



ROBERTTLANGE



@TUBERLIN



@ROBERTTLANGE

MLE-Toolbox: Training Neural Networks Like an Experimental Biologist Would

Robert Tjarko Lange



@SPREKELERLAB



@ECNBERLIN



@SCIOI



@FORAI_ML

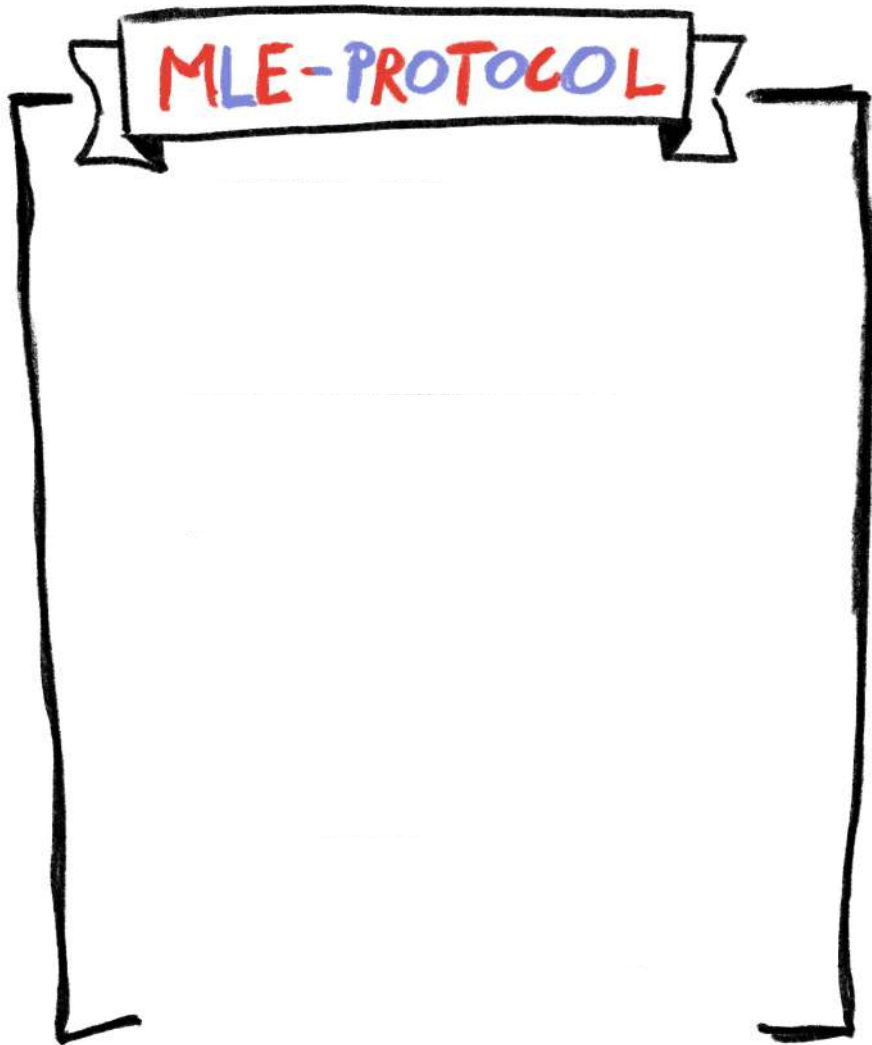
Science & Engineering - Hypothesis Formulation & Testing

400 Networks
x 5 random seeds
x 4 Lifetimes
+ Post-Processing



- ① Experimental protocol - Efficient testing & reproducibility
- ② Pipeline - Coordination of parallel scheduling across resources

MLE-Protocol: Never run a simulation twice!



- ① Purpose/Hypothesis
What question are you trying to answer?
- ② Materials
What code version? Git Commit Hash
- ③ Methods
Which hyperparams? Hash of configs
- ④ Evaluation
What figures shall be created?



