75% of configurations, its 1st-order partial derivative became significant 25.37% more often. This indicates that `l2_reg`'s response must be steepest nearest to its optimum, possibly with a sub-optimal plateau embedded in its response – indeed, this is precisely what we observed in the one- and two-dimensional hyper-parameter response slices for it (see Figures 1 and 2).

Finally, we applied our simple test, wherein we optimized each hyper-parameter independently a single time in a random sequence. Because Xgboost has nearly 40 million possible permutations for the order in which its hyper-parameters can be optimized, we restricted ourselves to a small, 30 minute computation budget, which allowed for a random 537 100 permutations. For all of the other scenarios, we performed the analysis for all possible hyper-parameter permutations. The only scenario for which we found any permutations of the hyper-parameter optimization order that did not yield a result tied with optimal was online LDA, with an interval size of ±0.00%. However, even in this case, the optimization procedure found the optimal configuration in all but one of the six permutations. This result is surprising, especially given that we have observed non-trivial percentages of the landscapes that are dependent upon hyper-parameter interactions, as it suggests

Table 5. Hyper-parameter partial derivative significance result summary – part 1. Each column represents the mean percentage of the landscape with statistically significant partial derivatives for each partial derivative order.

Model	Dataset	Interval	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}	6^{th}
FCNet	SL	95%	66.45%	45.25%	34.06%	28.66%	25.62%	23.59%
	PS	95%	68.16%	44.02%	31.68%	26.74%	24.58%	23.60%
	NP	95%	53.66%	37.29%	29.93%	26.68%	25.15%	24.33%
	PT	95%	70.96%	45.82%	31.62%	26.25%	24.08%	22.98%
LSSVM	UniProbe	95%	19.10%	4.43%	0.96%	–	–	–
LogReg	MNIST	95%	36.21%	15.21%	7.77%	4.24%	–	–
XGB	CT	95%	43.57%	13.35%	4.88%	3.30%	2.98%	2.91%
OLDA	Wiki	± 1.36%	67.53%	17.07%	0.00%	–	–	–

that AutoML loss landscapes may be much simpler to optimize than previously assumed. One possible explanation for this result is our previous observation that in many (albeit not all) cases, the hyper-parameters tend to interact most strongly in regions of the configuration space that are far from optimal. An example can be seen in the bottom left pane of Figure 2, where the effect of `kappa` has no significant impact on the response in the three best values of `s`.

5 CONCLUSIONS AND FUTURE WORK

We introduced novel methods to test for significant deviations from uni-modality and convexity in n-dimensional AutoML hyper-parameter loss landscapes. We applied these methods to empirical data from a diverse set of state-of-the-art machine learning models and algorithms. All but two of the AutoML loss landscapes we studied appear to be uni-modal at a 95% significance level, and those that significantly deviate from uni-modality do so only slightly. At first glance, this result may appear contradictory with the 2.2% of the one-dimensional and 4.8% of the two-dimensional hyper-parameter response slices for which uni-modality was rejected. However, the reason for this must lie in the hyper-parameter interactions, which allow a path from the sub-optimal modes in the lower-dimensional slices to circumnavigate their barriers. We were able to reject convexity for all of the landscapes we studied at a 95% significance level; however, we nevertheless observed that 82.2% of the hyper-parameters yield convex responses when considered independently and when other hyper-parameters are fixed to their optimal values.

Furthermore, we showed that an intuitively related metric, fitness distance correlation (FDC) [Jones and Forrest 1995], fails to accurately characterize the simplicity of the structure present in most of the full AutoML loss landscapes. However, despite the clear evidence that most hyper-parameters induce highly-structured and exploitable responses in the corresponding loss landscapes, we also observed a small number of exceptions to this rule. For example, small values of latent-structured SVM's `alpha` hyper-parameter yielded a very smooth, wavy response in `c`, whereas large values of `alpha` yielded a saw-tooth response (as seen in the top left pane of Figure 2). Somewhat similarly, logistic regression's `l2_reg` hyper-parameter yielded a relatively noisy and rugged response, with some sub-optimal plateaus (see the top right and bottom right panes of Figure 2). Finally, we also observed that certain values of hyper-parameters (*e.g.*, `eta` in xgboost) are correlated with longer running times, which can cause the training algorithm to be censored, thus yielding spurious clusters of spikes and ridges, due to censored loss values.

We further introduced a novel method that assesses the significance of finite difference approximations of partial derivatives. Using this method, we found that most landscapes have only a small number of hyper-parameters that interact strongly. However, the FCNet landscapes we studied were qualitatively very different and exhibited large percentages of statistically significant high-order

partial derivatives. We leave as future work the investigation of whether or not this behaviour could be attributed to the fact that the FCNet landscape we studied may have been in the so-called "over-parameterized regime" [Belkin et al. 2019]. We observed that many hyper-parameters tend to have flatter responses near high-quality solutions; however, we did observe some exceptions – the most notable of which was logistic regression (particularly its `l2_reg` hyper-parameter), for which the response contains some sub-optimal plateaus and a steep drop near the optimum (see, *e.g.* the top right and bottom right panes of Figure 2).

