PRACTICAL BAYESIAN OPTIMIZATION OF MACHINE LEARNING ALGORITHMS

Cole Richardson [1]

Abstract

As modern machine learning algorithms increase in complexity, the tuning of their hyperparameters also becomes a critical concern. Due to their complexity, today's machine learning models require greater computational resources and time to tune. Tuning of hyperparameters usually consists of empirical knowledge about the model, educated guessing and brute force searching. In this paper, we discuss Bayesian Optimization as an automated technique to tune our hyperparameters. We start with an overview of Gaussian Process Regression (GPR) models as this provides the Bayesian statistics for which our hyperparameter optimization routine is based. We show how changing the parameters of the GPR model itself can influence the Bayesian Optimization results. Finally, we close by using the developed knowledge to implement a Bayesian Optimization routine and empirically test against the Branin-Hoo function.

1. Introduction

Tuning of hyperparameters for computationally expensive learning models is too costly in terms of time and money to apply methods such as grid search or random search. Instead, a more algorithmically optimal solution such as Bayesian Optimization should be employed.

Bayesian Optimization is a probabilistic technique to find the global minimum/maximum of a multi-variable function f on a bounded domain $\mathcal{X}$. This function can be a standard function you would encounter in a Calculus course or it could be the hyperparameters for a learning model such as the regularization coefficient in a logistic regression model. We will mainly focus on simple to evaluate functions for ease of demonstration. Bayesian Optimization will use information from all previous objective function evaluations in order to chose the next best point to evaluate, in this sense, it is making optimal use of the information available.

The process of Bayesian Optimization requires us to select a GPR **prior distribution** as well as an **acquisition function**. The acquisition function is a utility function used to choose the next point in the objective functions domain to evaluate. We will also discuss how the parameters of the GPR prior itself can effect results. Finally, we simulate Bayesian Optimization using a few different acquisition functions.

2. Related Work

There are plenty of existing papers discussing the topic of Bayesian Optimization. We primarily make use of three papers in the following discussion. We reference (Wang, 2022) for a beginner friendly and intuitive understanding of Gaussian process regression modeling. For details and formulas related to tuning the parameters associated with the GPR model, see (Rasmussen). The goal of this paper is to present a subset of the information and results derived in (Snoek et al., 2012), as such, it may be referenced as well. This paper doesn't attempt to provide a novel contribution to the field, instead, we aim to understand the work in (Snoek et al., 2012) using clear explanations aided by graphics and code.

3. Method

3.1. Gaussian Process Regression

Let's start by developing an understanding for the underlying Gaussian process regression model. The purpose of GPR is to provide a non-linear model of our objective function f. This model is much less computationally expensive to evaluate than our real objective function and thus we will use it as a surrogate to our objective function f. The GPR is a distribution over functions parameterized primarily by a mean function and a covariance (kernel) function.

$$f(\mathbf{x}) = \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$$

The phrasing *distribution over functions* is used because

[1]Department of Electrical and Computer Engineering, University of Purdue, West Lafayette, Indiana, United States. Correspondence to: Cole Richardson <richa553@purdue.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

the GPR model is a collection of infinitely many random variables and has a continuous domain. In other words, we can evaluate our GPR model with any input point (Williams, 2006). We will first define the GPR prior which does not depend on any training data and requires us to specify the mean and covariance functions. We will choose the zero mean function because once our model is updated with training data, the mean will likely no longer be zero. The most important decision is the covariance function. We will choose the ARD Matérn 5/2 kernel (Snoek et al., 2012):

$$K_{\mathbf{M52}}(\mathbf{x},\mathbf{x}') = \theta_0\Big(1+\sqrt{5r^2(\mathbf{x},\mathbf{x}')} + \frac{5}{3}r^2(\mathbf{x},\mathbf{x}')\Big)exp\Big\{-\sqrt{5r^2(\mathbf{x},\mathbf{x}')}\Big\}\Big\} \tag{1}$$

where,

$$r^2(\mathbf{x},\mathbf{x}') = \sum_{d=1}^{D}(x_d = x'_d)^2/\theta_d^2 \tag{2}$$

As seen, the kernel itself has hyperparameters (θ_0 and $\theta_{1:D}$) that we will need to learn. However, these hyperparameters are not the same ones being optimized by the Bayesian Optimization procedure, instead these will be learned when fitting the GPR model to the training data. To learn these hyperparameters, we will need to optimize the *log marginal likelihood* (Rasmussen). The result of this optimization will be the updated GPR posterior distribution that we will use as the surrogate for our objective function.

