Hyperparameter optimization

1 Grid Search

Grid Search performs an exhaustive searching through a manually specified subset of the hyperparameter space defined in the searchspace file.

1.1 Advantages and disadvantages

It will find the best parameter.
Slow, High cost. Local optimization is not necessarily globally optimal.
bad on high spaces

1.2 Suggested scenario

low spaces, small data sets. Experienced expert.

1.3 Reference

sklearn.model_selection.GridSearchCV

2 Random Search

In Random Search for Hyper-Parameter Optimization show that Random Search might be surprisingly simple and effective. We suggest that we could use Random Search as the baseline when we have no knowledge about the prior distribution of hyper-parameters.

2.1 Advantages and disadvantages

good on high spaces
Give better results in less iterations
it doesn't guarantee to find the best hyperparameters.

2.2 Suggested scenario

Random search is suggested when each trial does not take too long (e.g., each trial can be completed very soon, or early stopped by assessor quickly), and you have enough computation resource. Or you want to uniformly explore the search space. Random Search could be considered as baseline of search algorithm.

2.3 Reference

hyperopt:`https://github.com/hyperopt/hyperopt`

3 Bayesian Optimization

Bayesian Optimization (BayesOpt) is an approach to solving challenging optimization problems. It uses a machine learning technique (Gaussian process regression) to estimate the objective function based on past evaluations, and then uses an acquisition function to decide where to sample next. It typically takes longer than other optimization methods to decide where to sample, but uses fewer evaluations to find good solutions.

3.1 Suggested scenario

- Each evaluation of the objective function takes a long time to evaluate.In most BayesOpt applications we expect to evaluate the objective between 50 and 1000 times.
- The objective function is a continuous function of the inputs.
- The objective function lacks other special structure, such as convexity or concavity, that could be used by an optimization method purpose-built for problems with that structure. We say that the objective is a ”black box”.

- evaluations are noisy

3.2 Reference

Paper:`https://arxiv.org/pdf/1807.02811.pdf`
tutorial:`http://mcqmc2016.stanford.edu/Frazier-Peter.pdf`
Spearmint: `https://github.com/HIPS/Spearmint`
Cornell-MOE: `https://github.com/wujian16/Cornell-MOE`

4 BOHB

4.1 Suggested scenario

Modern deep learning methods are very sensitive to many hyperparameters, and, due to the long training times of state-of-the-art models, vanilla Bayesian hyperparameter optimization is typically computationally infeasible. On the other hand, bandit-based configuration evaluation approaches based on random search lack guidance and do not converge to the best configurations as quickly. We propose to combine the benefits of both Bayesian optimization and bandit-based methods, in order to achieve the best of both worlds: strong anytime performance and fast convergence to optimal configurations.

HpBandSter:`https://github.com/automl/HpBandSter`

5 SMAC

5.1 Suggested scenario

Similar to TPE, SMAC is also a black-box tuner which can be tried in various scenarios, and is suggested when computation resource is limited. It is optimized for discrete hyperparameters, thus, suggested when most of your hyperparameters are discrete.

SMAC3:`https://github.com/automl/SMAC3`

6 TPE

6.1 Suggested scenario

TPE, as a black-box optimization, can be used in various scenarios and shows good performance in general. Especially when you have limited computation resource and can only try a small number of trials. From a large amount of experiments, we could found that TPE is far better than Random Search.

hyperopt:`https://github.com/hyperopt/hyperopt`

hyperopt.tpe.suggest

7 Anneal

7.1 Suggested scenario

Anneal is suggested when each trial does not take too long, and you have enough computation resource(almost same with Random Search). Or the variables in search space could be sample from some prior distribution.

hyperopt:`https://github.com/hyperopt/hyperopt`

hyperopt.anneal.suggest

8 Other references

Introduction to Tuner in nni:
`https://github.com/microsoft/nni/blob/master/docs/en_US/Builtin_Tuner.md#TPE`
AutoML: `https://www.ml4aad.org/automl/`