Finally, we used a simplistic optimization procedure that naïvely optimizes each hyper-parameter independently, a single time and in a random order, to determine the extent to which interactions between hyper-parameters increase the complexity of the optimization problem. We found that in all cases this procedure was able to find final configurations that were statistically tied with the optimal configuration, according to the respective 95% confidence intervals. While this optimization procedure lacks several essential ingredients of state-of-the-art hyper-parameter optimizers (*e.g.*, the ability to make use of multi-fidelity estimates for the loss of a hyper-parameter configuration), it nevertheless demonstrates that the effects of the hyper-parameter interactions are relatively benign, despite their statistically significant presence.

Also somewhat unexpectedly, we found that the landscapes induced by the categorical hyper-parameters for the FCNet benchmarks are either completely or very close to uni-modal. Indeed, we speculate that this behaviour will not generalize to the combinatorial landscapes induced by model and pre-processor selection in machine learning pipelines. In particular, while a machine learning method that uses model m_1 might perform best with feature encoding mechanism e_1, it is not clear that e_1 will be optimal for a qualitatively different machine learning method that relies on model m_2. Therefore, if the choice for each pre-processor and model are encoded as categorical hyper-parameters (as is often done, see *e.g.*, Feurer et al. 2019) it is not clear that the neighbourhood relation defined in Section 2 will induce a uni-modal landscape.

A crucial ingredient of state-of-the-art hyper-parameter optimizers is their ability to make use of low-fidelity estimates of the loss of a given configuration, in order to quickly eliminate poorly performing configurations – often, such methods require orders of magnitude less computing resources to obtain high-quality configurations (see *e.g.*, Li et al. [2017] or Falkner et al. [2018]). Given the outstanding successes obtained by these methods, it is clear that low- and medium-fidelity estimates of AutoML loss landscapes must bear at least some level of similarity to their full-fidelity counterparts. However, better understanding how AutoML loss landscapes change as a function of fidelity, as well as a comparison of how different measures of fidelity (*e.g.*, a lower number of training iterations, a smaller training set or fewer cross-validation folds) impact those changes could help spark further breakthroughs in the state of the art of AutoML. We leave the investigation of these and other AutoML loss landscape analysis questions as future work.

In their current form, our landscape analysis methods remain prohibitively expensive to be useful as a means of *selecting* a given hyper-parameter optimization procedure for solving a particular AutoML scenario. However, it may be possible to modify or approximate some of these methods to obtain computationally-cheap exploratory landscape analysis features [Derbel et al. 2019; Mersmann et al. 2011]. If successful, the logical next step from this line of research is to consider an algorithm selection procedure [Abell et al. 2012; Belkhir et al. 2016; Kerschke et al. 2019; Malan 2018] to decide which hyper-parameter optimization procedure is most likely to produce high-quality results on a given AutoML scenario.

Of course, the most interesting question raised by our analysis is how to best exploit these insights for the development of faster, more efficient and more effective AutoML methods. Similar landscape structure [Pushak and Hoos 2018] in a related optimization problem has recently been provably exploited in one-dimensional problems [Hall et al. 2020] and empirically exploited in

higher-dimensional problems [Pushak and Hoos 2020]. In the latter, we used a generalization of the golden section search algorithm [Kiefer 1953] – which has optimal worst-case performance for one-dimensional uni-modal functions – to high-dimensional problems using a coordinate descent-based [Wright 2015] approach. Given the relatively benign hyper-parameter interactions that we observed in most scenarios, this and other coordinate descent-based algorithms may be especially well-suited to hyper-parameter optimization. However, one key difference in running time minimization landscapes [Pushak and Hoos 2018] and those investigated here is that all but a very small number of AutoML hyper-parameters appear to yield responses in the loss that are much less prone to random fluctuations between independent runs of the training algorithm. Nevertheless, our observations may also explain some of the recent success observed by Yakovlev et al. [2020], who combine gradient-based hyper-parameter optimization with coordinate descent – a method which could possibly be further improved by exploiting the approximate uni-modality and convexity that appears to arise in many AutoML loss landscapes.

REFERENCES

Tinus Abell, Yuri Malitsky, and Kevin Tierney. 2012. *Fitness landscape based features for exploiting black-box optimization problem structure.* IT University of Copenhagen.

Christof Angermueller, Tanel Pärnamaa, Leopold Parts, and Oliver Stegle. 2016. Deep learning for computational biology. *Molecular systems biology* 12, 7 (2016).

Nacim Belkhir, Johann Dréo, Pierre Savéant, and Marc Schoenauer. 2016. Feature based algorithm configuration: A case study with differential evolution. In *Proceedings of the Fourteenth International Conference on Parallel Problem Solving from Nature (PPSN 2016)*. 156–166.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. 2019. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences* 116, 32 (2019), 15849–15854.

James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for hyper-parameter optimization. In *Proceedings of the Twenty-fifth Conference on Neural Information Processing Systems (NeurIPS 2011)*. 2546–2554.

James S Bergstra and Yoshua Bengio. 2012. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research* 13, 10 (2012), 281–305.

André Biedenkapp, Marius Lindauer, Katharina Eggensperger, Frank Hutter, Chris Fawcett, and Holger H Hoos. 2017. Efficient parameter importance analysis via ablation with surrogates. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI 2017)*.

