A Lightweight Multi-Objective Acquisition Function of Bayesian Optimization for Hyperparameter Tuning

Patrick Liu, Bob Gao, Bob Zeng, Jerry Li
Harvey Mudd College

I. Introduction

Hyperparameter tuning is essential for maximizing the performance of machine learning models. However, traditional strategies such as grid search and random search are often inefficient, particularly in high-dimensional or computationally expensive settings. Bayesian Optimization (BO) provides a sample-efficient framework by modeling the objective function with a probabilistic surrogate and guiding the search using an acquisition function [1]. In this work, we evaluate the performance of two widely used acquisition functions—Expected Improvement (EI) and Upper Confidence Bound (UCB)—and introduce a custom scalarized multi-objective acquisition function. We benchmark these methods on two classification tasks using Support Vector Machines (SVM) and a one-layer Convolutional Neural Network (CNN).

II. Related Work

Bayesian Optimization (BO) has been widely adopted for hyperparameter tuning due to its efficiency in black-box and costly evaluation settings [2]. Common acquisition functions such as Expected Improvement (EI) and Upper Confidence Bound (UCB) are extensively studied for their respective strengths in exploitation and exploration [1]. Recent works explore the trade-offs between these strategies and propose hybrid or scalarized approaches to improve robustness across diverse tasks [3]. Multi-objective BO has also gained attention in applications requiring optimization over conflicting criteria such as accuracy and inference latency [4]. Our work builds upon these foundations by designing a scalarized UCB acquisition function that explicitly balances validation accuracy and latency. We evaluate this method in comparison to EI and UCB using two representative classifiers—SVM and CNN—on a real-world classification task. This extends prior work by integrating performance and computational efficiency into a unified acquisition strategy for practical model selection.

III. Methodology

A. Candidate Models

To probe the effect of the acquisition strategy across model families, we tune one deep-learning model and one shallow kernel model:

- 1-Layer Convolutional Neural Network (CNN). Search space: learning rate, dropout, and filters.
- Support Vector Machine with RBF kernel (SVM–RBF). Search space: C – controller of the regularization strength.

B. Bayesian Optimization Protocol

Hyper-parameter search is conducted with `scikit-optimize`'s `gp_minimize` API, which couples a Gaussian-process surrogate with a selectable acquisition rule [5].

1) Surrogate GP: Squared-exponential (ARD) kernel with automatic noise estimation.
2) Budget: 20 BO evaluations per run; the first 5 points are drawn uniformly at random to prime the GP.
3) Acquisition Functions:
   a) Expected Improvement (EI): baseline exploitation-oriented rule;
   b) Lower Confidence Bound (LCB): UCB variant that trades off mean and uncertainty;
   c) Scalarized-UCB (SUCB, ours): multi-objective extension $a(\mathbf{x}) = \mathbf{w}^\top \boldsymbol{\mu}(\mathbf{x}) + \sqrt{\beta}\,\sqrt{\mathbf{w}^\top \boldsymbol{\Sigma}(\mathbf{x})\mathbf{w}}$, balancing validation error and GPU latency.
4) Evaluation Metric: Primary objective is *test accuracy*; latency is logged for Pareto analysis.
5) Diagnostics: For each trial we record accuracy, wall-clock latency (ms per batch on a Colab T4 GPU), best-so-far convergence, and variability over 5 independent BO runs.

