iDDS HyperParameter Optimization
development for machine learning

Wen Guan, Tadashi Maeno, Gancho Dimitrov
Brian Bockelman, Torre Wenaus
Fernando Barreiro, Fahui Lin, Rui Zhang, Misha Borodin, Paul Nilsson

June 15, 2020
ATLAS S&C
HyperParameterOptimization (HPO)

- **HyperParameterOptimization (HPO)**
  - A hyperparameter is a parameter to control the learning/training process in ML
  - HPO is to find a set of optimal parameters for a learning algorithms.
  - Ways to find hyperparameters:
    - Randomly/Gridly generate a lot of hyperparameters in one round to evaluate and then select the one producing the best results – not efficient in fine granularity
    - Searching hyperparameters based on previous hyperparameter results using some algorithms – multiple rounds
      - If not running locally, the users need to submit jobs, wait jobs to finish, read the results and then generate new hyperparameter to submit a new round of jobs: Users need to duplicate this operations in tens of hundreds of rounds.

- **iDDS development in HPO**
  - The whole idea is that users only need to submit one task for a HPO process, instead of working as a babysitter to monitor and react it again and again
  - iDDS HPO is developed to automate the HPO workflow with one submission, after a task is submitted:
    - iDDS waits and monitors the evaluation results of every set of hyperparameters
    - Generate new hyperparameters for a new round based on previous results
    - Trigger production system to evaluate the new hyperparameters.
When a HPO request is registered, iDDS will monitor the request

➢ Only one request per task
➢ Step 1, 2, 3 is a loop
➢ iDDS will terminate the task when some requirements reached.

A HPO task should include two parts
➢ Hyperparameter generating:
  ■ Option 1: define search space with predefined methods
  ■ Option 2: develop user container
➢ Evaluation
  ■ User ML training/learning process

HPO workflow

Submit a HPO task

Register a HPO request

1. Generating hyperparameters

2. Evaluate hyperparameters

3. Report results
iDDS HPO

❖ iDDS HPO functions
➢ HyperParameter generating:
  ■ iDDS predefined two Hyperparameter generating methods, to use them users only need to define the search space when registering the request:
    ● Bayesian
    ● Nevergrad
  ■ iDDS also supports user developed container
➢ Integrating with JEDI, pilot
  ■ iDDS provides Rest API and messaging (ActiveMQ) to pass new hyperparameters to JEDI
  ■ iDDS provides Rest API to report results to iDDS
➢ Logs integration:
  ■ iDDS also provides API to download logs for the hyperparameter generating step.
iDDS HPO status and future

❖ **status**
- Pre-production status being tested by ATLAS ML users
  - A example successful task: https://bigpanda.cern.ch/task/21592607/
- Possible to feed real workflow useful for ML with GPU resources like BNL IC GPU.

❖ **future**
- Improve the predefined hyperparameters generating methods and add more methods to cover most of the requirements.
- Currently only docker is supported, improve to support different types of containers.
iDDS Current Status

❖ Main architecture (production)
  ➢ iDDS database, core, REST API
  ➢ Plugins
  ➢ Agents
  ➢ Watchdogs

❖ Documents & monitors
  ➢ Home page: https://idds.cern.ch
  ➢ Codes: https://github.com/HSF/iDDS
  ➢ Documents: https://idds.readthedocs.io (dev)
    ■ HPO documents:
      ione.html
  ➢ ATLAS monitor: https://bigpanda.cern.ch/idds/

❖ Use cases
  ➢ Fine-grained data carousel -- ready
  ➢ Hyper Parameter Optimization (HPO) -- integrating
  ➢ Decision making for active learning -- developing
  ➢ Other use cases
backups
iDDS prod workflow: data carousel

❖ Data Carousel Integration with all datasets (mainly data15 and zerobias)

➢ 667 datasets
➢ 768, 115 files
➢ 1.2 PB
➢ iDDS was not fully occupied, will test with more tasks, finished zerobias datasets in few days.