3.2. Acquisition Function

Given the set of real observations of f and the samples of the GPR posterior distribution, how do we select where to evaluate f next? The approach in Bayesian optimization is to design an acquisition function $a(x)$. The acquisition function is typically a computationally inexpensive function that can be evaluated at a given point and represents how desirable evaluating f at x is expected to be for the optimization problem. We will then optimize the acquisition function to select the location of the next $\mathbf{x}$. With an acquisition function, we have replaced our original optimization problem with another optimization problem; however, the acquisition function is much cheaper to evaluate than our objective function f.

The acquisition function denoted as $a : \mathcal{X} \rightarrow \mathcal{R}^+$ is used to determine which point in $\mathcal{X}$ should be evaluated next via the optimization $\mathbf{x}_{best} = \text{argmax}_{\mathbf{x}} a(\mathbf{x})$ (Snoek et al., 2012). There are a few different acquisition functions we will now consider. We will use the following notations to define the acquisition functions:

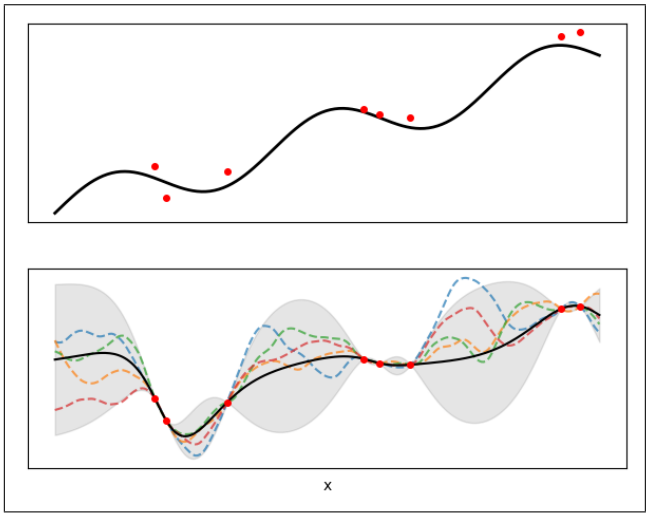

Figure 1. Fitting objective function with GPR model
The top plot shows the simple 1-D objective function and the eight noisy samples that were used to train/fit the GPR model. The bottom plot shows the posterior samples/predictions drawn from the GPR model (black middle line is the mean for each sample, lighter grey areas show the 95% confidence interval for each sample). Notice how the uncertainty decreases near the points of observation. The colored lines represent other sets of samples from the GPR model. Overall, the GPR model is a fair representation of our true objective function and will only improve as more samples are added.

$$\begin{aligned}
\mu(\mathbf{x}) &\rightarrow \text{GPR mean function}\\
\sigma^2(\mathbf{x}) &\rightarrow \text{GPR covariance function}\\
f_{best} &\rightarrow \text{current best (max/min)}\\
&\quad\text{of objective function}\\
\Phi(\cdot) &\rightarrow \text{standard normal CDF}\\
\phi(\cdot) &\rightarrow \text{standard normal PDF}\\
\gamma(\mathbf{x}) &= \frac{f_{best}-\mu(\mathbf{x})}{\sigma^2(\mathbf{x})}
\end{aligned} \tag{3}$$

3.2.1. Expected Improvement (EI)

$$a_{EI}(\mathbf{x}) = \sigma(\mathbf{x})(\gamma(\mathbf{x})\phi(\mathbf{x})) + \mathcal{N}(\gamma(\mathbf{x};0,1) \tag{4}$$

3.2.2. Probability of Improvement (PI)

$$a_{PI}(\mathbf{x}) = \phi(\gamma(\mathbf{x})) \tag{5}$$

3.2.3. Upper Confidence Bound (UCB)

$$a_{UCB}(\mathbf{x}) = \mu(\mathbf{x}) - \kappa\sigma(\mathbf{x}) \tag{6}$$

We see that the UCB acquisition function has its own tunable parameter κ, for more information about these acquisition functions and their properties/derivations see (Schulz et al., 2018) and (Snoek et al., 2012). In the Experiment section below, we will present some empirical performance data for each of the acquisition functions.