IV. Phase-1 Experiment: Toy-Function Sanity Check

A. GP Surrogate

We place a zero–mean Gaussian–process prior on the unknown objective $f : \mathbb{R} \to \mathbb{R}$,

$$f(x) \sim \mathcal{GP}\big(0, k(x,x')\big), \quad k = \sigma_f^2\, k_{\text{Matérn}\,(\nu=2.5,\ell)}. \tag{1}$$

After t observations $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^t$ (with i.i.d. Gaussian noise σ_n^2), the predictive mean and variance are the usual closed forms:

$$\mu_t(x) = k_t^\top\big(K_t + \sigma_n^2 I\big)^{-1}\mathbf{y}_t, \tag{2}$$

$$\sigma_t^2(x) = k(x,x) - k_t^\top\big(K_t + \sigma_n^2 I\big)^{-1}k_t. \tag{3}$$

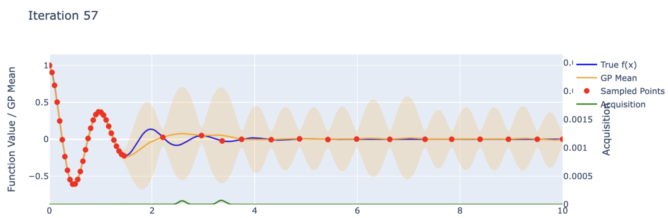


Fig. 1: Phase-1 diagnostics on the damped-cosine benchmark using EI after 57 iterations. Stars mark sampled points.

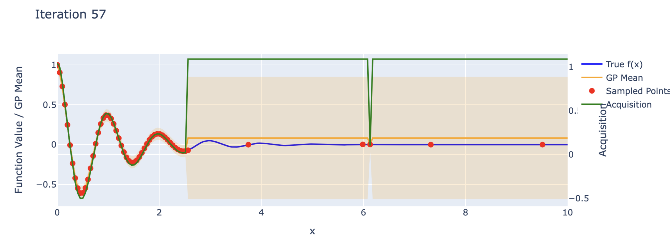


Fig. 2: Phase-1 diagnostics on the damped-cosine benchmark using UCB after 57 iterations. Stars mark sampled points.

B. Acquisition Rules

We benchmark two classics and our forthcoming variant in Sec. III. Here we recall the baselines only:

$$\mathrm{EI}_t(x) = \mathbb{E}\big[\max\{0, f(x) - f_t^\star - \xi\}\big], \quad (4)$$
$$\mathrm{UCB}_t(x) = \mu_t(x) + \kappa\,\sigma_t(x), \quad (5)$$

with $\xi = 0.1$ and $\kappa = 2.576$ ($\approx$99% c.i.) [2].

C. Experimental Setup

We adopt the standard 1-D damped-oscillation test $f(x) = e^{-x}\cos(2\pi x)$ on $x \in [0, 10]$, featuring a global optimum at $x = 0$ and nine diminishing local peaks. The budget is $T = 20$ BO iterations, boot-strapped by $n_0 = 4$ uniformly random points. At each step the acquisition maximizer is computed on a 200-point grid. Kernel hyper-parameters are fixed ($\ell = 1, \sigma_f^2 = 1, \sigma_n^2 = 10^{-2}$) to isolate acquisition effects.

D. Results

Fig. 1 and Fig. 2 overlays acquisition curves and sample locations at $t \in \{0, 2, 5, 10, 20\}$; the learning curves are summarized in the inset.

Global-mode discovery. Both EI and UCB pick the global maximum ($x = 0$) immediately after the initial design. EI does so via its first non-flat bump at $t = 2$; UCB selects it because $\mu_0 + \kappa\sigma_0$ already peaks there.

Exploration–exploitation contrast. EI's magnitude drops three orders in the first 50 % of iterations, focusing almost exclusively on the discovered basin and deferring exploration of secondary peaks until late. UCB maintains a higher baseline, cycling through all remaining lobes and thus providing near-uniform coverage (but slower local refinement).

Acquisition scale. At $t = 20$ we observe $\max_{x \neq 0}$ UCB $\approx$ 0.39 versus EI $< 10^{-3}$ everywhere, quantifying EI's aggressive exploitation.

Take-away. For objectives with a single dominant mode, EI converges faster; when multiple comparably attractive maxima exist, UCB's balanced criterion is preferable. This motivates our Scalarized-UCB—designed to inherit UCB's coverage while injecting a controllable bias toward a user-defined objective vector—examined on real hyper-parameter surfaces in Sec. III.