André Biedenkapp, Joshua Marben, Marius Lindauer, and Frank Hutter. 2018. CAVE: Configuration Assessment, Visualization and Evaluation. In *Proceedings of the Twelfth International Conference on Learning and Intelligent Optimization (LION 2018)*.

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

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the Twenty-Second ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD 2016)*. 785–794.

Andrea Coraddu, Luca Oneto, Aessandro Ghio, Stefano Savio, Davide Anguita, and Massimo Figari. 2016. Machine learning approaches for improving condition-based maintenance of naval propulsion plants. *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment* 230, 1 (2016), 136–153.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning* 20, 3 (1995), 273–297.

Pierre Simon de Laplace. 1820. *Théorie analytique des probabilités*. Vol. 7. Courcier.

Bilel Derbel, Arnaud Liefooghe, Sébastien Vérel, Hernan Aguirre, and Kiyoshi Tanaka. 2019. New features for continuous exploratory landscape analysis based on the SOO tree. In *Proceedings of the Fifteenth ACM/SIGEVO Conference on Foundations of Genetic Algorithms (FOGA 2019)*. 72–86.

Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

Katharina Eggensperger, Matthias Feurer, Frank Hutter, James S Bergstra, Jasper Snoek, Holger H Hoos, and Kevin Leyton-Brown. 2013. Towards an Empirical Foundation for Assessing Bayesian Optimization of Hyperparameters. In *NeurIPS workshop on Bayesian Optimization in Theory and Practice*.

Bashkansky Emil and Gadrich Tamar. 2013. Some statistical aspects of binary measuring systems. *Measurement* 46, 6 (2013), 1922–1927.

Stefan Falkner, Aaron Klein, and Frank Hutter. 2018. BOHB: Robust and Efficient Hyperparameter Optimization at Scale. In *Proceedings of the Thirty-Fifth International Conference on Machine Learning (ICML 2018)*. 1437–1446.

Chris Fawcett and Holger H Hoos. 2016. Analysing differences between algorithm configurations through ablation. *Journal of Heuristics* 22, 4 (2016), 431–458.

Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. 2015. Efficient and robust automated machine learning. In *Proceedings of the Twenty-Ninth Conference on Neural Information Processing Systems (NeurIPS 2015)*. 2962–2970.

Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost T Springenberg, Manuel Blum, and Frank Hutter. 2019. Auto-sklearn: efficient and robust automated machine learning. In *Automated Machine Learning*. Springer, Cham, 113–134.

Nicolo Fusi, Rishit Sheth, and Melih Elibol. 2018. Probabilistic matrix factorization for automated machine learning. *Proceedings of the Thirty-Second Conference on Neural Information Processing Systems (NeurIPS 2010)* 31 (2018).

Unai Garciarena, Roberto Santana, and Alexander Mendiburu. 2018. Analysis of the Complexity of the Automatic Pipeline Generation Problem. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2018)*. IEEE, 1–8.

Franz Graf, Hans-Peter Kriegel, Matthias Schubert, Sebastian Pölsterl, and Alexander Cavallaro. 2011. 2D image registration in CT images using radial image descriptors. In *Proceedings of the Fourteenth International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI 2011)*. 607–614.

George T Hall, Pietro S Oliveto, and Dirk Sudholt. 2020. Fast Perturbative Algorithm Configurators. In *Proceedings of the Sixteenth International Conference on Parallel Problem Solving from Nature (PPSN 2020)*. Springer, 19–32.

Matthew Hoffman, Francis R Bach, and David M Blei. 2010. Online learning for latent Dirichlet allocation. In *Proceedings of the Twenty-fourth Conference on Neural Information Processing Systems (NeurIPS 2010)*. 856–864.

Holger H. Hoos and Thomas Stützle. 2005. *Stochastic Local Search: Foundations & Applications*. Morgan Kaufmann Publishers Inc.

Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. 2013. Identifying key algorithm parameters and instance features using forward selection. In *Proceedings of the Seventh International Conference on Learning and Intelligent Optimization (LION 2013)*. 364–381.

Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. 2014a. An Efficient Approach for Assessing Hyperparameter Importance. In *Proceedings of the Thirty-First International Conference on Machine Learning (ICML 2014)*. 754–762.

Frank Hutter, Manuel López-Ibáñez, Chris Fawcett, Marius Lindauer, Holger H Hoos, Kevin Leyton-Brown, and Thomas Stützle. 2014b. AClib: A Benchmark Library for Algorithm Configuration. In *Proceedings of the Fourteenth International Conference on Learning and Intelligent Optimization (LION 2014)*. 36–40.

Terry Jones and Stephanie Forrest. 1995. Fitness distance correlation as a measure of problem difficulty for genetic algorithms.. In *Proceedings of the Sixth International Conference on Genetic Algorithms (ICGA 1995)*, Vol. 95. 184–192.

Kirthevasan Kandasamy, Gautam Dasarathy, Jeff Schneider, and Barnabás Póczos. 2017. Multi-fidelity bayesian optimisation with continuous approximations. In *Proceedings of the Thirty-Fourth International Conference on Machine Learning (ICML 2017)*. 1799–1808.

