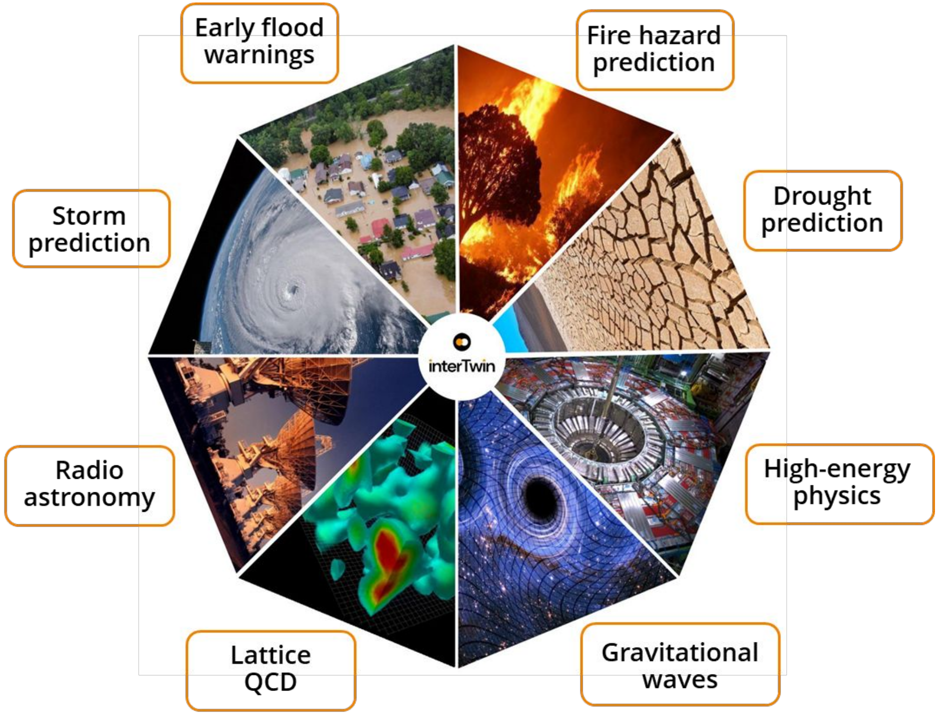
Digital Twins in Physics and Climate Science - interTwin

Anna Elisa Lappe, Jarl Sondre Saether, Matteo Bunino, Maria Girone

Funded by the European Union

The interTwin project is funded by the European Union - Grant Agreement Number 101058386

interTwin Use Cases



Virgo — Noise Simulation for Gravitational Waves Detector

Background: Gravitational Wave (GW) interferometers detect GWs produced by the acceleration of massive objects, such as black holes or neutron stars

Detector measures deformation of interferometer arms
→ **the strain**

Constant monitoring of interferometer status and environmental conditions to control noise
→ **auxiliary channels**

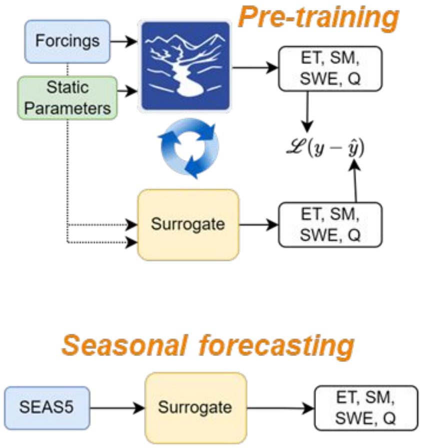
Goal: Denoise main detector channel using AI-generated signal glitches from auxiliary channels

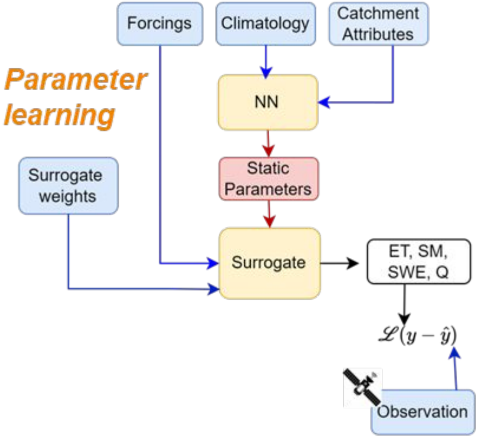
CERN openlab

EURAC — Drought Early Warning in the Alps 🏔️

Motivation — Climate change is making precipitation and temperature more extreme, increasing pressure on Alpine water resources

