Hyperparameter Optimization using Learning Curves

AutoML Assignment 2

Rwik Kamilya, Luc van Driel

1 Introduction

Hyperparameter optimization (HPO) aims to find a configuration of model and hyperparameters that minimizes validation error on a given dataset. Exhaustively training every configuration on the full training set is often prohibitively expensive. Learning–curve based methods exploit performance measurements at increasing training set sizes ("anchor sizes") to extrapolate the final performance and early–stop poor configurations that are unlikely to perform well.

In this report, we study two such vertical HPO methods: *Learning Curve Cross–Validation (LCCV)* with optimistic linear extrapolation [2] and an *Inverse Power Law (IPL)* approach [1]. We use k–NN learning curves from the LCDB (Learning Curve DataBase) [3] for datasets 6, 11, and 1457, and train a Random Forest surrogate per dataset to simulate ground–truth validation scores. On top of these surrogates, we simulate vertical HPO runs where LCCV and IPL decide, based on partial learning curves, which configurations to stop early.

Our research question is: *How do LCCV and IPL compare in terms of final regret and evaluation cost across different datasets when applied to LCDB k–NN learning curves?*

2 Methodology and Implementation

The LCDB provides validation scores for many k–NN hyperparameter configurations at multiple anchor sizes for each dataset. For datasets 6, 11, and 1457 we train one *Random Forest Regressor (RFR)* surrogate per dataset (scikit-learn defaults), using the hyperparameters and the anchor size as input features and the validation score as target. Categorical parameters are one–hot encoded and missing values are mean–imputed. We randomly split each dataset into 80% training and 20% validation and report Spearman rank correlation between predicted and true scores at the final anchor s_T.

In all subsequent experiments, surrogate predictions are treated as oracle scores. Vertical HPO proceeds along the dataset–specific anchor schedule. At each anchor s_t we query the surrogate to obtain a score and apply LCCV or IPL to extrapolate to the target anchor size s_T and decide whether to continue or early–stop a configuration. Regret is defined as $C_{\text{method}}(s_T) - C_{\text{oracle}}(s_T)$, where C is the validation error (lower is better), so regret is non-negative. The cost of evaluating a configuration at anchor size s_t is $c(s_t) = \left(\frac{s_t}{s_T}\right)^{\zeta}$, with $\zeta > 0$ a dataset–independent exponent, reflecting that training cost grows with the fraction of data used. We report three metrics per dataset and method: final regret, total cost (sum of $c(s_t)$ over all evaluations), and the fraction of configurations that are early–stopped.

In all HPO simulations, the first configuration is evaluated on all anchors to initialise *the best score observed so far* at s_T. For every subsequent configuration, we evaluate anchors sequentially; after each new anchor we extrapolate to s_T using either LCCV or IPL. If the extrapolated score does not exceed *the best score observed so far*, the configuration is early–stopped; otherwise the configuration proceeds to the next anchor. Whenever a configuration is fully evaluated at s_T, *the best score observed so far* is updated if improved.

2.1 LCCV

LCCV linearly extrapolates the tail of the learning curve. Let $s_1 < s_2 < \cdots < s_T$ denote the anchor sizes and C_t the surrogate score at anchor s_t. After observing two anchors (s_{t-1}, C_{t-1}) and (s_t, C_t) we estimate the local slope $\beta = \frac{C_t - C_{t-1}}{s_t - s_{t-1}}$ and extrapolate linearly to s_T as $\widehat{C}_T = C_t + \beta(s_T - s_t)$, which we use as an optimistic prediction of the final score.

2.2 Inverse Power Law

The IPL method instead fits a parametric inverse power law [1]. Given scores C at anchors s, we fit $\hat{\mu}_s = \alpha + \beta s^{-\gamma}$ by ordinary least squares on all observed anchors of the configuration, clamp any negative α, β, γ to zero, and use $\hat{\mu}_{s_T}$ as the prediction at the final anchor. As more anchors are observed, the fit can capture the curvature of the learning curve, potentially enabling more informed early–stopping decisions than the purely local linear method.

For each dataset and method, we sample $n_{\text{configs}} = 100$ candidate configurations and run 5 random seeds. For our experiments, we use $\zeta = 1.0$ in the cost model.