GRID ENGINE

run-experiment



slurm
workload manager

monitor-cluster



retrieve-experiment



Google Cloud Platform

report-experiment

ML-Toolbox

Types of Supported Jobs and Ingredients

Parallel
Experiments

Seeds

Folds

Configs

Hyperparameter
Search

Random

Grid

SMBO

Training Script

<train>.py

DeepLogger

get_configs_ready

Training Config

<job>.json

net_config

train_config

log_config

Experiment Config

<meta>.yaml

meta_job

single_job

spec_job

<train>.py

MNIST-CNN (I): Training Code/Pipeline

DeepLogger

get_configs_ready

```
from mle_toolbox.utils import get_configs_ready, DeepLogger

def main(net_config, train_config, log_config):
    """ Train a network on MNIST dataset. """
    ...
    # Setup the experiment run - device/dataloader/net, etc.
    train_loader, test_loader = ..., ...
    mnist_net = MNIST_CNN(**net_config).to(device)
    nll_loss = nn.CrossEntropyLoss()
    optimizer = torch.optim.Adam(mnist_net.parameters(), lr=train_config.l_rate)
    run_log = DeepLogger(**log_config)

    # Train the MNIST Network using the training loop
    train_mnist_cnn(...)

def train_mnist_cnn(...):
    """ Run the training loop over a set of epochs. """
    for epoch in range(1, num_epochs + 1):
        # Run the training loop ...
        train_loss, test_loss = ..., ...

        # Log newest results
        time_tick = [epoch]
        stats_tick = [train_loss, test_loss]
        train_log.update_log(time_tick, stats_tick)
        train_log.save_log()
        train_log.save_network(model)
    return model, train_log

if __name__ == "__main__":
    conf = get_configs_ready(default_config_fname="mnist_cnn_config_1.json")
    train_config, net_config, log_config = conf
    main(net_config, train_config, log_config)
```

<job>.json

MNIST-CNN (II): Base Configuration

net_config

log_config

train_config

```
{
  "net_config": {"dropout_prob": 0.5,
                 "hidden_fc_dim": 128},
  "log_config": {"time_to_track": ["epoch"],
                 "what_to_track": ["train_loss", "test_loss"],
                 "tboard_fname": "config_1",
                 "time_to_print": ["epoch"],
                 "what_to_print": ["train_loss", "test_loss"],
                 "print_every_k_updates": 1,
                 "overwrite_experiment_dir": 1},
  "train_config": {"seed_id": 0,
                   "num_epochs": 2,
                   "net_type": "CNN",
                   "torch_num_threads": 3,
                   "train_batch_size": 256,
                   "test_batch_size": 256,
                   "l_rate": 0.001}
}
```


<meta>.yaml

MNIST-CNN (III): Experiment Configuration

meta_job

spec_job

single_job

```
# Meta Arguments: What job? What train .py file? Base config? Where to store?
meta_job_args:
  project_name: "examples"
  job_type: "hyperparameter-search"
  base_train_fname: "mnist/run_mnist_cnn.py"
  base_train_config: "mnist/mnist_cnn_config_1.json"
  experiment_dir: "experiments/"

# Parameters specific to the hyperparameter search
param_search_args:
  search_type: "grid"
  hyperlog_fname: "ppo_hyper_log.pkl"
  maximize_objective: True
  problem_type: "final"
  eval_score_type: "test_loss"
  num_search_batches: 1
  num_iter_per_batch: 2
  num_evals_per_iter: 3
  params_to_search:
    categorical:
      opt_type:
        - "Adam"
        - "RMSprop"
    real:
      l_rate:
        begin: 1e-5
        end: 1e-2
        bins: 2

# Parameters specific to an individual job
single_job_args:
  job_name: "cnn"
  num_gpus: 1
  num_logical_cores: 4
```


mkp-cluster (ssh)

(mle-toolbox) RobTLange@cognition12:~/mle-toolbox/mle_toolbox/examples\$

mkp-cluster (ssh)

cognition13 Wed Oct 7 20:12:09 2020

```
[0] GeForce RTX 2080 Ti | 29'C, 0 % | 0 / 11019 MB |
[1] GeForce RTX 2080 Ti | 29'C, 0 % | 0 / 11019 MB |
[2] GeForce RTX 2080 Ti | 27'C, 0 % | 0 / 11019 MB |
[3] GeForce RTX 2080 Ti | 25'C, 0 % | 0 / 11019 MB |
```

mkp-cluster (ssh)

1 Meta Arguments: What job? What train .py file? Base config? Where to store?

2 meta_job_args:

3 project_name: "examples"

4 job_type: "hyperparameter-search"

5 base_train_fname: "mnist/run_mnist_cnn.py"

6 base_train_config: "mnist/mnist_cnn_config_1.json"

7 experiment_dir: "experiments/"

8

9 # Parameters specific to the hyperparameter search

10 param_search_args:

11 search_type: "grid"

12 hyperlog_fname: "ppo_hyper_log.pkl"

13 maximize_objective: True

14 problem_type: "final"

15 eval_score_type: "test_loss"

16 num_search_batches: 2

17 num_iter_per_batch: 2

18 num_evals_per_iter: 1

19 params_to_search:

20 categorical:

21 opt_type:

22 - "Adam"

23 - "RMSprop"

24 real:

25 l_rate:

26 begin: 1e-5

27 end: 1e-2

28 bins: 2

29

30 # Parameters specific to an individual job

31 single_job_args:

32 job_name: "cnn"

33 num_gpus: 1

34 num_logical_cores: 4

35 exclude_nodes:

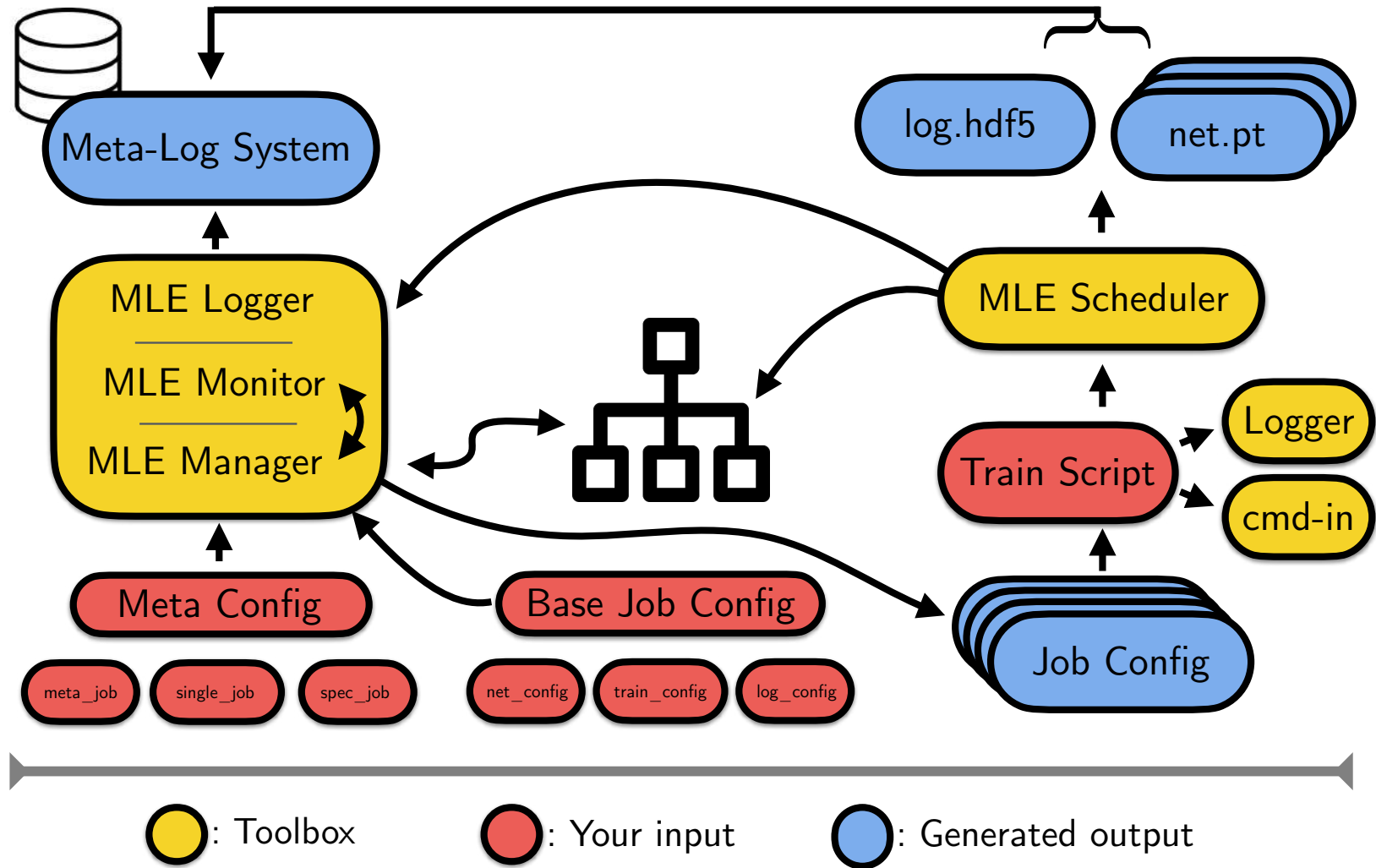
36 - 14

NORMAL mnist_search.yaml

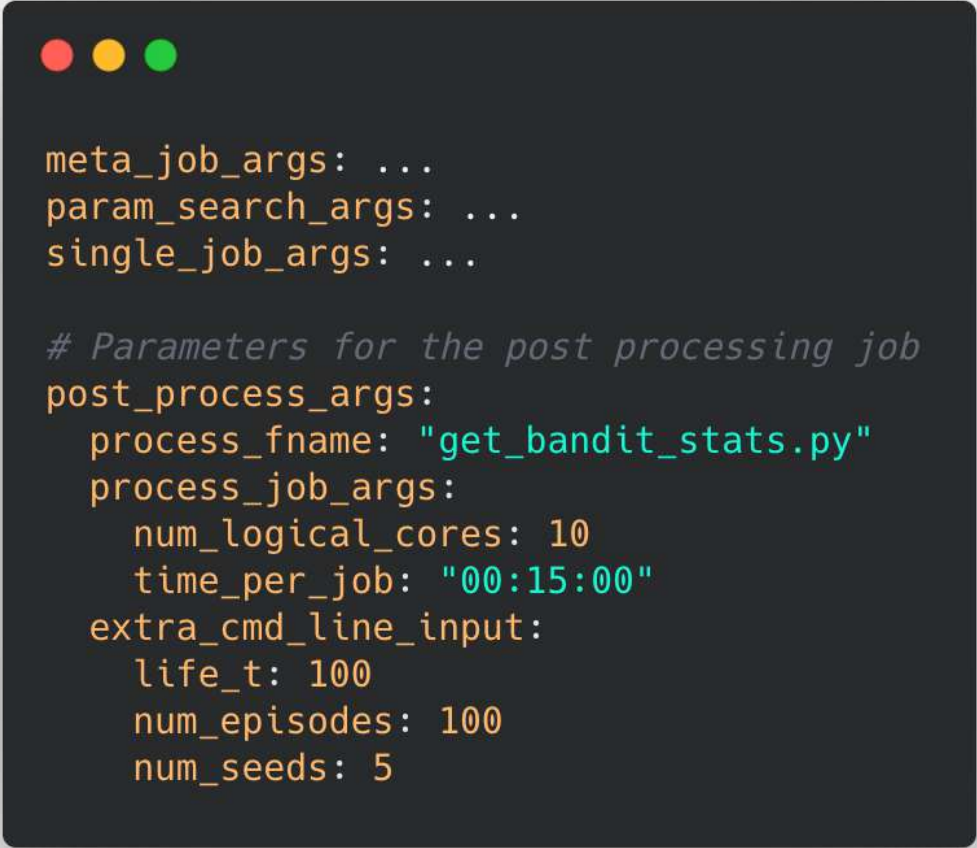
unix | utf-8 | yaml 2% 1:1

:q

The MLE-Toolbox Experiment Workflow