Objective — Develop probabilistic forecasts on hydrologically relevant variables to improve early drought warnings

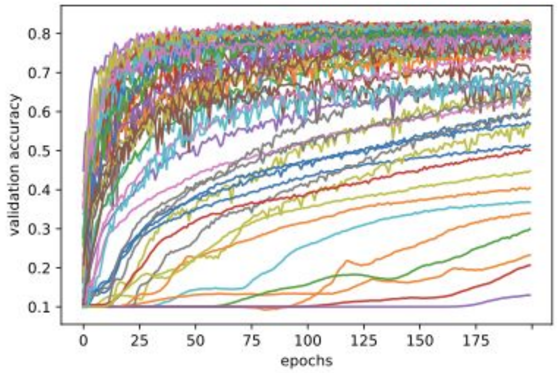




Hyperparameter Optimisation (HPO)

Motivation - why HPO?

- ❖ Machine learning models rely on hyperparameters (e.g., learning rate, batch size) that significantly impact performance
- ❖ Finding the best hyperparameters manually is inefficient and time-consuming
- ❖ Automated HPO improves model accuracy, efficiency, and reproducibility

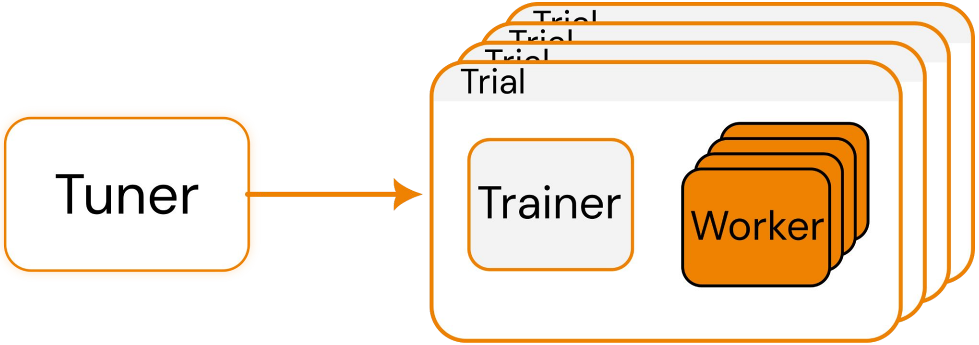


Example learning curves of different convolutional neural networks (CNNs) on the CIFAR-10 dataset

(Wulff, E., Garcia Amboage, J.P., Aach, M. *et al.* Distributed hybrid quantum-classical performance prediction for hyperparameter optimization. *Quantum Mach. Intell.* 6, 59 (2024). https://doi.org/10.1007/s42484-024-00198-5)

Distributed HPO

- Some trials might be very computationally expensive
  - Non-distributed could be infeasible
- **Itwinai** fully supports distributed HPO
- Allowing each trial to be distributed across multiple workers