Kirthevasan Kandasamy, Jeff Schneider, and Barnabás Póczos. 2015. High dimensional Bayesian optimisation and bandits via additive models. In *Proceedings of the Thirty-Second International Conference on Machine Learning (ICML 2015)*. 295–304.

Pascal Kerschke, Holger H Hoos, Frank Neumann, and Heike Trautmann. 2019. Automated algorithm selection: Survey and perspectives. *Evolutionary Computation* 27, 1 (2019), 3–45.

Jack Kiefer. 1953. Sequential minimax search for a maximum. *Proceedings of the American mathematical society* 4, 3 (1953), 502–506.

Aaron Klein and Frank Hutter. 2019. Tabular benchmarks for joint architecture and hyperparameter optimization. *arXiv preprint arXiv:1905.04970* (2019).

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Proceedings of the Twenty-Sixth Conference on Neural Information Processing Systems (NeurIPS 2012)*. 1097–1105.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436–444.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. 2017. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research* 18, 1 (2017), 6765–6816.

Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Jonathan Ben-tzur, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. 2020. A System for Massively Parallel Hyperparameter Tuning. In *Proceedings of Machine Learning and Systems (MLSys 2020)*. Vol. 2. 230–246.

Katherine M Malan. 2018. Landscape-aware constraint handling applied to differential evolution. In *Proceedings of the Seventh International Conference on Theory and Practice of Natural Computing (TPNC 2018)*. 176–187.

Katherine M Malan. 2021. A survey of advances in landscape analysis for optimisation. *Algorithms* 14, 2 (2021), 40.

Katherine M Malan and Andries P Engelbrecht. 2013. A survey of techniques for characterising fitness landscapes and some possible ways forward. *Information Sciences* 241 (2013), 148–163.

Olaf Mersmann, Bernd Bischl, Heike Trautmann, Mike Preuss, Claus Weihs, and Günter Rudolph. 2011. Exploratory landscape analysis. In *Proceedings of the Thirteenth Annual Conference on Genetic and Evolutionary Computation (GECCO 2011)*. 829–836.

Kevin Miller, M Pawan Kumar, Ben Packer, Danny Goodman, and Daphne Koller. 2012. Max-margin min-entropy models. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics (AIStats 2012)*. 779–787.

Matheus Nunes, Paulo M Fraga, and Gisele L Pappa. 2021. Fitness landscape analysis of graph neural network architecture search spaces. In *Proceedings of the Twenty-Third International Genetic and Evolutionary Computation Conference (GECCO 2021)*. 876–884.

Cristiano G Pimenta, Alex GC de Sá, Gabriela Ochoa, and Gisele L Pappa. 2020. Fitness Landscape Analysis of Automated Machine Learning Search Spaces. In *Proceedings of the Twentieth European Conference on Evolutionary Computation in Combinatorial Optimization (EVOCOP 2020 – Part of EvoStar)*. Springer, 114–130.

Erik Pitzer and Michael Affenzeller. 2012. A comprehensive survey on fitness landscape analysis. *Recent Advances in Intelligent Engineering Systems* (2012), 161–191.

Yasha Pushak and Holger H Hoos. 2018. Algorithm Configuration Landscapes: More Benign than Expected? In *Proceedings of the Fifteenth International Conference on Parallel Problem Solving from Nature (PPSN 2018)*. 271–283.

Yasha Pushak and Holger H Hoos. 2020. Golden Parameter Search: Exploiting Structure to Quickly Configure Parameters in Parallel. In *Proceedings of the Twenty-Second International Genetic and Evolutionary Computation Conference (GECCO 2020)*.

Anna Rakitianskaia, Eduan Bekker, Katherine M Malan, and Andries Engelbrecht. 2016. Analysis of error landscapes in multi-layered neural networks for classification. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2016)*. IEEE, 5270–5277.

Prashant S Rana. 2013. Physicochemical properties of protein tertiary structure data set. https://archive.ics.uci.edu/ml/datasets/Physicochemical+Properties+of+Protein+Tertiary+Structure. *UCI Machine Learning Repository* (2013).

Christian M Reidys and Peter F Stadler. 2001. Neutrality in fitness landscapes. *Appl. Math. Comput.* 117, 2-3 (2001), 321–350.

Nuno M Rodrigues, Sara Silva, and Leonardo Vanneschi. 2020. A Study of Fitness Landscapes for Neuroevolution. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2020)*. IEEE, 1–8.

Samuel S Shapiro and Martin B Wilk. 1965. An analysis of variance test for normality (complete samples). *Biometrika* 52, 3/4 (1965), 591–611.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Vedavyas Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy P Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature* 529 (2016), 484–489.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. In *Proceedings of the Twenty-sixth Conference on Neural Information Processing Systems (NeurIPS 2012)*. 2951–2959.

Jost T Springenberg, Aaron Klein, Stefan Falkner, and Frank Hutter. 2016. Bayesian optimization with robust Bayesian neural networks. In *Proceedings of the Thirtieth Conference on Neural Information Processing Systems (NeurIPS 2016)*. 4134–4142.

