Hyperparameter Optimization

AutoML Assignment 2

Po-Chin Chang, Jiaxuan Yu

1 Introduction

Hyperparameter Optimization (HPO) is essential in machine learning but computationally intensive. To address this, surrogate models offer an alternative by predicting configuration performance using prior learning curve data without actual model training. Learning curve extrapolation methods enhance efficiency by using partial training results to predict the full training performance [1]. In this experiment, we implement simplified versions of two learning curve extrapolation methods: Learning Curve Cross-Validation (LCCV) and Inverse Power Law (IPL) extrapolation and evaluate their effectiveness in hyperparameter optimization tasks based on their efficiency, performance, and robustness, providing insights into the practical applicability and limitations.

2 Methods

2.1 Hyperparameter Optimization using learning curves

2.1.1 Learning Curve Cross-Validation (LCCV)

LCCV [1] evaluates configurations at increasing anchor sizes, performing extrapolations after each anchor using a simplified formula 1, based on the slope between points.:

$$C_T = C_t + (s_T - s_t)\,\frac{C_{t-1} - C_t}{s_{t-1} - s_t} \tag{1}$$

Here, C_t represents the performance at the current anchor s_t, and C_{t-1} is the performance at the previous anchor s_{t-1}. This formula estimates the performance at s_T, the final anchor by calculating the slope between the last two anchor sizes, which assumes linear performance behavior between the anchors. This method estimates an optimistic performance in the full dataset. Therefore, if the estimated performance at the final anchor exceeds the best performance observed so far, the anchor size will increase, allowing further evaluation. Conversely, if the estimated performance is lower than the best-known value, the configuration is discarded to conserve computational resources.

2.1.2 IPL (Inverse Power Law)- Extrapolation

To predict configuration performance with the largest anchor size based on previous performance at a fixed schedule of anchor sizes, the simplest approach involves fitting a model of three-parameter inverse power law (IPL) [2] to the empirical learning curve given by the equation 2:

$$\mu_s = \alpha + \beta s^{-\gamma} \tag{2}$$

where μ_s represents the performance of the model at a given anchor size s, α, β, and $\gamma > 0$ are parameters to be optimized. The approach fits performance data from a fixed learning curve schedule to the IPL model, extrapolating performance at full anchor size. Configurations predicted to show no improvement are discarded. Otherwise, the configuration will be evaluated on the full dataset.

2.2 Surrogate Model

The surrogate model is trained on previously collected data from the Learning Curve Database (LCDB) and is designed to reduce the computational cost by predicting the performance of configurations.

3 Experimental Setup

In our experiment, all methods are executed with 100 iterations, corresponding to 100 configurations. The performance is measured as the error rate, and the best performance is recorded for each approach. Additionally, the accumulated anchor size, representing the total anchor sizes processed by the surrogate model is calculated to indicate the computational resource consumption. For LCCV, the process starts from the tenth-to-last anchor size in the Learning Curve Database (LCDB). If performance improves, the anchor size is incrementally increased, continuing the evaluation. We consider this number of anchor sizes sufficient to effectively demonstrate the discarding process of LCCV. In the case of IPL, the first half of the anchor sizes in the LCDB datasets are used to fit the power law model (pow3). As even the smallest dataset, dataset-1457, includes 5 anchor sizes, it provides sufficient data for the process. Lastly, Random Search only uses the final anchor size to get performance by the surrogate model. The results of Random Search serve as a baseline for comparing the performance and computational savings of other methods.

dataset	mse	R2 score	spearman correlation
dataset-1457	0.0001	0.9952	0.9886
dataset-11	0.0000	0.9986	0.9983
dataset-6	0.0000	0.9998	0.9997

Table 1: Performance of surrogate model on different datasets.

Method	Metric	dataset-1457	dataset-11	dataset-6
LCCV	Best Performance	0.7177	0.0356	0.0395
	Accumulated Anchor Size	24522	33353	4961845
IPL	Best Performance	0.6366	0.0943	0.0395
	Accumulated Anchor Size	42300	25000	951800
Random Search	Best Performance	0.5281	0.0285	0.0395
	Accumulated Anchor Size	120000	50000	1600000

Table 2: Comparison of best performance and accumulated anchor size of different methods across datasets.

