Qualification Task Final Presentation

Hyperparameter Optimization on the Grid

Alkaid Cheng

University of Wisconsin-Madison

chi.lung.cheng@cern.ch

September 17, 2020

Contents

Introduction and Motivation

- This project serves to provide a workflow for users to submit jobs for hyperparameter optimization (HPO) on ATLAS grid sites (with GPU)
- If you had one of these problems when trying to do hyperparameter tuning using the ATLAS grid resources, you come to the right talk:
  - "How to set up the environment so that my scripts will run on the grid?"
  - "What are the grid sites that I can use?"
  - "What command line options should I include when using 'prun' to submit grid jobs?"
  - "Can I retrieve the results without going through the trouble of downloading the output from PanDA?"
  - "Can I run multiple search points concurrently to save some time?"
  - "Is there an easier way do hyperparameter tuning without drastically modifying my training script?"
- A python package called hpogrid is developed to help users solve the above problems with just a few simple command lines.
- Git repository for this project and its documentation can be found here
- Previous talks:
  - First Talk
  - Second Talk
  - Third Talk

Grid Computing Resource in ATLAS

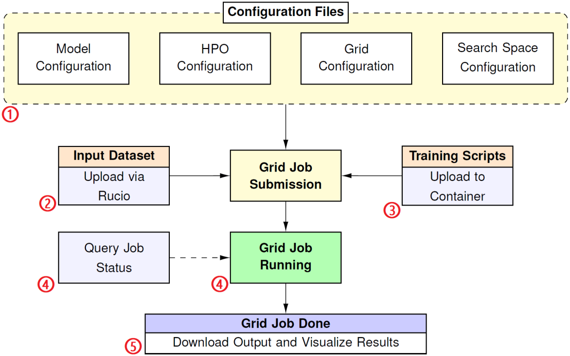
- Currently available GPU sites

Site Name	N GPUs	Max Memory (GB)	Max Run Time (Hr)	GPU per job
ANALY_MANC_GPU_TEST	6,4	12	72	1
ANALY_MWT2_GPU	8	11	120	**8**
DESY-HH_GPU	?	24	60	1
ANALY_BNL_GPU_ARC	12	24	24	1
ANALY_QMUL_GPU_TEST	2,4	12	96	1
ANALY_SLAC_GPU	326	9	2	1
ANALY_INFN-T1_GPU	2	9	60	1
ANALY_OU_OSCER_GPU_TEST	80	2	0.5	?
ANALY_LRZ_GPU	?	6	48	?
ANALY_CSCS-HPC_GPU	?	6	24	?

Sites in bold were tested to be working.

- Refer to this link for more details
- The detailed configurations of ATLAS grid sites can be found in /cvmfs/atlas.cern.ch/repo/sw/local/etc/agis_schedconf.json

The HPOGrid Workflow



1 Prepare a configuration file for the hyperparameter optimization task.

2 Upload the input dataset via rucio which will be retrieved by the grid site when the hyperparameter optimization task is executed.

3 Adapt the training script(s) to conform with the format required by the hyperparameter optimization library (Ray Tune).

4 Submit the hyperparamter optimization task and monitor its progress.

5 Retrieve the hyperparamter optimization results after completion. The results can be output into various formats supported by the hpogrid tool for visualization.