Athanasios Tsanas, Max A Little, Patrick E McSharry, and Lorraine O Ramig. 2010. Accurate telemonitoring of Parkinson's disease progression by noninvasive speech tests. *IEEE transactions on Biomedical Engineering* 57, 4 (2010), 884–893.

Willem A van Aardt, Anna S Bosman, and Katherine M Malan. 2017. Characterising neutrality in neural network error landscapes. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2017)*. IEEE, 1374–1381.

Jean-Paul Watson. 2010. An introduction to fitness landscape analysis and cost models for local search. In *Handbook of Metaheuristics*. Springer International Publishing, 599–623.

Sewall Wright. 1932. The roles of mutation, inbreeding, crossbreeding, and selection in evolution. *Proceedings of the Eleventh International Congress of Genetics* 8 (1932).

Stephen J Wright. 2015. Coordinate descent algorithms. *Mathematical Programming* 151, 1 (2015), 3–34.

Anatoly Yakovlev, Hesam F Moghadam, Ali Moharrer, Jingxiao Cai, Nikan Chavoshi, Venkatanathan Varadarajan, Sandeep R Agrawal, Sam Idicula, Tomas Karnagel, Sanjay Jinturkar, and Nipun Agarwal. 2020. Oracle AutoML: a fast and predictive AutoML pipeline. *Proceedings of the Forty-Sixth International Conference on Very Large Data Bases (VLDB 2020)* 13, 12 (2020), 3166–3180.

Chengrun Yang, Yuji Akimoto, Dae W Kim, and Madeleine Udell. 2019. OBOE: Collaborative filtering for AutoML model selection. In *Proceedings of the Twenty-Fifth ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (SIGKDD 2019)*. 1173–1183.

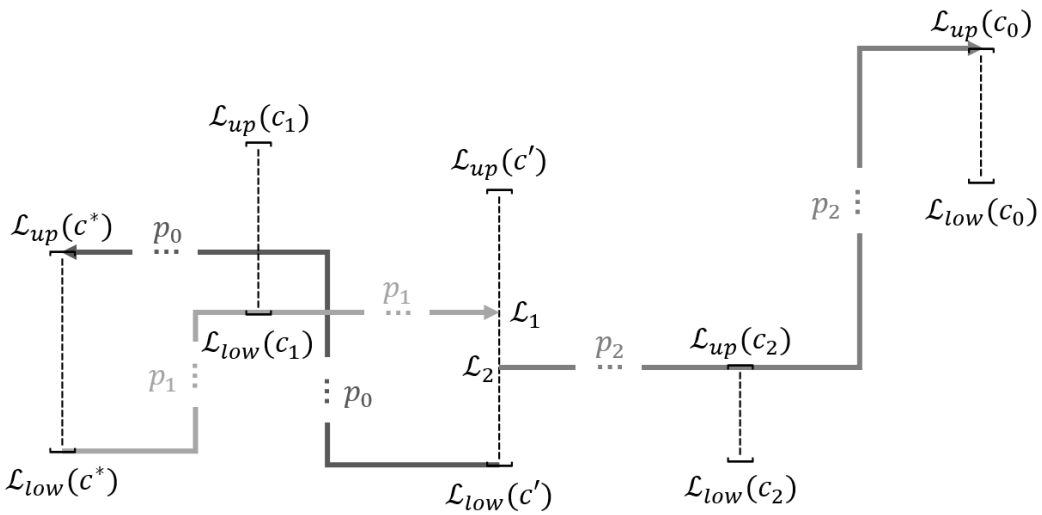

Fig. 4. Diagram used in the proof of correctness for test for uni-modality.

A PROOF OF CORRECTNESS FOR TEST FOR UNI-MODALITY

Theorem 1 (Correctness of Test for Uni-Modality). *Let $G' = (V', E')$ be a neighbourhood relation graph defined for an AutoML loss landscape that contains a set of pre-evaluated configurations C, such that each configuration $c \in C$ has a corresponding confidence interval $[\mathcal{L}_{low}(c), \mathcal{L}_{up}(c)]$ for the loss of the machine learning method. If $\mathcal{L}_{up}(c^*) \leq \mathcal{L}_{up}(c)$ for all $c \in C$, and there exists $c_0 \in C$ that is not reachable from c^*, then no uni-modal, piece-wise affine function exists that is contained within the confidence intervals, and hence uni-modality can be rejected for the landscape.*

Proof. Let c^* be a global minimum of C, *i.e.*,

$$\mathcal{L}_{\text{up}}(c^*) \leq \mathcal{L}_{\text{up}}(c) \text{ for all } c \in C, \tag{9}$$

and let $c_0 \in C$ be a configuration that is not reachable from c^*. Because this c_0 exists, we cannot construct a piece-wise affine function that is contained within the confidence intervals of C that also contains c^* as a global minima. It remains to be shown that this is a sufficient condition to conclude that no uni-modal, piece-wise affine function exists that is contained within the confidence intervals of C.

Assumption. For eventual contradiction, assume that a piece-wise affine uni-modal function does exists within the confidence intervals of C. This function must have at least one global minima $c' \in C$ from which all other points $c \in C$ must be reachable. We will show (1) that there must exist a path p_1 from c^* to c' and a path p_2 from c' to c_0 (see Figure 4); however, since they cannot be concatenated (otherwise c_0 would be reachable from c^*), we will show (2) that c^* must not have been a global minima of C, which is a contradiction.