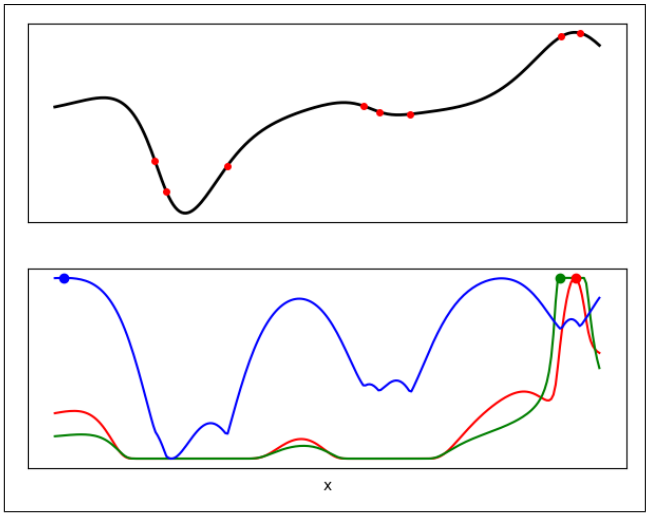

Figure 2. Visualization of Acquisition Functions
The top plot is the same posterior mean plot as the bottom plot in figure 1. The bottom plot is the evaluation of all three acquisition functions we are considering (RED - Expected Improvement, GREEN - Probability of Improvement, BLUE - Upper Confidence Bound). The maximum of the acquisition functions are also shown, these are important because the x-value associated with them represents the next x point that should used to evaluate the Bayesian Optimization objective function. As seen, all three acquisition functions are predicting a different x-value.

3.3. Bayesian Optimizer

Now we have all the knowledge necessary to develop our own Bayesian Optimization routine. The following provides a high level description of the algorithm.

- Perform a few evaluations of the objection function to serve as initial data for fitting the GPR model.
- Define the GPR model, with special emphasis on it's kernel.
- Begin iterations (desired number of evaluations):

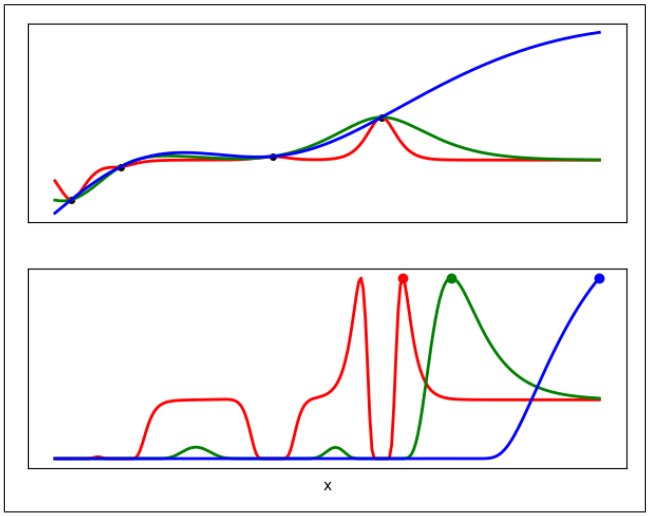

Figure 3. EI Acquisition with Varying GPR Hyperparameters
The top plot is the same posterior mean plot as in figures 1 and 2; however, there are fewer samples so it is a worse approximation of the true objective function. Also, there are there different curves because we are varying the hyperparameters of the GPR model used to predict these curves. Specifically, we are varying the length-scale parameter in the Matérn 5/2 kernel (θ_d). The bottom plot is the Expected Improvement acquisition function evaluated for each function value in the top plot. As seen, varying the hyperparameters has caused the EI acquisition function to make different suggestions about what x-value to evaluate next.

  - Fit the GPR model to the existing evaluations
  - Evaluate the GPR posterior model at many different x-values of your choosing.
  - Use the GPR posterior samples as inputs to the acquisition function.
  - Find the x-value associated with the maximum of the acquisition function evaluations.
  - Evaluate the objective function at the x-value from the previous step.
  - Add the objective function result to the GPR training data list

- Find the maximum/minimum from all evaluations of objective function.

4. Experiment

To measure the performance of our Bayesian Optimization algorithm, we are going to test it's ability to find the minimum of the Branin-Hoo function. As seen in figure 4, the Branin-Hoo function is a multi-modal 2-D function. Figure 5 show the performance of the Bayesian Optimization

against the Branin-Hoo function utilizing all the different acquisition functions we introduced.