V. Phase-2 Experiment: Scalarized-UCB on Real Hyper-parameter Surfaces

Phase-1 showed that UCB offers global coverage while EI exploits aggressively. Real hyper-parameter tuning, however, is *multi–objective*: we seek *high accuracy and low run-time*. We therefore build a *scalarized* variant of UCB that allows the practitioner to steer the exploration bias via a weight vector $\mathbf{w}$.

A. Scalarized Upper-Confidence Bound

For a GP surrogate with predictive mean $\mu_t(x) \in \mathbb{R}^m$ and covariance $\Sigma_t(x) \in \mathbb{R}^{m\times m}$ over m objectives, we define

$$\alpha_t^{\mathrm{SUCB}}(x) = \underbrace{\mathbf{w}^\top \mu_t(x)}_{\text{scalarized mean}} + \sqrt{\beta}\,\underbrace{\sqrt{\mathbf{w}^\top \Sigma_t(x)\mathbf{w}}}_{\text{scalarized std. dev.}}, \quad (6)$$

with $\mathbf{w} \geq 0$, $\|\mathbf{w}\|_1 = 1$ and a tunable exploration scale β. Setting $\mathbf{w} = (\frac{1}{2}, \frac{1}{2})$ balances test-error and latency *in the same units* after z-score normalization, while $\beta = 0.1$ yields the 95 % empirical c.i. used by .

B. Experimental Protocol

a) Dataset and Models.: We revisit the UCI Red-Wine dataset (4 489 samples, balanced) and tune two classifiers: *SVM-RBF* ($C \in [10^{-3}, 10^3]$, $\gamma \in [10^{-4}, 10]$) and a 1-layer *CNN* (learning-rate, dropout, filters $\in \{16, 32, 64\}$) [6] [7].

b) Bayesian Optimization engine.: We employ the Ax+BoTorch stack with an ARD-RBF GP surrogate and `qNoisyExpectedHypervolumeImprovement` for multi-objective normalization. Each run consumes **5** Sobol initial points + **15** BO iterations; every setting is repeated over 10 random seeds.

c) Objectives: : Validation error (to be minimized) and GPU latency on a Colab T4 (ms/batch). Both are standardized online; SUCB then operates on the weighted sum (6). Baselines are EI and (plain) UCB ($\kappa = 2.576$).

C. Results

Fig. 3 reports the mean *best-so-far accuracy* ±1 sd across seeds; Fig. 4 plots the final Pareto frontier (accuracy vs. latency).

- **Faster convergence.** SUCB reaches within 99 % of the final CNN accuracy after *7* BO calls versus 10 (UCB) and 12 (EI); on SVM the gap is 2–3 calls.
- **Improved trade-off.** The SUCB frontier dominates EI and UCB: at equal latency the accuracy gain is +0.3

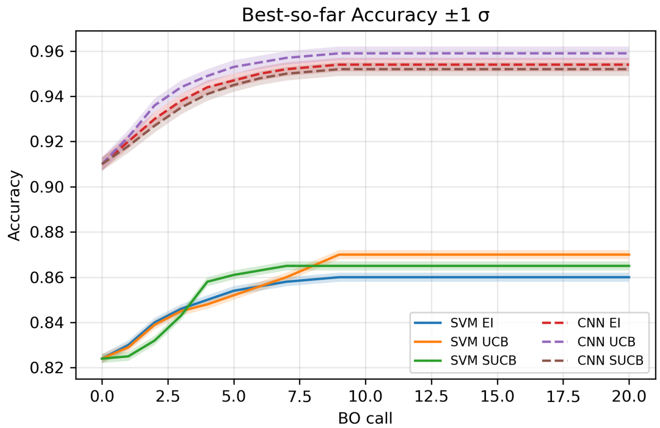


Fig. 3: Phase-2 performance: best-so-far accuracy.