Step 1. First, we show that there must exist a path p_1 from $(c^*, \mathcal{L}_{\text{low}}(c^*))$ to $(c', \mathcal{L}_1)$. Since c^* is reachable from c', there must be a path p_0 from $(c', \mathcal{L}_{\text{low}}(c'))$ to $(c^*, \mathcal{L}_{\text{up}}(c^*))$. Furthermore, based on the definition of reachability, the height of path p_0 must never exceed $\mathcal{L}_{\text{up}}(c^*)$, therefore $\mathcal{L}_{\text{low}}(c) \leq \mathcal{L}_{\text{up}}(c^*)$ for all c on p_0. Combining this with Equation 9, we obtain

$$\mathcal{L}_{\text{low}}(c) \leq \mathcal{L}_{\text{up}}(c^*) \leq \mathcal{L}_{\text{up}}(c) \text{ for all } c \text{ on } p_0. \tag{10}$$

Hence, we can clearly construct at least one path from c^* to c' with height less than or equal to $\mathcal{L}_{\text{up}}(c^*)$ for all c on p_0 that is the reverse of p_0. Let p_1 be the path from $(c^*, \mathcal{L}_{\text{low}}(c^*))$ to $(c', \mathcal{L}_1)$ with minimal height at all configurations c on p_1. The final height, $\mathcal{L}_1$, of p_1 must be constrained by the lower bound of the configuration, c_1, that has the smallest lower bound in p_1, *i.e.*, $\mathcal{L}_1 = \mathcal{L}_{\text{low}}(c_1) \leq \mathcal{L}_{\text{low}}(c)$ for all c on p_1. Combining this with Equation 10, we have

$$\mathcal{L}_{\text{low}}(c_1) = \mathcal{L}_1 \leq \mathcal{L}_{\text{up}}(c^*). \tag{11}$$

Step 2. Let p_2 be the certifying path from $(c', \mathcal{L}_2)$ to $(c_0, \mathcal{L}_{\text{up}}(c_0))$ with the maximum possible height for all configurations c on p_2. The beginning height, $\mathcal{L}_2$, of p_2 must be constrained by the upper bound of the configuration, c_2, that has the smallest upper bound in p_2, *i.e.*,

$$\mathcal{L}_2 = \mathcal{L}_{\text{up}}(c_2) \leq \mathcal{L}_{\text{up}}(c) \text{ for all } c \text{ on } p_2. \tag{12}$$

Since c_0 is not reachable from c^*, we must not be able to concatenate the paths p_1 (from c^* to c') and p_2 (from c' to c_0). This can only happen if the height, $\mathcal{L}_1$, at the end of p_1 is above the height, $\mathcal{L}_2$, at the beginning of p_2, *i.e.*,

$$\mathcal{L}_2 < \mathcal{L}_1. \tag{13}$$

Contradiction. Finally, combining Equations 11, 12 and 13, we obtain

$$\mathcal{L}_{\text{up}}(c_2) = \mathcal{L}_2 < \mathcal{L}_1 \leq \mathcal{L}_{\text{up}}(c^*), \tag{14}$$

i.e.,

$$\mathcal{L}_{\text{up}}(c_2) < \mathcal{L}_{\text{up}}(c^*). \tag{15}$$

However, this contradicts our precondition that c^* is a global minima of C (see Equation 9). Therefore, our original assumption must be false: No uni-modal, piece-wise affine function exists within the confidence intervals of C. ■

B XGBOOST AUTOML LOSS LANDSCAPE HYPER-PARAMETER GRID

The grid of hyper-parameters configurations we used to evaluate Xgboost on the covertype dataset can be obtained by taking the cross product of the lists of hyper-parameter values in Table 6

Table 6. XGBoost hyper-parameter grid

Hyper-parameter	Grid of Values
eta	[0, 0.15, ..., 0.9]
gamma	[0, 5, 10]
max_dept	[2, 6, 10]
min_child_weight	[1, 10.5, 20]
max_delta_step	[0, 5, 10]
subsample	[0.01, 0.505, 1]
colsample_bytree	[0.5, 0.75, 1.0]
colsample_bylevel	[0.5, 0.75, 1.0]
lambda	[1, 5.5, 10]
alpha	[0, 5, 10]
num_round	[50, 150, 250]

C MAXIMUM BARRIER HEIGHT FOR ONLINE LDA

Let c_a be a configuration that is a sub-optimal local minima with a corresponding barrier c_b, blocking it from the optimal configuration c^*. That is, c_b is a configuration with the smallest loss through which any path from c_a to the c^* must pass. Technically, a barrier c_b for a local-minima c_a is defined to be any configuration such that

$$\mathcal{L}(c_b) = \min_{p_{c_a \to c^*}} \max_{c \text{ on } p} \mathcal{L}(c), \tag{16}$$

where $p_{c_a \to c^*}$ denotes a path from c_a to c^*.