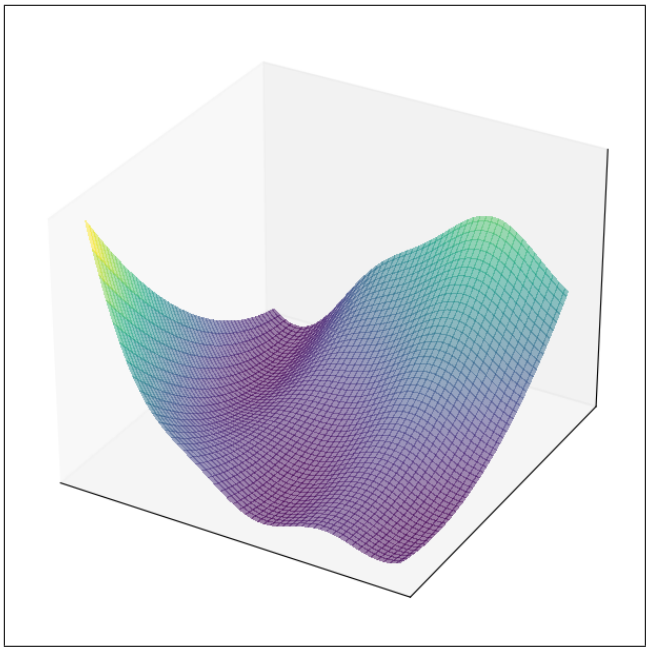

Figure 4. Branin-Hoo Surface Plot
The Branin-Hoo function is a multi-modal 2-D function that is often used when testing optimization algorithms. It's domain is $x \epsilon R^2$ where $0 \leq x1 \leq 15$ and $5 \leq x2 \leq 15$.

5. Conclusion

In this paper we presented an overview of Bayesian Optimization and showed that it can be used to tune the hyperparameters of a computationally expensive machine learning algorithm. We provided detailed descriptions of the two major topics underlying Bayesian Optimization, Gaussian Process Regression models and using Acquisition functions to serve as surrogates for the true objective function. This paper doesn't cover a fully Bayesian approach as that would require using some sort of slice sampling algorithm such as Markov Chain Monte Carlo (MCMC) to be used when sampling the GPR posterior samples instead of using a log-marginal likelihood point estimate. A fully Bayesian approach like this is covered in (Snoek et al., 2012) and would be a great way to extend this paper in the future. Additionally, the algorithms developed in this paper were all implemented using single-core processing, in a production environment, machine learning algorithms would be developed and trained on state-of-the-art hardware. As such, a treatment of how to extend the Bayesian Optimization procedure to multi-core would be required to most efficiently use of the hardware.

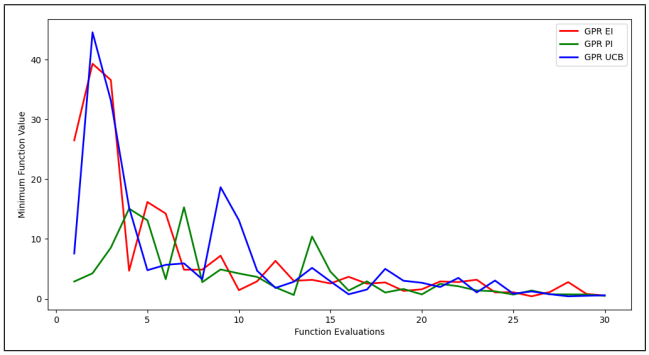

Figure 5. Bayesian Optimization performance (Branin-Hoo)
This figure presents the performance of the Bayesian Optimization procedure on the Branin-Hoo function. As the number of iterations increase, the minimum function value trends to 0 as expected.

Software and Data

Code used to generate figures

Code for simple 1-D optimization with debug plots

Code for Bayesian Optimization algorithm

Acknowledgements

I would like to acknowledge Professor Qi Guo and the teaching staff of ECE 524 Spring 2023 for teaching me the necessary prerequisites for this paper and providing a great first introduction to Machine Learning.

References

Rasmussen, C. E. Gaussian processes in machine learning.

Schulz, E., Speekenbrink, M., and Krause, A. A tutorial on gaussian process regression: Modelling, exploring, and exploiting functions. *Journal of Mathematical Psychology*, 85:1–16, 2018.

Snoek, J., Larochelle, H., and Adams, R. P. Practical bayesian optimization of machine learning algorithms, 2012.

Wang, J. An intuitive tutorial to gaussian processes regression, 2022.

Williams, C. E. R. . C. K. I. *Gaussian Processes for Machine Learning*. MIT Press, 2006.