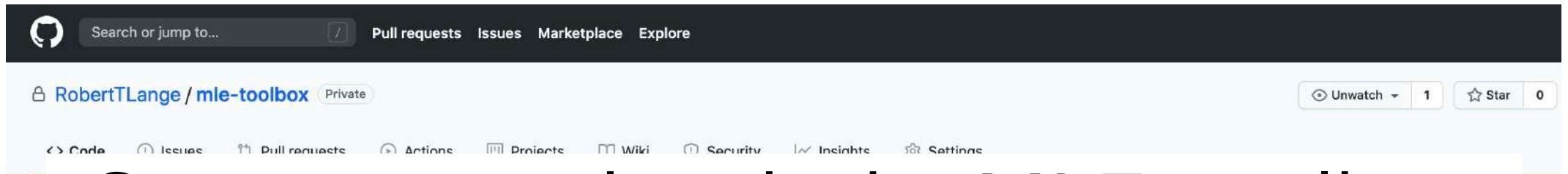
Post-Processing Pipelines: Automatic Figure Generation



```
meta_job_args: ...
param_search_args: ...
single_job_args: ...

# Parameters for the post processing job
post_process_args:
  process_fname: "get_bandit_stats.py"
  process_job_args:
    num_logical_cores: 10
    time_per_job: "00:15:00"
  extra_cmd_line_input:
    life_t: 100
    num_episodes: 100
    num_seeds: 5
```

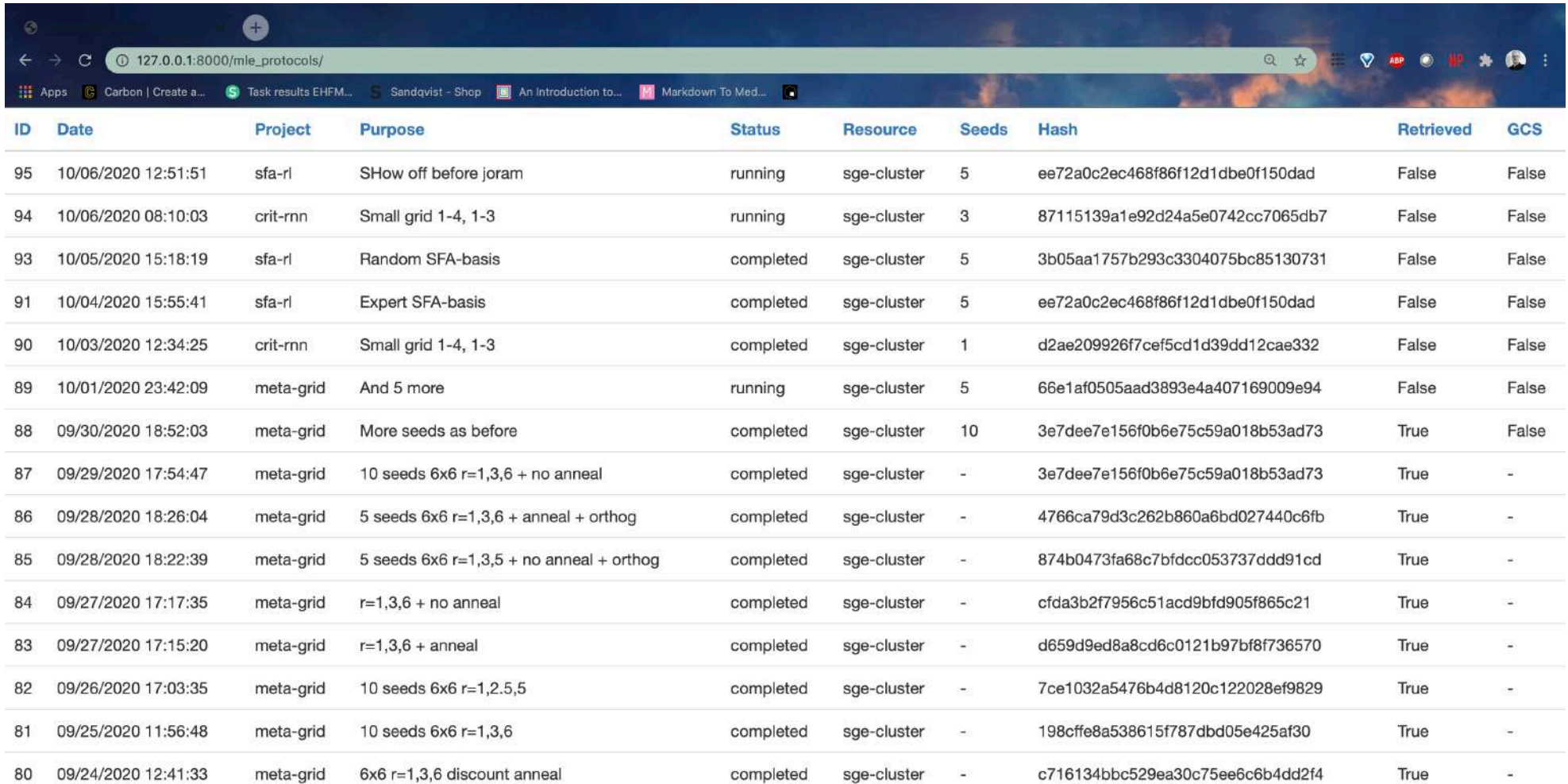
PyPI & Manual Installation from GitHub Repository