3 Experiments and Results

To assess the validity of the results, we evaluate the RFR surrogates using Spearman rank correlation. On the held-out split, Spearman rank correlations at the final anchor s_T are all above 0.9, indicating that the surrogates mostly preserve the ordering of configurations. For each dataset and method, the results are averaged over multiple random seeds. We record the surrogate score at s_T at each evaluation, measure final regret, and compute the total cost. Below, the total cost, early-stops, and final regret are summarized for each dataset.

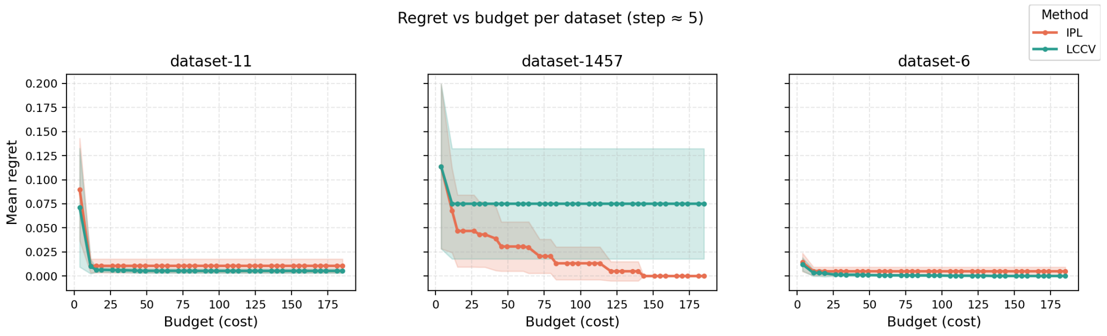

Figure 1: Average regret + spread of 5 iterations of IPL and LCCV for increasing costs on datasets 11, 1457, 6 from left to right.

The run-level summaries in Figure 2 summarize total cost, fraction early–stopped, and final regret across datasets and three main patterns appear. On dataset 11, both methods achieve very small regret with comparable costs. On dataset 6, IPL reaches almost zero regret at relatively low cost by early-stopping nearly all configurations, whereas LCCV keeps more configurations alive until s_T and therefore spends substantially more budget. On dataset 1457, LCCV is remarkably cheap, because it prunes aggressively, but suffers from higher and more variable regret. IPL attains consistently lower regret at the expense of a much higher total cost.

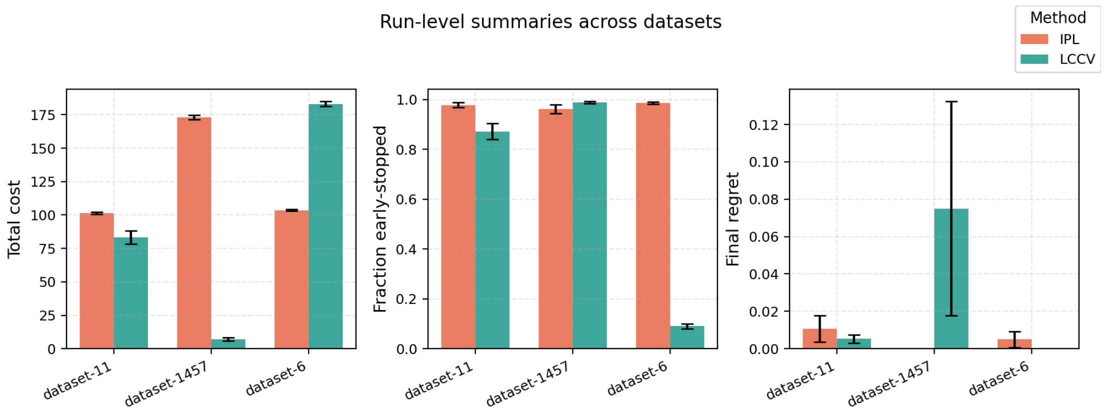

Figure 2: Total cost, fraction early-stopped, and final regret of IPL and LCCV per dataset, averaged over 5 seeds.