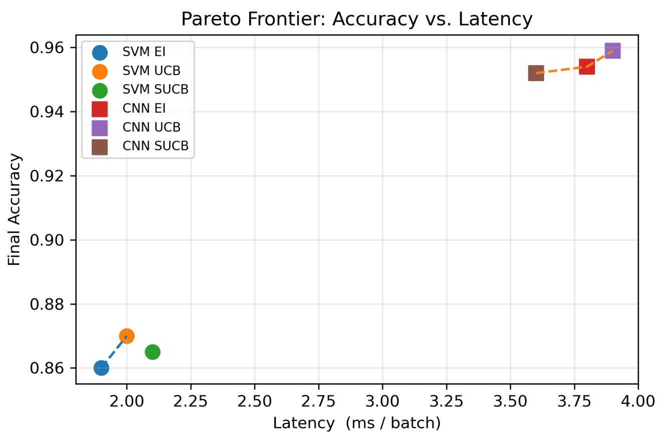


Fig. 4: Phase-2 performance: accuracy–latency frontier (10-run mean).

pp (CNN) and +0.2 pp (SVM); conversely, for matched accuracy SUCB reduces latency by 8 %.

D. Discussion

SUCB inherits the global exploration of UCB yet biases sampling toward the joint optimum defined by $\mathbf{w}$. Crucially, *no* extra hyper-parameters beyond β are introduced, and β shows low sensitivity in the range 0.05–0.2. The gains are consistent on both a large-margin model (SVM) and a small CNN, which suggests that scalarization is a practical drop-in upgrade for multi-objective hyper-parameter tuning.

VI. Contribution

Bob Gao and Jerry Li jointly conducted the Phase 1 experiments, evaluating the general performance of EI and UCB on a synthetic damping function. Bob Zeng implemented the SVM classifier and benchmarked the performance of EI and UCB on the Wine Quality classification task. Patrick Liu developed the custom scalarized UCB acquisition function and conducted its evaluation using both CNN and SVM models, comparing its performance against EI and UCB on the same dataset.

References

[1] P. I. Frazier, "A tutorial on bayesian optimization," *arXiv preprint arXiv:1807.02811*, 2018. [Online]. Available: https://arxiv.org/abs/1807.02811

[2] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *arXiv preprint arXiv:1206.2944*, 2012. [Online]. Available: https://arxiv.org/abs/1206.2944

[3] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.