HPOGrid Features: The configuration file

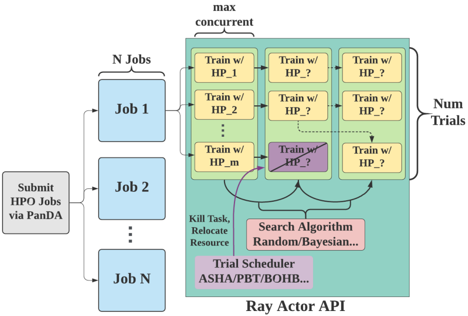
- Question: "Is there an easier way do hyperparameter tuning without drastically modifying my training script?"
- Answer: In the HPOGrid workflow, an hpo task is controlled by a configuration file (in json or yaml format) that looks something like this (check actual file here)

```yaml
project_name: RNN_TRT
scripts_path: /afs/cern.ch/work/u/user/scripts/RNN_TRT
model_config:
  script: train.py
  model: RNN_TRT
  param:
    num_epochs: 100
    verbose: 0
hpo_config:
  algorithm: random
  metric: accuracy
  mode: max
  num_trials: 4
  max_concurrent: 4
grid_config:
  site: ANALY_MANC_GPU_TEST,ANALY_MWT2_GPU,ANALY_BNL_GPU_ARC
  inDS: user.chlcheng:user.chlcheng.RNNTRT.v04.dataset
search_space:
  batch_size:
    method: categorical
    dimension:
      categories:
      - 128
      - 256
      - 512
      - 1024
      - 2048
      - 4096
      - 8192
```

```yaml
lr:
  method: loguniform
  dimension:
    low: 1.0e-05
    high: 0.1
lstm_hidden_size:
  method: categorical
  dimension:
    categories:
    - 32
    - 64
    - 128
    - 256
dense_size:
  method: categorical
  dimension:
    categories:
    - 32
    - 64
    - 128
    - 256
dense_activation:
  method: categorical
  dimension:
    categories:
    - relu
    - elu
    - softmax
    - sigmoid
    - tanh
```

HPOGrid Features: Hyperparameter Tuning using the Ray Tune library

- Question: ”Is there an easier way do hyperparameter tuning without drastically modifying my training script?”
- Question: ”Can I run multiple search points concurrently to save some time?”
- Answer: In the HPOGrid workflow, the Ray tune library is responsible for running the hyperparamter optimization task. It serves to provide support multiple hyperaparameter optimization libraries such as hyperopt, skopt and nevergrad and optimize the process by **parallelizing the tasks** and **kill tasks that have poor results**.
- With the configuration file, users can submit HPO tasks to the grid that runs through the following



HPOGrid Features: The Docker container

- Question: "How to set up the environment so that my scripts will run on the grid?"
- Answer: A docker container has been developed that will contain most of the essential machine learning and hyperparameter tuning libraires, such as
  tensorflow, pytorch, sklearn, keras, xgboost, skopt, hyperopt, nevergrad, ax and ROOT
- In the HPOGrid workflow, this container is used by default
- To test the container inside lxplus, one can try:

```
$singularity shell /cvmfs/unpacked.cern.ch/gitlab-registry.cern.ch
    /aml/hyperparameter-optimization/alkaid-qt/hpogrid:latest
```

- To test inside your pc:

```
$docker exec -t --rm gitlab-registry.cern.ch/aml/hyperparameter-
    optimization/alkaid-qt/hpogrid:latest
```

HPOGrid Features: Running HPO Tasks

- Question: "What command line options should I include when using 'prun' to submit grid jobs?"
- Answer: The HPOGrid package has implemented a command line interface for users to perform various steps of the workflow
- For example, to run an HPO task locally with the configuration file:

```
$ hpogrid run <configuration_file>
```

- Or to submit an HPO task to the grid (this will handle the command line options in 'prun'):

```
$ hpogrid submit <configuration_file>
```

An example output will be

```
(ml-base) -bash-4.2$ hpogrid submit RNN_TRT
INFO: Checking validity of configurations
Info: Validating model configuration
Info: Successfully validated model configuration
Info: Validating search space configuration...
Info: Successfully validated search space configuration
Info: Validating hpo configuration
Info: Successfully validated hpo configuration
Info: Validating grid configuration
Info: Successfully validated grid configuration
INFO: Submitting 1 grid job(s)
WARNING : The grid queue names could change due to consolidation, migration, etc
ne/valid queues when site and/or excludedSite options are specified.
INFO : gathering files under /afs/cern.ch/work/c/chlcheng/Repository/hpogrid/pro
INFO : upload source files
INFO : submit user.chlcheng.hpogrid.RNN_TRT.out.20200917152936/
INFO : succeeded. new jediTaskID=22636301
```

HPOGrid Features: Fetching Information of ATLAS Grid Sites

- Question: "What are the grid sites that I can use?"
- Answer: The HPOGrid package allows you to see what GPU sites are available by using the command

```
$ hpogrid sites
```