4 Discussion

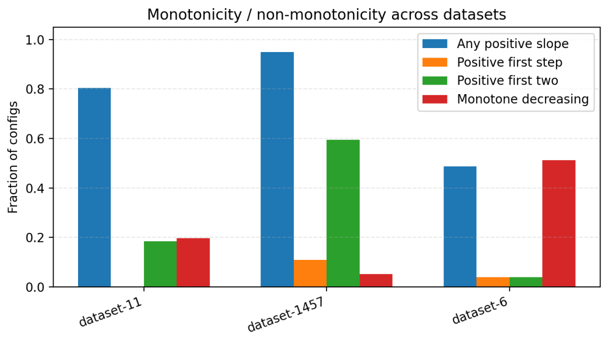

Figure 3: Fractions of configurations satisfying different monotonicity or non–monotonicity criteria across datasets.

Figure 3 summarizes monotonicity statistics of the learning curves for each dataset. Figures 1 and 2 show that neither LCCV nor IPL dominates across all datasets; their behavior looks to be dependent on properties of the learning curves. On dataset 11, both methods achieve very low regret at modest cost, with LCCV being slightly cheaper and marginally better. Here, the learning curves are mostly monotonic, although many configurations show at least one non–decreasing step. Only a minority of configurations increase in error in the first anchors (Figure 3), so both extrapolation rules make reasonably reliable early–stopping decisions.

Dataset 6 seems to be remarkably suited to learning curve extrapolation. LCCV gets to a regret of zero, but IPL achieves a similar result at almost half the cost. Because the extrapolated scores under LCCV often stay above the best score observed so far, most configurations are evaluated up to the final anchor, explaining its high cost. For IPL, the shape of the learning curves aligns well with the convex inverse power law; Figure 3 shows that about half of the configurations are monotonically decreasing, allowing IPL to prune aggressively while maintaining low regret.

On dataset 1457 simple optimistic linear extrapolation falls apart. Almost all configurations have at least one positive slope, and nearly 60% already increase in error in the first two steps (Figure 3), which makes early anchors a poor predictor of final performance. In this highly non–monotonic regime, LCCV often observes a small improvement followed by a noisy or slightly worsening step; extrapolating the last segment then yields a pessimistic prediction at s_T that does not beat the current best configuration. As a result, LCCV aggressively prunes most configurations early, leading to a remarkably low evaluation cost, but substantially higher and more variable regret. IPL is more robust here, because it is able to fit anchors jointly with its convex model. Even when individual steps are noisy, the global fit can still recover a plausible continuation, so more promising configurations survive to the final anchor. This behavior is consistent with the low regret of IPL on dataset 1457, at the price of a much higher cost.

5 Conclusion

We compared two learning–curve–based vertical HPO strategies: LCCV with optimistic linear extrapolation and an Inverse Power Law (IPL) model, on LCDB k-NN learning curves using Random Forest surrogates. The experiments show that their relative performance is strongly dataset–dependent. On smooth, mostly monotonic curves with informative early anchors (dataset 6), IPL can prune aggressively and still find near–optimal configurations at low cost. On moderately noisy curves (dataset 11), both methods achieve similarly low regret with comparable budgets. On highly non–monotonic curves with misleading early anchors (dataset 1457), LCCV's local optimistic extrapolation becomes brittle and prunes too aggressively, resulting in low cost, but relatively high regret, whereas IPL maintains low regret at the expense of more evaluations, indicating robustness.

In our experiments, there is no single best vertical HPO rule. When evaluation budget is tight and the learning curves are smooth, the simple optimistic extrapolation of LCCV is appealing. For datasets with non–monotonic or noisy curves, IPL's extra cost could be worth it. In practice, considering rudimentary learning-curve properties such as monotonicity should guide the decision for an early-stopping method.

References

[1] Tobias Domhan, Jost Tobias Springenberg, and Frank Hutter. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves, 2015.

[2] Felix Mohr and Jan N. van Rijn. Fast and informative model selection using learning curve cross-validation, 2021.

[3] Felix Mohr, Tom J. Viering, Marco Loog, and Jan N. van Rijn. LCDB 1.0: An extensive learning curves database for classification tasks, 2022.