Let the height of a barrier c_b for local minima c_a be defined as

$$H(c_b, c_a) := \mathcal{L}(c_b) - \mathcal{L}(c_a). \tag{17}$$

Let s define the size of the intervals for a configuration c, *i.e.*,

$$\begin{aligned} \mathcal{L}_{\text{up}}(c) &:= \mathcal{L}(c) \cdot (1+s) \text{ and} \\ \mathcal{L}_{\text{low}}(c) &:= \mathcal{L}(c) \cdot (1-s). \end{aligned} \tag{18}$$

Theorem 2 (Maximum Barrier Height). *Any interval size, $s < 1$, for which the landscape cannot have uni-modality rejected, bounds the height, $H(c_b, c_a)$, of any barrier, c_b, for any local minima, c_a, such that*

$$H(c_b, c_a) \leq \mathcal{L}(c_a) \cdot \left(\frac{1+s}{1-s} - 1 \right). \tag{19}$$

Proof. By definition, since interval size s yields a landscape for which uni-modality cannot be rejected, we know that c_a must be reachable from c^*, and hence

$$\mathcal{L}_{\text{low}}(c_b) \leq \mathcal{L}_{\text{up}}(c_a) \iff 0 \leq \mathcal{L}_{\text{up}}(c_a) - \mathcal{L}_{\text{low}}(c_b). \tag{20}$$

Trivially,

$$\begin{aligned} H(c_b, c_a) &= \mathcal{L}(c_b) - \mathcal{L}(c_a) \\ &\leq \mathcal{L}(c_b) - \mathcal{L}(c_a) + \left(\mathcal{L}_{\text{up}}(c_a) - \mathcal{L}_{\text{low}}(c_b) \right) \\ &\leq \left(\mathcal{L}(c_b) - \mathcal{L}_{\text{low}}(c_b) \right) + \left(\mathcal{L}_{\text{up}}(c_a) - \mathcal{L}(c_a) \right). \end{aligned} \tag{21}$$

From the definition in Equation 18, we know that

$$\begin{aligned} \mathcal{L}(c_b) - \mathcal{L}_{\text{low}}(c_b) &= s \cdot \mathcal{L}(c_b) \text{ and} \\ \mathcal{L}_{\text{up}}(c_a) - \mathcal{L}(c_a) &= s \cdot \mathcal{L}(c_a), \end{aligned} \tag{22}$$

which we can substitute into Equation 21 to obtain

$$\mathcal{L}(c_b) - \mathcal{L}(c_a) \leq s \cdot \mathcal{L}(c_b) + s \cdot \mathcal{L}(c_a). \tag{23}$$

By re-arranging and solving for $\mathcal{L}(c_b)$ we obtain

$$\mathcal{L}(c_b) \leq \mathcal{L}(c_a) \cdot \frac{1+s}{1-s}, \tag{24}$$

which we can plug back into the definition in Equation 17 to obtain the desired result. ∎

Using Theorem 2, and the fact that a confidence interval of size $s = 0.0136$ yields a landscape for online lDA for which uni-modality cannot be rejected, we know that the maximum height of any barrier for any sub-optimal local minima in the landscape can be no more than $\left(\frac{1+0.0136}{1-0.0136} - 1\right) \approx 0.0276$ times the height of the local minima.

Table 7. The mean percentage of locally significant hyper-parameter interactions based on an analysis of the partial derivatives (Sig ∂^2), the 2nd order fANOVA scores (fANOVA), and the mean probability of obtaining a configuration that is tied with optimal if each hyper-parameter is optimized once sequentially in a random order (Tied $\overline{w}$ Opt). All results are for the two-dimensional hyper-parameter response slices centered around the global optimizers of the AutoML loss landscapes.

Type	# Slices	Model	Dataset	Interval	Sig ∂^2	fANOVA	Tied $\overline{w}$ Opt
N×N	15	FCNet	NP	95%	52.80%	15.05%	76.67%
			PS	95%	54.56%	4.26%	76.67%
			PT	95%	50.56%	10.20%	83.33%
			SL	95%	61.53%	11.36%	86.67%
	3	LSSVM	UP	95%	1.78%	5.38%	83.33%
	6	LogReg	MNIST	95%	20.97%	10.17%	91.67%
	55	XGB	CT	95%	20.91%	9.35%	84.55%
	3	OLDA	Wiki	± 8.70%	0.00%	16.61%	100.00%
				± 0.16%	75.81%	16.61%	50.00%
				± 0.00%	100.00%	16.61%	50.00%
N×C	18	FCNet	NP	95%	63.89%	18.95%	72.22%
			PS	95%	60.93%	4.14%	80.56%
			PT	95%	43.89%	15.15%	72.22%
			SL	95%	63.70%	15.26%	86.11%
C×C	3	FCNet	NP	95%	0.00%	17.24%	66.67%
			PS	95%	33.33%	3.38%	100.00%
			PT	95%	66.67%	28.60%	50.00%
			SL	95%	33.33%	2.81%	83.33%

Table 8. Hyper-parameter partial derivative significance result summary – part 2.

