25 November 2019

BEAM PARAMETER OPTIMIZATION

INTRODUCTION

Calibrating BEAM (https://github.com/LBNL-UCB-STI/beam) simulation outputs by tuning parameters to achieve real world targets (for example: mode shares, average car travel time, pool match, deadheading share etc.) in reduced times.

Step 1

Build a calibration API (existing: https://github.com/LBNL-UCB-STI/beam/tree/develop/src/main/scala/beam/calibration) around BEAM for hyperparameter optimization development i.e., an interface for Python development around native Java/ Scala codebase.

Step 2

Code experiments compatible with the chosen formulations* and BEAM.

* formulations to choose from:

- Microsoft Neural Network Intelligence (https://www.microsoft.com/en-us/research/project/neural-network-intelligence/)
- Ray Tune from riselab UC Berkeley (https://github.com/ray-project/ray/tree/master/python/ray/tune)
- Hyperopt (https://github.com/hyperopt/hyperopt)

Step 3

Investigate the feasibility of own algorithm(s)/ improvements to the existing formulations.

FORMULATIONS COMPARISON

MSR NNI/ Hyperopt (Tree Parzen Estimator)

Established: November 2017, support indefinitely

Compatible Frameworks: Tensorflow, Keras, Pytorch, Sklearn

Existing Tuning Algorithms: TPE, Random Search, Anneal, Naïve Evolution, SMAC, Batch Tuner, Grid Search, Hyperband, Network Morphism, Metis Tuner, BOHB, GP Tuner, PPO Tuner

Extension/ Integration: Can easily submit proposals for new features suitable for BEAM use case by raising a PR.

Parallelizing Sequential Algorithm TPE:
Existing implementations: SMBO (Sequential Model Based Optimization) which narrows down the search space based on previous results, where input is (parameters and loss) and output is parameter suggestion for the next step for a given search space & objective function.
Objective is to maximize expected improvement, which is:

$$EI_{yk}(x) = \int_{-\infty}^{\infty} max\left(y^k - y, 0\right) p(y|x) dy$$

Where x is the hyperparameter, y^k is target performance and y is the loss, where the algorithm returns:

$$argmin_{(x)} \left[\frac{g(x)}{q(x)}\right]$$

Where g and q are two random distribution of split (parameter – loss) pairs which are assumed to be good and bad distribution respectively.
This formulation also presents additional deeper investigations like Kriging Believer Strategy, Constant Liar Strategy etc.

System requirements:

Requires NVIDIA GeForce GTX 460/ 660/ better (https://www.geforce.com/hardware/desktop-gpus/geforce-gtx-660)

Test run on my machine:

Since I own a ThinkPad workstation 541 with NVIDIA Quadro K1100M 2G (lower spec than required), my build is unstable:

```
(calib) berkeleylab@KiranCHHATRE:~/Misc_code/Comparison/NNI$ nnictl create --
config nni/examples/trials/mnist/config.yml
INFO: expand searchSpacePath: search_space.json to
/home/berkeleylab/Misc_code/Comparison/NNI/nni/examples/trials/mnist/search_sp
ace.json
INFO: expand codeDir: . to
/home/berkeleylab/Misc_code/Comparison/NNI/nni/examples/trials/mnist/.
INFO: Starting restful server...
ERROR: Restful server start failed!
INFO:  Stdout:
-----------------------------------------------------------------------
                Experiment start time 2019-11-24 18:29:09
-----------------------------------------------------------------------
INFO:  Stderr:
-----------------------------------------------------------------------
                Experiment start time 2019-11-24 18:29:09
-----------------------------------------------------------------------
(calib) berkeleylab@KiranCHHATRE:~/Misc_code/Comparison/NNI$
```

Sometimes the build is successful as follows:

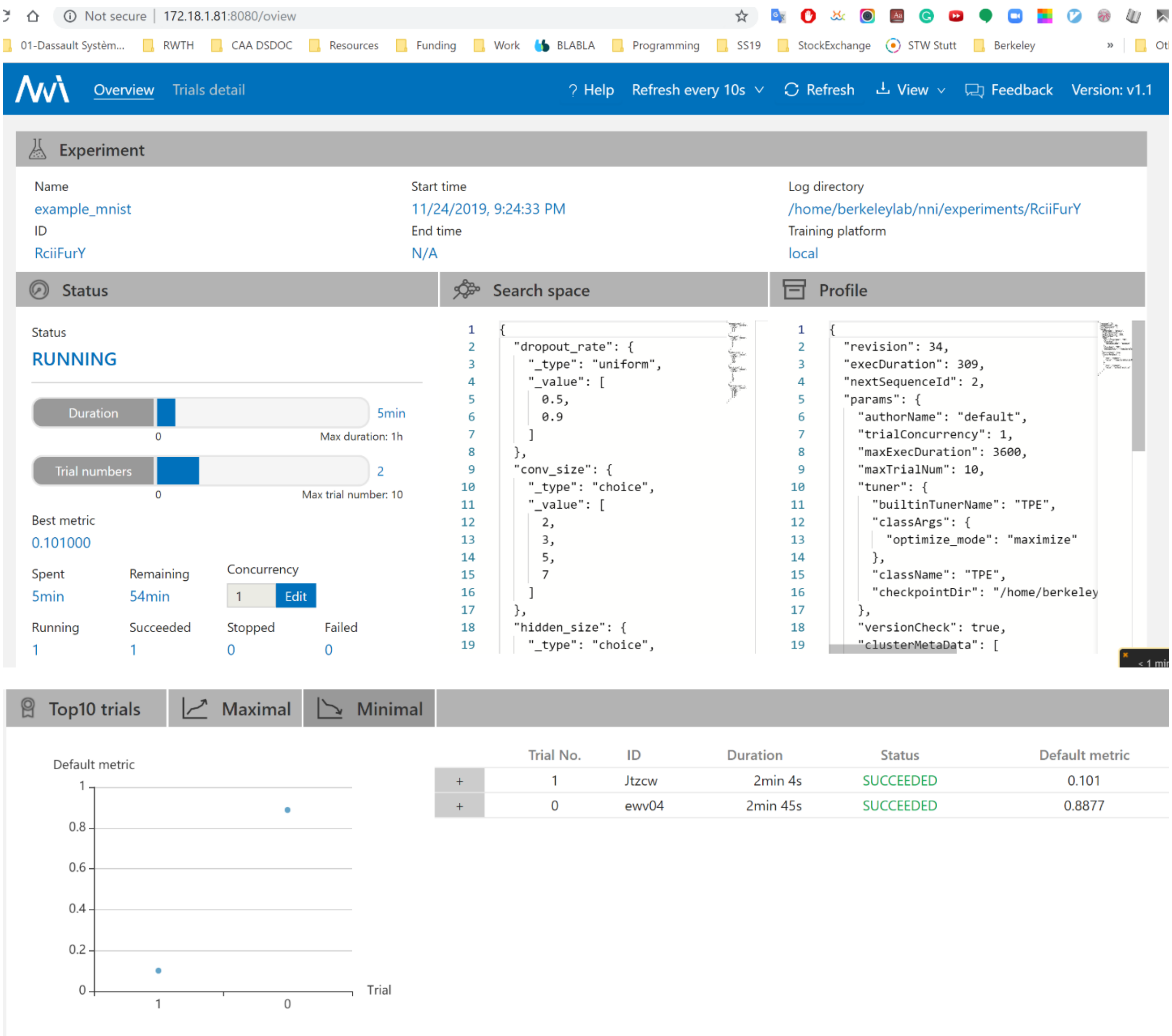
```
(calib) berkeleylab@KiranCHHATRE:~/Misc_code/Comparison/NNI$ nnictl create --
config nni/examples/trials/mnist/config.yml
INFO: expand searchSpacePath: search_space.json to
/home/berkeleylab/Misc_code/Comparison/NNI/nni/examples/trials/mnist/search_sp
ace.json
INFO: expand codeDir: . to
/home/berkeleylab/Misc_code/Comparison/NNI/nni/examples/trials/mnist/.
INFO: Starting restful server...
INFO: Successfully started Restful server!
INFO: Setting local config...
INFO: Successfully set local config!
INFO: Starting experiment...
INFO: Successfully started experiment!
------------------------------------------------------------------------------
------
The experiment id is RciiFurY
The Web UI urls are: http://169.254.114.5:8080   http://192.168.56.1:8080
http://172.18.1.81:8080   http://127.0.0.1:8080   http://10.142.69.147:8080
http://169.254.225.113:8080   http://169.254.86.147:8080
------------------------------------------------------------------------------
------

You can use these commands to get more information about the experiment
------------------------------------------------------------------------------
------
         commands                       description
1. nnictl experiment show        show the information of experiments
2. nnictl trial ls               list all of trial jobs
3. nnictl top                    monitor the status of running experiments
4. nnictl log stderr             show stderr log content
5. nnictl log stdout             show stdout log content
6. nnictl stop                   stop an experiment
7. nnictl trial kill             kill a trial job by id
8. nnictl --help                 get help information about nnictl
------------------------------------------------------------------------------
------
```

```
Command reference document
https://nni.readthedocs.io/en/latest/Tutorial/Nnictl.html
------------------------------------------------------------------------------
------

(calib) berkeleylab@KiranCHHATRE:~/Misc_code/Comparison/NNI$
```

This also have a web interface as follows:



for monitoring the evolution in the search vector space. PS. The test run is on MNIST computer vision neural network problem, only for comparison study purpose.