An example output will be

```
(ml-base) -bash-4.2$ hpogrid sites
```

	state	status	maxinputsize	maxmemory	maxtime
ANALY_BNL_GPU_ARC	ACTIVE	brokeroff	14336	48000	86400
ANALY_CSCS-HPC_GPU	ACTIVE	test	25000	6000	86400
ANALY_INFN-T1_GPU	ACTIVE	brokeroff	50000	9000	216000
ANALY_LRZ_GPU	ACTIVE	test	15000	6000	172800
ANALY_MANC_GPU_TEST	ACTIVE	online	30000	4000	259200
ANALY_MWT2_GPU	ACTIVE	online	20480	4100	432000
ANALY_ORNL_Summit_GPU	ACTIVE	test	400000	0	0
ANALY_OU_OSCER_GPU_TEST	ACTIVE	test	15000	2000	1800
ANALY_QMUL_GPU_TEST	ACTIVE	test	25000	3500	345600
ANALY_SLAC_GPU	ACTIVE	test	30000	9000	7200
DESY-HH_GPU	ACTIVE	brokeroff	50000	9000	216000

HPOGrid Features: Monitoring HPO Tasks

- After submitting jobs, users can query the job status in the terminal using:

```
$ hpogrid tasks show
```

An example output will be

```
(ml-base) -bash-4.2$ hpogrid tasks show --days 2 --status done finished running
INFO: Verifying Grid Proxy...
PBook user: Chi Lung Cheng
INFO: Verification is Successful.
INFO: Fetching PanDA Tasks...
INFO: Showing only max 1000 tasks in last 2 days. One can set "--days N" to see tasks in last N days,
      and "--limit M" to see at most M latest tasks

JediTaskID  Status     Fin% TaskName
__________________________________________________________________________________________
  22626964  finished     0% user.chlcheng.hpogrid.idds_test.out.20200916122721/
  22626963  finished     0% user.chlcheng.hpogrid.idds_test.out.20200916112834/
  22626061  finished     0% user.chlcheng.hpogrid.idds_test.out.20200916103517/
  22626056  running      0% user.chlcheng.hpogrid.idds_test.out.20200916103506/
  22624328  running      0% user.chlcheng.hpogrid.idds_test.out.20200916073347/
  22624327  finished     0% user.chlcheng.hpogrid.idds_test.out.20200916073336/
  22600878  done       100% user.chlcheng.hpogrid.FastCaloGAN_photon_200_205.out.20200914222805/
  22600872  done       100% user.chlcheng.hpogrid.FastCaloGAN_photon_200_205.out.20200914222727/
  22600868  done       100% user.chlcheng.hpogrid.FastCaloGAN_photon_200_205.out.20200914222659/
  22600866  done       100% user.chlcheng.hpogrid.FastCaloGAN_photon_200_205.out.20200914222640/
  22600864  done       100% user.chlcheng.hpogrid.FastCaloGAN_photon_200_205.out.20200914222630/
  22600861  done       100% user.chlcheng.hpogrid.FastCaloGAN_photon_200_205.out.20200914222621/
  22600855  done       100% user.chlcheng.hpogrid.FastCaloGAN_photon_200_205.out.20200914222552/
  22600759  done       100% user.chlcheng.hpogrid.FastCaloGAN_photon_200_205.out.20200914221109/
  22600756  done       100% user.chlcheng.hpogrid.FastCaloGAN_photon_200_205.out.20200914221100/
  22600748  running      0% user.chlcheng.hpogrid.FastCaloGAN_photon_200_205.out.20200914221022/
  22600740  done       100% user.chlcheng.hpogrid.FastCaloGAN_photon_200_205.out.20200914220945/
```

Visualization of HPO Result

- Question: ”Can I retrieve the results without going through the trouble of downloading the output from PanDA?”
- Answer: HPO results can be fetched from the PanDA server through the command line and converted into different output formats for visualization:

```
$ hpogrid report <project_name> [--options]
```

Available output formats are:

json, csv, html, interactive parallel coordinate plot, mlflow artifact

An example output will be

```
(ml-base) -bash-4.2$ hpogrid report RNN_TRT --to_csv --to_html
INFO: Verifying Grid Proxy...
PBook user: Chi Lung Cheng
INFO: Verification is Successful.
INFO: Showing only max 1000 tasks in last 30 days. One can set "--days N" to see tasks in last N days,
      and "--limit M" to see at most M latest tasks
INFO: Setting up HPO report with the following attributes
Project Name      : RNN_TRT
Hyperparameters   : ['batch_size', 'lr', 'lstm_hidden_size', 'dense_size', 'dense_activation']
Metric            : accuracy
Mode              : max
+----+-------------------+-------------+-------------+-------------+------------------+------------+
|    | dense_activation  | dense_size  | batch_size  |          lr | lstm_hidden_size |  accuracy  |
|----+-------------------+-------------+-------------+-------------+------------------+------------|
|  0 |      softmax      |        256  |       1024  | 0.00328537  |               32 |  0.85365   |
|  1 |      sigmoid      |         64  |        128  | 0.000377629 |               32 |  0.8526    |
|  2 |      softmax      |        256  |        512  | 0.00213682  |               64 |  0.8521    |
|  3 |       relu        |         32  |       2048  | 0.00542035  |               32 |  0.852     |
|  4 |      sigmoid      |        128  |        256  | 0.000858966 |               64 |  0.85175   |
|  5 |      softmax      |         32  |        256  | 0.000460081 |              128 |  0.8517    |
|  6 |       tanh        |        256  |        128  | 0.00115264  |               32 |  0.85145   |
|  7 |      softmax      |         64  |        512  | 0.000727671 |               64 |  0.8507    |
|  8 |       relu        |        128  |       2048  | 0.0229392   |               32 |  0.8506    |
|  9 |      sigmoid      |         64  |       2048  | 0.00206482  |               64 |  0.8506    |
| 10 |       relu        |         64  |       2048  | 0.00134387  |               32 |  0.8505    |
```

Example Payload: FastCaloGAN - I

- The FastCaloGAN payload uses the WGANGP (Wasserstein Generative Adversarial Network with Gradient Penalty) models for fast shower simulation in ATLAS. Details about the FastCaloGAN model can be found from the previous talks by Michele and Serena.
- The hyperparameters of interest are

Hyperparameter	Description
batchsize	Number of training samples in each iteration
lr	Learning rate of the optimizer
beta1	Exponential decay rate for the 1st moment estimates
Lambda	Scaling factor for gradient penalty
n_disc	Number of discriminators for the GAN model

- Since a typical trial for a FastCaloGAN model takes roughly 10 hours to complete on a single GPU, the random search method is used for hyperparameter tuning. (With the iDDS workflow, sequential methods such as Bayesian optimization can also be used)

Example Payload: FastCaloGAN - II

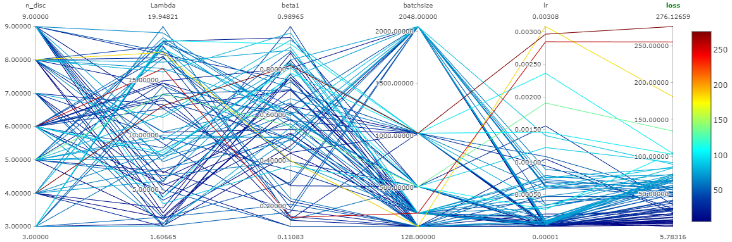
- Below is an example configuration file used to run the hyperparameter tuning for the FastCaloGAN payload (check the file here):


```yaml
project_name: FastCaloGAN_photon_200_205
scripts_path: /afs/cern.ch/u/username/examples/scripts/FastCaloGAN
model_config:
  script: conditional_wgangp.py
  model: WGANGP
  param:
    particle: photons
    pid: 22
    eta_min: 200
    eta_max: 205
hpo_config:
  algorithm: random
  metric: loss
  mode: min
  scheduler: asynchyperband
  num_trials: 1
  max_concurrent: 3
grid_config:
  site: ANALY_MANC_GPU_TEST,ANALY_MWT2_GPU,ANALY_BNL_GPU_ARC
  inDS: user.chlcheng:user.chlcheng.HPO.v2.FastCaloGAN.photons.eta_200_205.dataset
```

```yaml
search_space:
  n_disc:
    method: uniformint
    dimension:
      low: 3
      high: 10
  Lambda:
    method: uniform
    dimension:
      low: 1
      high: 20
  beta1:
    method: uniform
    dimension:
      low: 0.1
      high: 1
  batchsize:
    method: categorical
    dimension:
      categories:
      - 128
      - 256
      - 512
      - 1024
      - 2048
      grid_search: 0
  lr:
    method: loguniform
    dimension:
      low: 1.0e-05
      high: 0.01
```

Example Payload: FastCaloGAN - Results

- Hyperparameter optimization result from 82 trials (for photon with $0.2 < \eta < 0.205$):
  - Visualization using mlflow



  - Click here for an interactive parallel coordinate plot
  - Click here for an html summary table
  - Click here for a csv summary table
- Improvement with respect to the default hyperparameters:

Metric	Before Tuning	After Tuning	Improvement
χ^2	72.5	5.783	-66.7

Note: The default hyperparameters are used across all models (for different particle types and eta values). So it is probably performing poorly on this particular model.
For another model (pions with eta between 0.150 and 0.155), the χ^2 is around 15.