Getting started with the MLE-Toolbox:

1. `pip install mle-toolbox`
2. Edit `cluster_config.py`
3. Go through examples (`.yaml/.json`)
4. Run your own experiment

To Be Continued: The MLE-Laboratory



The image shows a screenshot of a web browser displaying a table of project results. The browser's address bar shows the URL '127.0.0.1:8000/mle_protocols/'. The table has 10 columns: ID, Date, Project, Purpose, Status, Resource, Seeds, Hash, Retrieved, and GCS. The data is sorted by ID in descending order, from 95 at the top to 80 at the bottom. The 'Status' column indicates whether a project is 'running' or 'completed'. The 'Retrieved' and 'GCS' columns show boolean values for data retrieval and storage status.

ID	Date	Project	Purpose	Status	Resource	Seeds	Hash	Retrieved	GCS
95	10/06/2020 12:51:51	sfa-rl	SHow off before joram	running	sge-cluster	5	ee72a0c2ec468f86f12d1dbe0f150dad	False	False
94	10/06/2020 08:10:03	crit-rnn	Small grid 1-4, 1-3	running	sge-cluster	3	87115139a1e92d24a5e0742cc7065db7	False	False
93	10/05/2020 15:18:19	sfa-rl	Random SFA-basis	completed	sge-cluster	5	3b05aa1757b293c3304075bc85130731	False	False
91	10/04/2020 15:55:41	sfa-rl	Expert SFA-basis	completed	sge-cluster	5	ee72a0c2ec468f86f12d1dbe0f150dad	False	False
90	10/03/2020 12:34:25	crit-rnn	Small grid 1-4, 1-3	completed	sge-cluster	1	d2ae209926f7cef5cd1d39dd12cae332	False	False
89	10/01/2020 23:42:09	meta-grid	And 5 more	running	sge-cluster	5	66e1af0505aad3893e4a407169009e94	False	False
88	09/30/2020 18:52:03	meta-grid	More seeds as before	completed	sge-cluster	10	3e7dee7e156f0b6e75c59a018b53ad73	True	False
87	09/29/2020 17:54:47	meta-grid	10 seeds 6x6 r=1,3,6 + no anneal	completed	sge-cluster	-	3e7dee7e156f0b6e75c59a018b53ad73	True	-
86	09/28/2020 18:26:04	meta-grid	5 seeds 6x6 r=1,3,6 + anneal + orthog	completed	sge-cluster	-	4766ca79d3c262b860a6bd027440c6fb	True	-
85	09/28/2020 18:22:39	meta-grid	5 seeds 6x6 r=1,3,5 + no anneal + orthog	completed	sge-cluster	-	874b0473fa68c7bfdcc053737ddd91cd	True	-
84	09/27/2020 17:17:35	meta-grid	r=1,3,6 + no anneal	completed	sge-cluster	-	cfda3b2f7956c51acd9bfd905f865c21	True	-
83	09/27/2020 17:15:20	meta-grid	r=1,3,6 + anneal	completed	sge-cluster	-	d659d9ed8a8cd6c0121b97bf8f736570	True	-
82	09/26/2020 17:03:35	meta-grid	10 seeds 6x6 r=1,2.5,5	completed	sge-cluster	-	7ce1032a5476b4d8120c122028ef9829	True	-
81	09/25/2020 11:56:48	meta-grid	10 seeds 6x6 r=1,3,6	completed	sge-cluster	-	198cffe8a538615f787dbd05e425af30	True	-
80	09/24/2020 12:41:33	meta-grid	6x6 r=1,3,6 discount anneal	completed	sge-cluster	-	c716134bbc529ea30c75ee6c6b4dd2f4	True	-

Backup Slides

[illegible]

Population-Based Training