4 Experiment Results

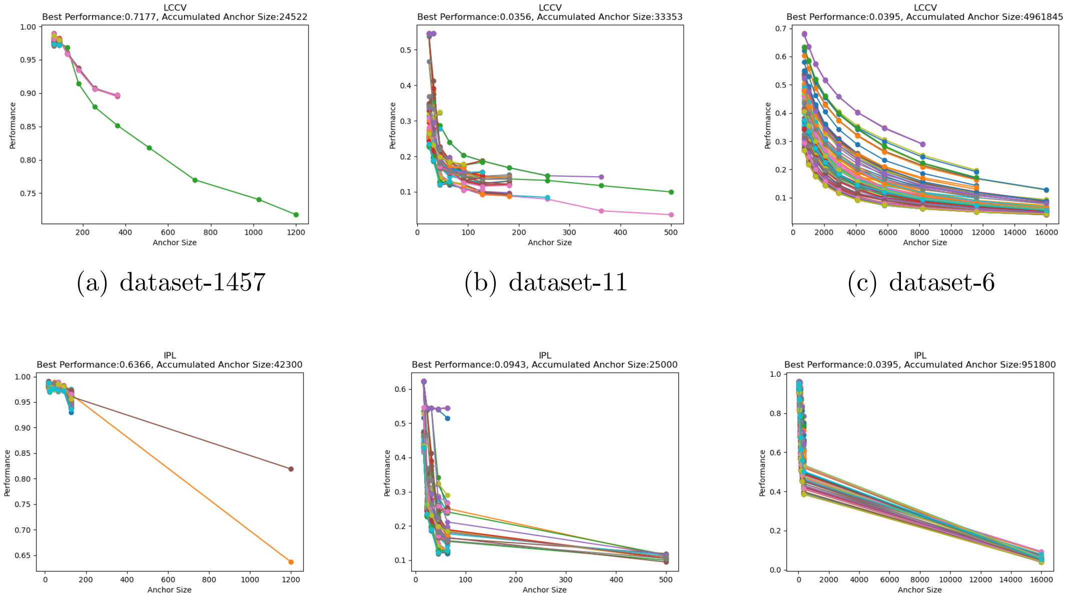
4.1 The Working of Surrogate Model

Table 1 summarizes the surrogate model's performance with LCCV and IPL using MSE, R^2 Score, and Spearman Correlation. The MSE values are extremely low (0.0001 or 0.0000), R^2 scores are near 1 (0.9952 to 0.9998), and Spearman correlations are nearly perfect (0.9886 to 0.9997), indicating the model's robustness and accuracy in predicting performance across datasets.

4.2 Performance Comparison of LCCV and IPL

From Figure 1 and Table 2, we observe that for dataset-1457, LCCV achieves an error rate of 0.7177 with an anchor size of 24,522, while IPL performs better at 0.6366 with 42,300. On dataset-11, LCCV outperforms IPL with an error rate of 0.0356 vs. 0.0943, with similar computational costs. For dataset-6, both achieve the same error rate of 0.0395, but IPL is more efficient, requiring only 951,800. Figure

1 shows that dataset size impacts the ability of LCCV and IPL to retain or discard configurations during optimization. On smaller datasets (dataset-1457 and dataset-11), both methods discard more configurations early, with IPL saving computational resources comparable to LCCV. For larger datasets like dataset-6, both retain more configurations, but IPL demonstrates higher efficiency with a smaller accumulated anchor size, highlighting the trade-off between performance and efficiency.



(a) dataset-1457

(b) dataset-11

(c) dataset-6

(d) dataset-1457

(e) dataset-11

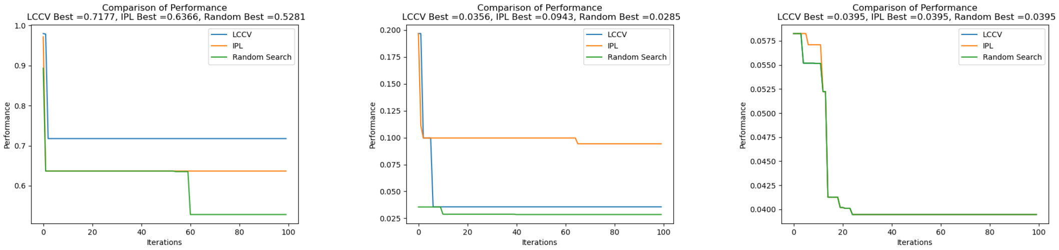
(f) dataset-6

Figure 1: Comparison of the performance and ccumulated anchor size of LCCV (Figures (a), (b), (c)) and IPL (Figures (d), (e), (f)) across three datasets. It demonstrates that the number of configurations extending to the final anchor size varies across datasets. Fewer configurations are extended to the final anchor size with smaller datasets while on the largest dataset(dataset-6), more configurations are retained until the end but IPL discards more according to the results of accumulated anchor size.