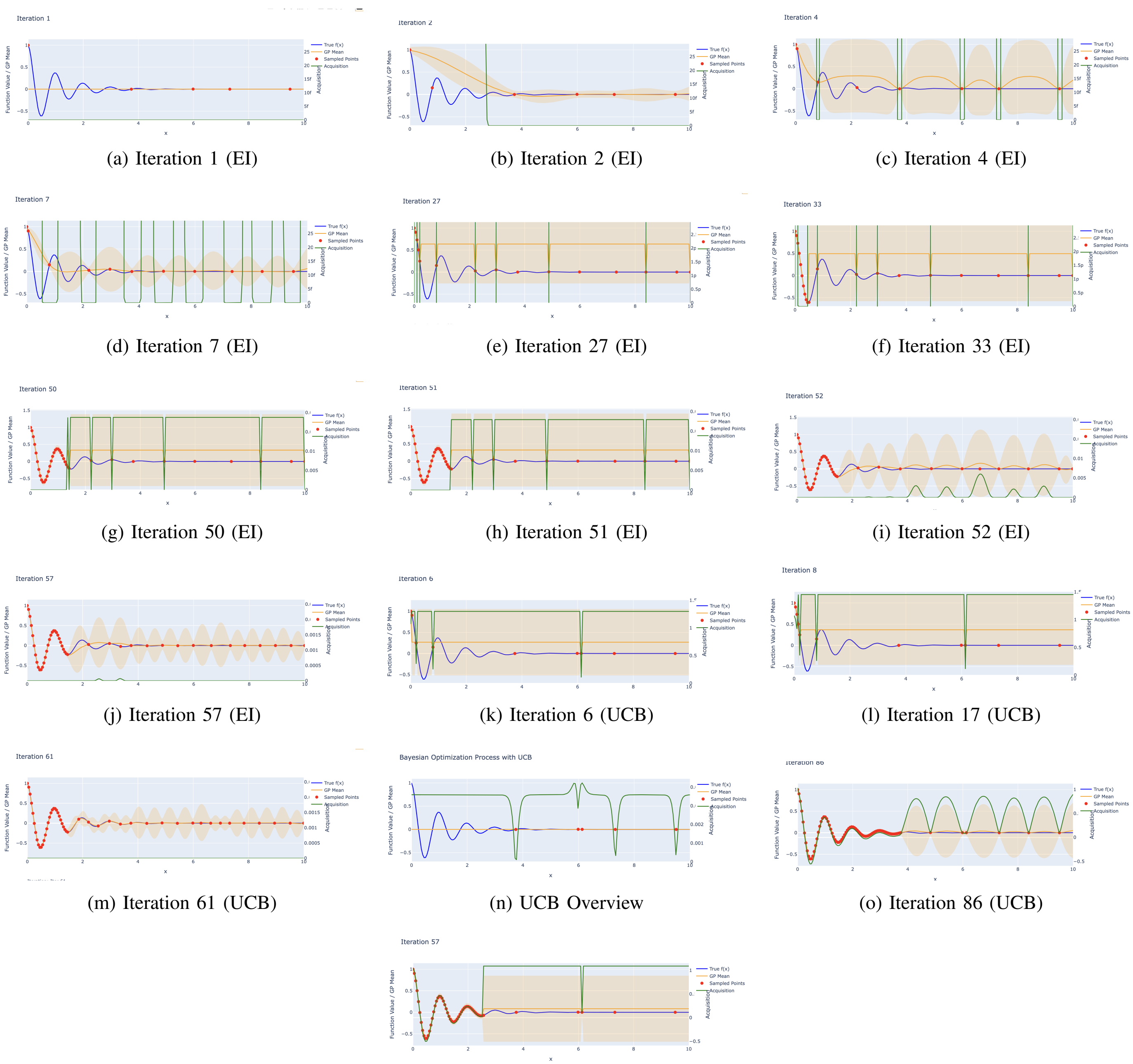
[4] S. Daulton, D. Eriksson, M. Balandat, and E. Bakshy, "Multi-objective bayesian optimization over high-dimensional search spaces," 2022. [Online]. Available: https://arxiv.org/abs/2109.10964

[5] T. Head, MechCoder, G. Louppe, J. Thomas, A. Gramfort, F. Pedregosa, and V. Niculae, "Scikit-optimize: Sequential model-based optimization with Scikit-Learn interface," https://scikit-optimize.github.io, 2018, version 0.8.1.

[6] M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy, "BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization," in *Advances in Neural Information Processing Systems 33*, 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/f5b1b89d98b7286673128a5fb112cb9a-Abstract.html

[7] E. Bakshy, L. Fan, G. Fierro, M. Balandat, B. Karrer, and D. Marthaler, "Ax: Adaptive experimentation platform," https://ax.dev, 2018, version 0.2.0.

APPENDIX



(a) Iteration 1 (EI)
(b) Iteration 2 (EI)
(c) Iteration 4 (EI)
(d) Iteration 7 (EI)
(e) Iteration 27 (EI)
(f) Iteration 33 (EI)
(g) Iteration 50 (EI)
(h) Iteration 51 (EI)
(i) Iteration 52 (EI)
(j) Iteration 57 (EI)
(k) Iteration 6 (UCB)
(l) Iteration 17 (UCB)
(m) Iteration 61 (UCB)
(n) UCB Overview
(o) Iteration 86 (UCB)
(p) Iteration 57 (UCB)

Fig. 5: Snapshots of EI and UCB acquisition functions (red dots = sampled points, green curve = acquisition) at key iterations.