Example Payload: RNN for Particle Identification in TRT detector - I

- The RNN_TRT payload uses a recurrent neural network for particle identification in the TRT detector. Details about the model can be found from the previous talk by Arif.
- The hyperparameters of interest are

Hyperparameter	Description
batchsize	Number of training samples in each iteration
lr	Learning rate of the optimizer
dense_size	Size of dense layer
lstm_hidden_size	Size of the LSTM hidden layer
dense_activation	Activation function for the dense layers

- Since a typical trial for an RNN_TRT model takes roughly 5 hours to complete on a single GPU, the random search method is used for hyperparameter tuning. (With the iDDS workflow, sequential methods such as Bayesian optimization can also be used)

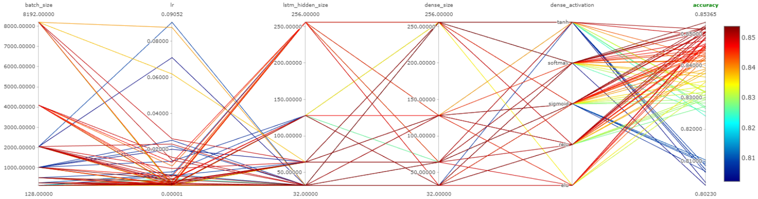
Example Payload: RNN for Particle Identification in TRT detector - II

- Below is an example configuration file used to run the hyperparameter tuning for the RNN TRT payload (check the file here):


```yaml
project_name: RNN_TRT
scripts_path: /afs/cern.ch/work/u/user/scripts/RNN_TRT
model_config:
  script: train.py
  model: RNN_TRT
  param:
    num_epochs: 100
    verbose: 0
hpo_config:
  algorithm: random
  metric: accuracy
  mode: max
  num_trials: 4
  max_concurrent: 4
grid_config:
  site: ANALY_MANC_GPU_TEST,ANALY_MWT2_GPU,ANALY_BNL_GPU_ARC
  inDS: user.chlcheng:user.chlcheng.RNNTRT.v04.dataset
search_space:
  batch_size:
    method: categorical
    dimension:
      categories:
      - 128
      - 256
      - 512
      - 1024
      - 2048
      - 4096
      - 8192
  lr:
    method: loguniform
    dimension:
      low: 1.0e-05
      high: 0.1
  lstm_hidden_size:
    method: categorical
    dimension:
      categories:
      - 32
      - 64
      - 128
      - 256
  dense_size:
    method: categorical
    dimension:
      categories:
      - 32
      - 64
      - 128
      - 256
  dense_activation:
    method: categorical
    dimension:
      categories:
      - relu
      - elu
      - softmax
      - sigmoid
      - tanh
```

Example Payload: RNN for Particle Identification in TRT detector - Results

- Hyperparameter optimization result from 74 trials:
  - Visualization using mlflow



  - Click here for an interactive parallel coordinate plot
  - Click here for an html summary table
  - Click here for a csv summary table
- Improvement with respect to the default hyperparameters:

Metric	Before Tuning	After Tuning	Improvement
Accuracy	0.843	0.854	1.3%

Example Payload: BDT for BBYY non-resonant analysis

- The BBYY payload uses the XGBoost BDT model to separate signal and background processes for the $HH \to bb\gamma\gamma$ non-resonant analysis.
- The hyperparameters from XGBoost are:
  - eta
  - alpha
  - gamma
  - lambda
  - subsample
  - max bin
  - max depth
  - max delta step
  - min child weight
  - scale pos weight
  - colsample bytree

  For more details, please refer to the XGBoost documentation
- Since a typical trial for a BBYY XGBoost model takes only a few minutes to complete on multiple CPUs, the DoubleFastGADiscreteOnePlusOne method from Nevergrad is used for hyperparameter tuning.