Model	Dataset	Interval	7^{th}	8^{th}	9^{th}	10^{th}	11^{th}
FCNet	SL	95%	22.05%	21.08%	19.80%	–	–
	PS	95%	23.23%	23.36%	23.80%	–	–
	NP	95%	23.92%	23.70%	24.27%	–	–
	PT	95%	22.24%	22.05%	20.20%	–	–
XGB	CT	95%	2.92%	2.97%	3.05%	3.14%	3.04%

D EXTENDED HYPER-PARAMETER INTERACTION RESULTS

In Table 7 we show the results of our analysis on the two-dimensional slices of the landscapes. Throughout the remainder of the tables, we denote by "–" a result does not exist because there are not enough hyper-parameters to compute the indicated quantity. In the case of the higher-order interaction tables wherein we excluded the worst 50% or 75% of the landscapes, we denote by "nan" the scenarios for which the exclusion of the indicated fraction of the configurations resulted in us being unable to calculate any of the indicated partial derivatives because all of them contained at least one such censored configuration. In Table 8 we show the remaining partial derivative significance summary for the scenarios with more than seven hyper-parameters. In Table 9 we show the sum of the variance explained by each order n of hyper-parameter interactions for each landscape. In Tables 10, 11, 12 and 13 we show the same analysis used to determine the fraction of the landscapes with locally significant partial derivatives of various orders; however, in the first and second two sets of tables we drop the worst 50% and 75% of the configurations from the analysis, respectively.

Table 9. Hyper-parameter fANOVA importance result summary.

Model	Dataset	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th
FCNet	SL	0.44	2.80	9.75	20.46	27.13	23.03	12.20	3.70	0.49	–	–
	PS	4.50	8.88	14.22	15.50	16.13	17.01	14.68	7.63	1.46	–	–
	NP	0.15	0.48	2.58	9.05	20.21	28.58	24.69	11.85	2.41	–	–
	PT	0.27	0.82	3.36	10.53	21.58	28.27	22.82	10.34	2.01	–	–
XGB	CT	58.94	12.96	14.47	9.24	3.70	0.24	0.15	0.14	0.10	0.11	0.02
LogReg	MNIST	76.25	21.89	1.76	0.11	–	–	–	–	–	–	–
LSSVM	UP	87.39	11.59	1.01	–	–	–	–	–	–	–	–
OLDA	Wiki	70.16	29.50	0.34	–	–	–	–	–	–	–	–

Table 10. Hyper-parameter partial derivative significance excluding worst 50% of configurations – part 1.

Model	Dataset	Interval	1st	2nd	3rd	4th	5th	6th
FCNet	SL	95%	74.72%	49.51%	36.26%	30.36%	27.40%	25.63%
	PS	95%	81.17%	50.58%	33.46%	26.70%	23.98%	23.05%
	NP	95%	64.97%	43.78%	33.42%	28.40%	26.01%	24.56%
	PT	95%	77.29%	50.04%	32.37%	25.60%	22.90%	21.84%
XGB	CT	95%	54.98%	13.41%	4.34%	3.44%	3.36%	3.36%
LogReg	MNIST	95%	32.70%	11.18%	4.79%	3.01%	–	–
LSSVM	UniProbe	95%	6.42%	0.00%	0.00%	–	–	–
OLDA	Wiki	± 1.36%	41.92%	0.42%	0.00%	–	–	–
		± 5.10%	1.23%	0.00%	0.00%	–	–	–

Table 11. Hyper-parameter partial derivative significance excluding worst 50% of configurations – part 2.

Model	Dataset	Interval	7th	8th	9th	10th	11th
FCNet	SL	95%	24.43%	25.36%	26.74%	–	–
	PS	95%	23.11%	24.60%	22.00%	–	–
	NP	95%	24.07%	23.53%	33.33%	–	–
	PT	95%	21.98%	23.34%	21.05%	–	–
XGB	CT	95%	3.41%	3.47%	3.51%	3.40%	3.70%

Table 12. Hyper-parameter partial derivative significance excluding worst 75% of configurations – part 1

Model	Dataset	Interval	1st	2nd	3rd	4th	5th	6th
FCNet	SL	95%	75.30%	50.46%	35.61%	29.34%	26.52%	23.74%
	PS	95%	81.34%	51.28%	32.41%	26.60%	24.65%	nan
	NP	95%	63.91%	43.22%	32.76%	27.36%	nan	nan
	PT	95%	72.67%	46.60%	30.42%	24.86%	nan	nan
XGB	CT	95%	64.86%	15.53%	3.95%	3.35%	3.28%	3.26%
LogReg	MNIST	95%	38.25%	14.34%	6.44%	3.82%	–	–
LSSVM	UP	95%	6.11%	0.00%	0.00%	–	–	–
OLDA	Wiki	± 1.36%	11.70%	0.00%	0.00%	–	–	–
		± 5.10%	0.00%	0.00%	0.00%	–	–	–

Table 13. Hyper-parameter partial derivative significance excluding worst 75% of configurations – part 2

Model	Dataset	Interval	7th	8th	9th	10th	11th
FCNet	SL	95%	nan	nan	nan	–	–
	PS	95%	nan	nan	nan	–	–
	NP	95%	nan	nan	nan	–	–
	PT	95%	nan	nan	nan	–	–
XGB	CT	95%	nan	nan	nan	nan	nan