Oríon algo robo Documentation

Release v0.1.1.post3

Epistímio

Jun 29, 2022

WELCOME

This wrapper provides access through Oríon to several Bayesian optimization algorithms in the library RoBO.

This `orion.algo` plugin was generated with Cookiecutter along with @Epistimio's cookiecutter-orion.algo template.

CHAPTER

ONE

INSTALLATION

The RoBO wrapper is currently only supported on Linux.

Before installing RoBO, make sure you have `libeigen` and `swig` installed. On ubuntu, you can install them with `apt-get`, like so:

```
$ sudo apt-get install libeigen3-dev swig
```

The RoBO wrapper can then be installed, either from source, like so:

```
$ pip install git+https://github.com/Epistimio/orion.algo.robo
```

Or from PyPI:

```
$ pip install orion.algo.robo
```

CHAPTER

TWO

CONTRIBUTE OR ASK

Do you have a question or issues? Do you want to report a bug or suggest a feature? Name it! Please contact us by opening an issue in our repository below and checkout Oríon's contribution guidelines:

- Issue Tracker: https://github.com/Epistimio/orion.algo.robo/issues
- Source Code: https://github.com/Epistimio/orion.algo.robo

Start by starring and forking our Github repo!

Thanks for the support!

CHAPTER

THREE

CITATION

If you use this wrapper for your publications, please cite both RoBO and Oríon. Please also cite the papers of the algorithms you used, such as DNGO or BOHAMIANN. See the documentation of the algorithms to find corresponding original papers.

CHAPTER

FOUR

LICENSE

Distributed under the terms of the BSD-3-Clause license, `orion.algo.robo` is free and open source software.

4.1 RoBO Gaussian Process

```
experiment:
    algorithms:
        RoBO_GP:
            seed: 0
            n_initial_points: 20
            maximizer: 'random'
            acquisition_func: 'log_ei'
            normalize_input: True
            normalize_output: False
```

4.2 RoBO Gaussian Process with MCMC

```
experiment:
    algorithms:
        RoBO_GP_MCMC:
            seed: 0
            n_initial_points: 20
            maximizer: 'random'
            acquisition_func: 'log_ei'
            normalize_input: True
            normalize_output: False
            chain_length: 2000
            burnin_steps: 2000
```

4.3 RoBO Random Forest

```
experiment:
    algorithms:
        RoBO_RandomForest:
            seed: 0
            n_initial_points: 20
            maximizer: 'random'
            acquisition_func: 'log_ei'
            num_trees: 30
            do_bootstrapping: True
            n_points_per_tree: 0
            compute_oob_error: False
            return_total_variance: True
```

4.4 RoBO DNGO

```
experiment:
    algorithms:
        RoBO_DNGO:
            seed: 0
            n_initial_points: 20
            maximizer: 'random'
            acquisition_func: 'log_ei'
            normalize_input: True
            normalize_output: False
            chain_length: 2000
            burnin_steps: 2000
            batch_size: 10
            num_epochs: 500
            learning_rate: 1e-2
            adapt_epoch: 5000
```

4.5 RoBO BOHAMIANN

```
experiment:
    algorithms:
        RoBO_BOHAMIANN:
            seed: 0
            n_initial_points: 20
            maximizer: 'random'
            acquisition_func: 'log_ei'
            normalize_input: True
            normalize_output: False
            burnin_steps: 2000
            sampling_method: "adaptive_sghmc"
            use_double_precision: True
            num_steps: null
```

(continues on next page)

(continued from previous page)

```
            keep_every: 100
            learning_rate: 1e-2
            batch_size: 20
            epsilon: 1e-10
            mdecay: 0.05
            verbose: False
```