Example Payload: BDT for BBYY non-resonant analysis - I

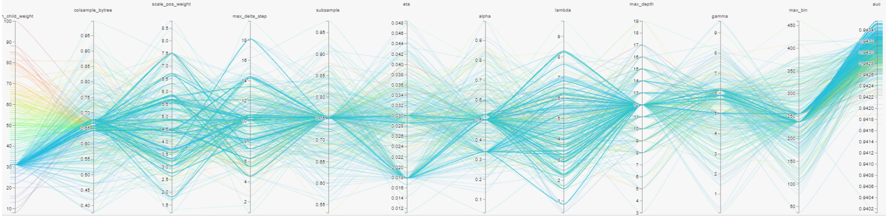
- Below is an example configuration file used to run the hyperparameter tuning for the BBYY (SM channel) payload (check the file here):

```
project_name: BBYY_SM
scripts_path: /afs/cern.ch/u/username/examples/scripts/BBYY/
model_config:
  script: train_bdt.py
  model: yybb_bdt
  param:
    channel: SM
    num_round: 10000
hpo_config:
  algorithm: nevergrad
  algorithm_param:
    method: DoubleFastGADiscreteOnePlusOne
  metric: auc
  mode: max
  num_trials: 1000
  max_concurrent: 3
grid_config:
  site: ANALY_MANC_GPU_TEST,ANALY_MWT2_GPU,ANALY_BNL_GPU_ARC
  inDS: user.chlcheng:user.chlcheng.hpo.bbyy.dataset.01
search_space:
  min_child_weight:
    method: uniformint
    dimension:
      low: 0
      high: 100
  colsample_bytree:
    method: uniform
    dimension:
      low: 0.3
      high: 1
  scale_pos_weight:
    method: uniform
    dimension:
      low: 0
      high: 9
  max_delta_step:
    method: uniform
    dimension:
      low: 0
      high: 20
```

```
subsample:
  method: uniform
  dimension:
    low: 0.5
    high: 1
eta:
  method: uniform
  dimension:
    low: 0.01
    high: 0.05
alpha:
  method: uniform
  dimension:
    low: 0
    high: 1
lambda:
  method: uniform
  dimension:
    low: 0
    high: 10
max_depth:
  method: uniformint
  dimension:
    low: 3
    high: 20
gamma:
  method: uniform
  dimension:
    low: 0
    high: 10
max_bin:
  method: uniformint
  dimension:
    low: 10
    high: 500
```

Example Payload: BDT for BBYY non-resonant analysis - Results

- Hyperparameter optimization result from 1000 trials:
  - Visualization using HiPlot



  - Click here for an interactive parallel coordinate plot
  - Click here for an html summary table
  - Click here for a csv summary table
- Improvement with respect to the default hyperparameters:

Metric	Before Tuning	After Tuning	Improvement
Significance	0.463	0.475	2.59%

Integration with iDDS

- For details about the iDDS workflow, please refer to presentation from Rui or the presentation from Wen earlier to this talk.
- Also check the links to iDDS tutorial and Twiki.
- With the HPOGrid package, one can submit grid jobs that run the idds workflow by doing

```
$ hpogrid submit <configuration_file> --mode idds
```

An example output will be

```
(ml-base) -bash-4.2$ hpogrid submit idds_test --mode idds
INFO: Checking validity of configurations
Info: Validating model configuration
Info: Successfully validated model configuration
Info: Validating search space configuration...
Info: Successfully validated search space configuration
Info: Validating hpo configuration
Info: Successfully validated hpo configuration
Info: Validating grid configuration
Info: Successfully validated grid configuration
INFO: Submitting 1 idds grid job(s)
INFO : submit user.chlcheng.hpogrid.idds_test.out.20200917154943/
INFO : succeeded. new jediTaskID=22636830
```

- This works by translating the configuration from the HPOGrid workflow to that of the iDDS workflow

Summary

- Running hyperparameter optimisation for your machine learning application using the ATLAS grid resource is easy and possibly fast.
- The HPOGrid workflow helps you implement the bulk of the codes and the steps required for running hyperparameter optimization.
- The HPOGrid workflow can run in iDDS.
- If you encounter any problems, I can always help you!