SMAC3: A Versatile Bayesian Optimisation Package for Hyperparameter Optimisation

Pedro Ramos[1]

[1] ISAE Supaero, Data science (SDD)

February 2, 2026

Plan

1. The basics
   - Problem Statement
   - Bayesian optimisation
2. SMAC3 library
3. Optimisation for deep learning

Notation definition

- $\mathcal{A}$ = a machine learning algorithm with N hyperparameters
- Λ_n = the domain of the n-th hyperparameter
- $\mathbf{\Lambda} = \Lambda_1 \times \Lambda_2 \times \ldots \Lambda_N$ = the overall hyperparameter configuration space
- $\lambda \in \mathbf{\Lambda}$ = a vector of hyperparameters
- $\mathcal{A}_\lambda$ = $\mathcal{A}$ with its hyperparameters instantiated to λ

Given a dataset $\mathcal{D}$ and a user-defined loss function $\mathcal{L}$, the goal (Hutter et al., 2019) is to find:

Problem statement

Given a dataset $\mathcal{D}$ and a user-defined loss function $\mathcal{L}$, the goal (Hutter et al., 2019) is to find:

$$\lambda^* = \underset{\lambda \in \mathbf{\Lambda}}{\operatorname{argmin}} \mathbb{E}_{(D_{\text{train}}, D_{\text{valid}}) \sim \mathcal{D}} \mathbf{V}\left(\mathcal{L}, \mathcal{A}_\lambda, D_{\text{train}}, D_{\text{valid}}\right)$$

Where **V** is called the validation protocol (e.g. cross-validation error).

Why Bayesian optimisation?

1. Advanced optimization framework for the global optimization of expensive blackbox functions
2. Obtained state-of-the-art results in tuning **deep** neural networks for image classification, speech recognition and neural language modeling (Feurer & Hutter, 2019)
3. Wide applicability to different problem settings
4. Great sample efficiency

BO is an iterative algorithm with two key ingredients:

1. A probabilistic surrogate model of the objective function

   Example: Gaussian Process (GP) or Random Forest

2. An acquisition function to decide which point to evaluate next

   Example: Expected Improvement (EI)

   $$\mathbb{E}[\mathbb{I}(\lambda)] = \mathbb{E}\left[\max\left(f_{\min} - y, 0\right)\right]$$

BO in a nutshell

BO is an iterative algorithm with two key ingredients

Algorithm 1: Bayesian optimization

1: **for** $n = 1, 2, \dots,$ **do**

2: select new $\mathbf{x}_{n+1}$ by optimizing acquisition function α

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

3: query objective function to obtain y_{n+1}

4: augment data $\mathcal{D}_{n+1} = \{\mathcal{D}_n, (\mathbf{x}_{n+1}, y_{n+1})\}$

5: update statistical model

6: **end for**

Figure: (Shahriari et al., 2016)

Visual example

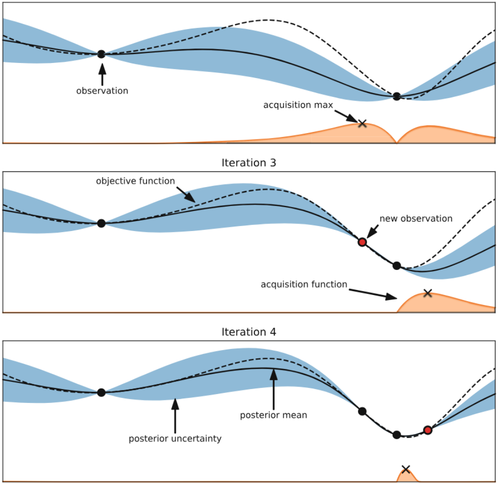

Figure: (Feurer & Hutter, 2019)

Plan

SMAC3 is a flexible open-source BO package that

- implements several BO approaches
- provides different facades, **hiding unnecessary complexity** and allowing easy usages
- can thus be robustly applied to **different HPO tasks**
- offers several acquisition functions, including EI per second (Snoek et al., 2012) **for evaluations with different runtimes**

The different facades

`SMAC3` has 4 pre-sets specifically designed to be efficient based on the characteristics of the use-case (Lindauer et al., 2022):

1. `SMAC4BB`: for Low-dimensional and Continuous Black-Box Functions
2. `SMAC4HPO`: for the combined algorithm selection and hyperparameter optimization problem (CASH)
3. `SMAC4MF`: for Expensive Tasks and Automated Deep Learning
4. `SMAC4AC`: for Algorithm Configuration

Plan

Successive Halving

According to Lin et al., 2025 ”Successive Halving (SH) allocates computational resources efficiently across many hyperparameter candidates by operating in a series of rungs. In each rung, a fraction of the candidates are discarded, typically based on their current intermediate performances. The surviving candidates are promoted to the next rung and provided with an exponentially larger resource budget.”

