Bayesian Optimization for Machine Learning

Math 693A: Numerical Optimization
Project Proposal

Pradeep Singh

Computational Science Research Center,
San Diego State University, CA

Abstract

This project explores Bayesian optimization techniques for hyperparameter tuning in machine learning algorithms and will compare it with different methods like: manual search, grid search, random search. Goal of this project is twofold: 1) To study how bayesian optimization can be used in hyperparameter tuning in order to improve the current methods, and 2) Comprehensive analysis of hyperparameter optimization algorithms in Machine Learning.

Introduction

All the machine learning algorithms involve optimization of a loss function in order to tune the learning parameters and model hyperparameters that return the best performance. The most common methods are: manual search, grid search, random search, etc. These methods are inefficient because they do not choose the next hyperparameters to evaluate based on previous results, as a result, they often spend a significant amount of time evaluating "bad" hyperparameters. This problem could be solved using Bayesian methods, as they takes into account the previous results and make an informed guess on top of that for the next hyperparameter value.

Bayesian Optimzation

Bayesian optimization falls in a class of optimization algorithms called *sequential model-based optimization (SMBO)* algorithms [E Brochu, 2010]. These algorithms use previous observations of the loss function, to determine the next (optimal) point to sample function for.

Bayesian approach could be used to tune hyperparameters; it keep track of past evaluation results which it use to form a probabilistic model mapping hyperparameters to a probability of a score on the objective function. This model is called a "surrogate" for the objective function and is represented as:

$$p(y|x) \quad \text{OR} \quad p(score|hyperparameters)$$

The surrogate is much easier to optimize than the objective function. Bayesian methods work by finding the next set of hyperparameters to evaluate on the actual objective function by selecting hyperparameters that perform best on the surrogate function. The aim of Bayesian reasoning is to become "less wrong" with more data which these approaches do by continually updating the surrogate probability model after each evaluation of the objective function.

In other words:

1. Build a surrogate probability model of the objective function.
2. Find the hyperparameters that perform best on the surrogate

3. Apply these hyperparameters to the true objective function.

4. Update the surrogate model incorporating the new results.

5. Repeat steps 2–4 until max iterations or time is reached.

At a high-level, Bayesian optimization methods are efficient because they choose the next hyperparameters in an informed manner. The basic idea is: spend a little more time selecting the next hyperparameters in order to make fewer calls to the objective function. In practice, the time spent selecting the next hyperparameters is inconsequential compared to the time spent in the objective function. By evaluating hyperparameters that appear more promising from past results, Bayesian methods can find better model settings than random search in fewer iterations.

What?

I plan to work on a machine learning algorithm using different hyperparameter tuning methods: Bayesian Optimization, Gird Search and Random Search. The idea is to compare how Bayesian Optimization performs in comparison to different methods.

Why?

Bayesian optimization methods are more efficient as compare to other hyperparameter tuning methods used in machine (deep) learning. They choose the next hyperparameters in an informed manner, unlike other methods. The basic idea is: spend a little more time selecting the next hyperparameters in order to make fewer calls to the objective function. In practice, the time spent selecting the next hyperparameters is inconsequential compared to the time spent in the objective function. By evaluating hyperparameters that appear more promising from past results, Bayesian methods can find better model settings than random search or grid search in fewer iterations.

How?

This project will be implemented in Python. I will be using some python packages like NumPy, SciPy, and Scikit-learn to code up the algorithms. I'm currently in the process to nail down the dataset, once that is done will decide which machine algorithm will work best on it. The aim of this project is not to do machine learning but to study and implement Bayesian optimization in any given machine learning model, so I'll work on a smaller dataset so that I can focus more on optimization part and not on the machine learning.

References

- (E Brochu, 2010), A Tutorial on Bayesian Optimization of Expensive Cost Functions. arXiv:1012.2599.
- (J. Bergstra, 2011), Algorithms for hyper-parameter optimization., Advances in Neural Information Processing Systems.
- (T. Huijskens, 2018), Bayesian optimization with scikit-learn.