4.3 Performance Comparison of LCCV, IPL, and Random Search

Figure2 shows that for dataset-1457, IPL converges quickly to find the best configuration, while LCCV converges more slowly. In the case of dataset-11, Random Search achieves the best performance (0.0285), followed closely by LCCV, while IPL performs the worst with the highest error rate (0.0943). Random Search and LCCV converge early with minimal further improvement, whereas IPL improves more slowly but fails to match the other two methods. Lastly, for dataset-6, all three methods achieve the same best performance, proposing LCCV and IPL are well performed on a larger dataset and the performance curves of all three methods are nearly overlapping. As analyzed in the previous section, both methods tend to retain more configurations on larger dataset, resulting in similar convergence trends with Random Search.

As shown in Figure3, for dataset-6, LCCV consumes more resources than Random Search, highlighting its inefficiency on larger datasets. In contrast, IPL efficiently reduces computational costs, saving around 40% compared to training 100 configurations on full datasets, indicating its effectiveness in handling larger datasets. For dataset-1457, LCCV is efficient, requiring 24,522 anchor size compared to Random Search's 120,000, saving over 70% of resources. However, for IPL on dataset-1457, by discarding configurations based on extrapolations, only those showing potential improvement over the best-seen value are evaluated on the full dataset. Therefore, the large accumulated anchor size on LCCV may be caused by the steep slope at initial anchor sizes, leading the estimated extrapolations to be too optimistic, and consuming more computational resources.

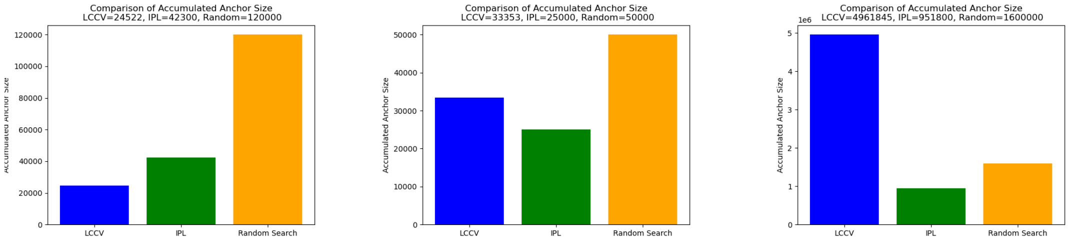


(a) dataset-1457

(b) dataset-11

(c) dataset-6

Figure 2: Comparison of best performance variations for LCCV, IPL, and Random Search across different datasets. The x-axis is iterations (number of configurations). The y-axis is the best performance so far. LCCV achieves the lowest error rate on dataset-11, while IPL underperforms. On dataset-1457, LCCV outperforms IPL but with a smaller difference. For dataset-6, LCCV and IPL achieve nearly the same error rates, and Random Search demonstrates the lowest error rates in all cases.



(a) dataset-1457

(b) dataset-11

(c) dataset-6

Figure 3: Comparison of accumulated anchor sizes for LCCV, IPL, and Random Search across different datasets. The x-axis is different methods. The y-axis is accumulated anchor size. IPL exhibits lower accumulated anchor sizes compared to Random Search across all datasets, while LCCV shows variability, being more efficient on dataset-1457 but less so on dataset-11.

5 Conclusion and Discussion

In conclusion, we attempt the potential of combining surrogate model with LCCV and IPL, benchmarking their performance and evaluating their ability to reduce computational costs. Our result indicates that IPL consistently can achieve robust performance with lower resource consumption across datasets, making it an efficient method. LCCV excels on smaller datasets but struggles on larger datasets, resulting in higher computational costs. Furthermore, LCCV could benefit from adaptive anchor schedules to handle complex datasets more effectively. Since the three-parameter IPL may not fit the data well due to uncertainties in fixed anchor schedules, fine-tuning parameters or exploring alternative extrapolation models could enhance performance and robustness, as IPL demonstrates strong scalability in this experiment.

References

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

[2] Felix Mohr and Jan N. van Rijn. Learning curves for decision making in supervised machine learning – a survey, 2022.