<table>
<thead>
<tr>
<th>request_id</th>
<th>scope</th>
<th>name</th>
<th>status</th>
<th>transform_status</th>
<th>in_status</th>
<th>in_total_files</th>
<th>in_processed_files</th>
<th>out_status</th>
<th>out_total_files</th>
<th>out_processed_files</th>
</tr>
</thead>
<tbody>
<tr>
<td>724</td>
<td>21423307</td>
<td>21423307</td>
<td>Finished</td>
<td>Finished</td>
<td>Closed</td>
<td>1</td>
<td>1</td>
<td>Closed</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>723</td>
<td>21422500</td>
<td>21422500</td>
<td>Transforming</td>
<td>Transforming</td>
<td>Closed</td>
<td>1</td>
<td>1</td>
<td>Processing</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>722</td>
<td>21420035</td>
<td>21420035</td>
<td>Transforming</td>
<td>Transforming</td>
<td>Closed</td>
<td>1</td>
<td>1</td>
<td>Processing</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>721</td>
<td>21407107</td>
<td>21407107</td>
<td>Finished</td>
<td>Finished</td>
<td>Closed</td>
<td>1</td>
<td>1</td>
<td>Closed</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>720</td>
<td>21405829</td>
<td>21405829</td>
<td>Transforming</td>
<td>Transforming</td>
<td>Closed</td>
<td>1</td>
<td>1</td>
<td>Processing</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>719</td>
<td>21404562</td>
<td>21404562</td>
<td>Transforming</td>
<td>Transforming</td>
<td>Closed</td>
<td>1</td>
<td>1</td>
<td>Processing</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>718</td>
<td>21404113</td>
<td>21404113</td>
<td>Transforming</td>
<td>Transforming</td>
<td>Closed</td>
<td>1</td>
<td>1</td>
<td>Processing</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>717</td>
<td>21391145</td>
<td>21391145</td>
<td>Finished</td>
<td>Finished</td>
<td>Closed</td>
<td>1</td>
<td>1</td>
<td>Closed</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>716</td>
<td>21391132</td>
<td>21391132</td>
<td>Transforming</td>
<td>Transforming</td>
<td>Closed</td>
<td>1</td>
<td>1</td>
<td>Processing</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>715</td>
<td>21390625</td>
<td>21390625</td>
<td>Transforming</td>
<td>Transforming</td>
<td>Closed</td>
<td>1</td>
<td>1</td>
<td>Processing</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>

Showing 1 to 10 of 695 entries

A lot of tasks have finished: The monitor has increased to 70 pages

❖ New functions
➢ Automatically create new rules to transfer files from other sites when some files are waiting too long time
   ■ Create a subdataset for long waiting files
   ■ Create a new replication rule to transfer from other sites.
iDDS HPO (Hyper Parameter Optimization)

- **Workflow**
  - User submits HPO tasks to JEDI with sampling step (steering container) and evaluation step.
  - JEDI submits the sampling step to iDDS.
  - iDDS generates multiple steps of hyperparameters based on the results of previous steps.
  - For every step, iDDS triggers JEDI to generate new PanDA jobs to evaluate new hyperparameters.

Diagram:
- User submits a HPO task to the Production system.
- The Production system submits a hyper parameter request to iDDS.
- iDDS generates new parameters through new content messages (ActiveMQ).
- iDDS registers the loss through the iDDS API.
- ML jobs generate ML jobs with ES.
- Production system generates new parameters through new content messages (ActiveMQ).
- Production system finishes the loop when HPO generators return an empty list or None.
iDDS Active Learning (AL)

Active learning
➢ Running tasks on top of results of old tasks
➢ Decision making to generate new tasks from old results
   ■ Light job, good to execute it immediately and then trigger next step. iDDS can get rid of some latency.

Workflow with grid entities
➢ Production system processes the normal task
➢ iDDS runs the Decision Making parts(with/without merge parts)

- Each job runs evgen and derivation sequentially
- Each task generates many multi-step jobs
- Once the first task is finished output are merged and some values are calculated
- Parameters for the next task are decided using the values
- Better to get rid of latency in the intermediate steps (merge+physics and decision making)