This approach allocates exponentially more resources to the best configurations while quickly wasting very little time on the bad ones.

SH's flaw is it requires the number of configurations (n) upfront. This creates a dilemma for a fixed total budget (B):

- **Too many configurations**: Each λ gets very little resource (B/n).A good configuration might be cut too early because it learns slowly (e.g., a low learning rate)
- **Too few configurations**: Each λ gets plenty of resources, but the search space is little explored and SH might miss the optimal setting entirely.

The fix: Hyperband

Hyperband (Li et al., 2017) **treats this trade-off as a grid search problem**. It runs the Successive Halving ”brackets” multiple times:

- **Aggressive Bracket** (n++,B): Designed to catch ”easy” configurations that perform well quickly.
- **Conservative Bracket** (n,B++): It behaves like standard Random Search. This protects against cases where good models need lots of time to distinguish themselves.
- **Intermediate Brackets**

Bayesian optimisation:

- is sample-efficient
- looks at the ”black box” function at full cost

Hyperband:

- uses ”early stopping” on cheap approximations (small budgets)
- relies on random search to pick configurations

BOHB (Falkner et al., 2018) uses **Hyperband to manage resources** (the budget) and **Bayesian Optimization to guide the search** for configurations.

1. **Hyperband Layer**: A standard Hyperband with Successive Halving on its brackets.
2. **Bayesian Optimization Layer**: In BOHB, Hyperband's random sampling is replaced by a model-based search (Bayesian Optimization).

Bayesian optimization and Hyperband

BOHB (Falkner et al., 2018) uses **Hyperband to manage resources** (the budget) and **Bayesian Optimization to guide the search** for configurations.

1. **Hyperband Layer**: A standard Hyperband with Successive Halving on its brackets.
2. **Bayesian Optimization Layer**: In BOHB, Hyperband's random sampling is replaced by a model-based search (Bayesian Optimization).

Building the Model (BO aquisition function) Falkner et al., 2018 BOHB builds a multidimensional Kernel Density Estimator, however, Lindauer et al., 2022 argue that their Random Forest Regressor is best.

Multiple Fidelities because `SMAC`

- keeps track of performance data across all budget levels (e.g., results from 10 epochs, 30 epochs, and 100 epochs)
- builds a model using only the observations from the largest budget that has enough data points to be reliable
- becomes more accurate as expensive, high-fidelity data becomes available

Falkner, S., Klein, A., & Hutter, F. (2018).Bohb: Robust and efficient hyperparameter optimization at scale. *Proceedings of the 35th International Conference on Machine Learning*, *80*, 1437–1446. http://proceedings.mlr.press/v80/falkner18a.html

Feurer, M., & Hutter, F. (2019). Hyperparameter optimization. In F. Hutter, L. Kotthoff, & J. Vanschoren (Eds.), *Automatic machine learning: Methods, systems, challenges* (pp. 3–38). Springer.

Hutter, F., Kotthoff, L., & Vanschoren, J. (Eds.). (2019). *Automatic machine learning: Methods, systems, challenges*. Springer.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., & Talwalkar, A. (2017).Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, *18*(185), 1–52. http://jmlr.org/papers/v18/16-558.html

Lin, J. A., Mayoraz, N., Rendle, S., Kuzmin, D., Praun, E., & Isik, B. (2025). Successive halving with learning curve prediction via latent kronecker gaussian processes. https://arxiv.org/abs/2508.14818

Lindauer, M., Eggensperger, K., Feurer, M., Biedenkapp, A., Deng, D., Benjamins, C., Ruhkopf, T., Sass, R., & Hutter, F. (2022).Smac3: A versatile bayesian optimization package for hyperparameter optimization. *Journal of Machine Learning Research*, *23*(54), 1–9. http://jmlr.org/papers/v23/21-0888.html

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016).Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, *104*(1), 148–175. https://doi.org/10.1109/JPROC.2015.2494218

Thank you for your attention.