HPO Study on EURAC Model — Setup

Parameters tuned

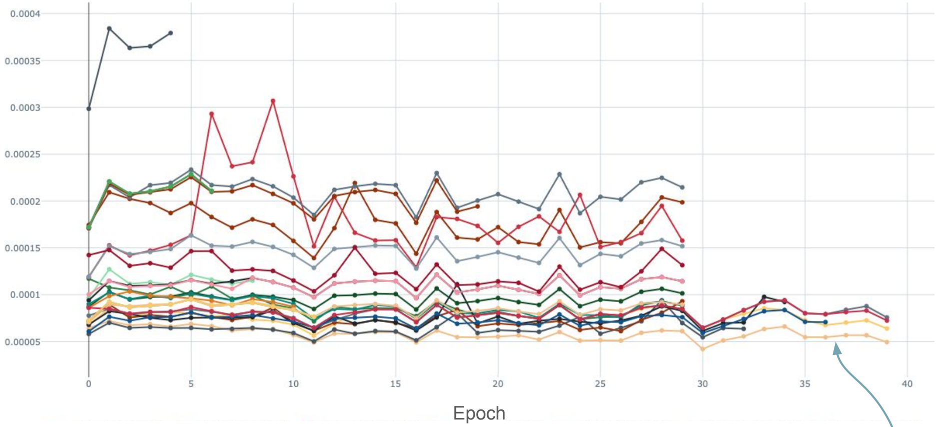
Trial name	status	batch	learning_rate	hidden_size	dropout	seq_length	interval_value
run_trial_ab248_00000	PENDING	1024	0.00724235	349	0.3	240	10
run_trial_ab248_00001	PENDING	768	0.00209553	152	0.2	330	5
run_trial_ab248_00002	PENDING	384	0.000335666	393	0.3	240	3
run_trial_ab248_00003	PENDING	640	0.00926136	307	0.5	210	3
run_trial_ab248_00004	PENDING	640	0.0044102	360	0.5	270	10

31 more PENDING

- Tuning Setup:
  - 28 trials for a maximum of 40 epochs
  - Using Random Search and ASHA (Asynchronous Hyperband) Scheduler
- This was run on 7 nodes with 4 GPUs each (28 GPUs), for 18 hours
- On a dataset with 80 000 samples

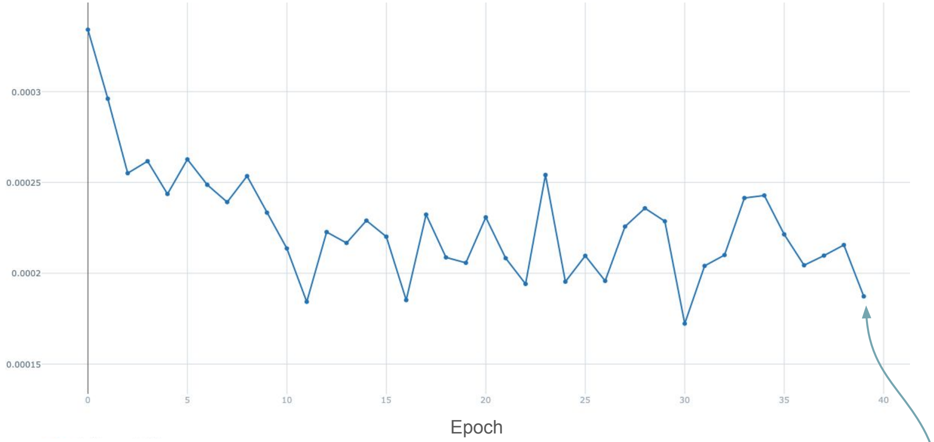
HPO Study on EURAC Model — Results

Validation losses with HPO:




```
Results of best configuration:
loss=4.795851782546379e-05 with params={'batch':
896, 'learning_rate': 0.00036054555544544169,
'hidden_size': 43, 'dropout': 0.4, 'seq_length':
240, 'interval_value': 3}
```

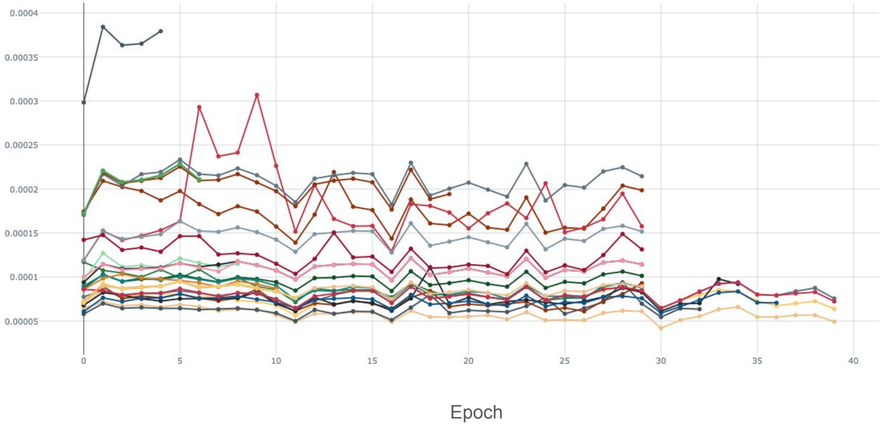
With developer-suggested hyperparameters:



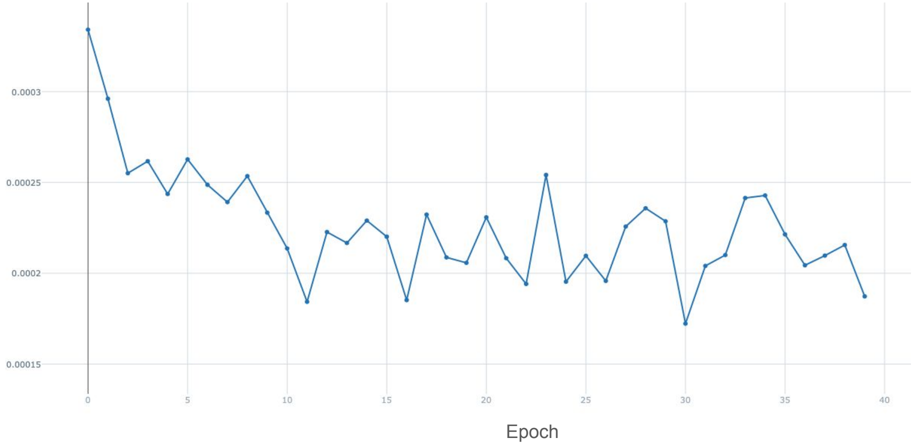

```
Results of configuration: loss=1.872707944130525e-04
with params={'batch': 256, 'learning_rate': 0.001,
'hidden_size': 136, 'dropout': 0.4, 'seq_length':
120, 'interval_value': 2}
```

HPO Study on EURAC Model — Results

Validation losses with HPO:



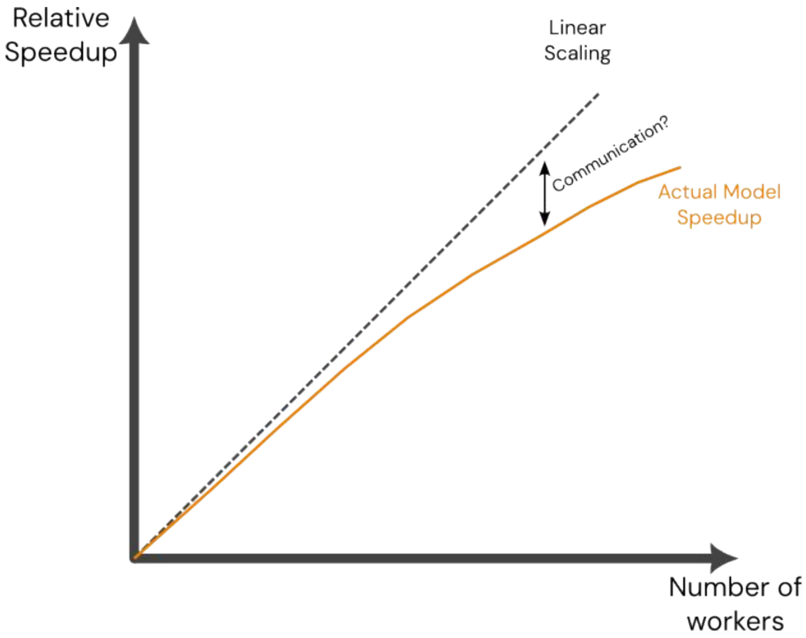
With developer-suggested hyperparameters:



~74 % decrease in validation loss

Scalability Analysis for Distributed Machine Learning

- 2x compute → ½ time?
- Communication overhead
- Different strategies—which to choose?



Different Distributed Strategies

itwinai currently supports three strategies:

- PyTorch’s Distributed Data Parallel (DDP)
- Microsoft’s DeepSpeed (DS)
- Horovod

Out of these, DDP is considered the *de facto* standard

Image Sources:
- https://github.com/pytorch/pytorch
- https://github.com/horovod/horovod
- https://www.microsoft.com/en-us/research/project/deepspeed/

The itwinai Scalability Report

Goals:

- Measure the *model's* scalability wrt. number of workers
- Find the best distributed strategy for you

Five metrics:

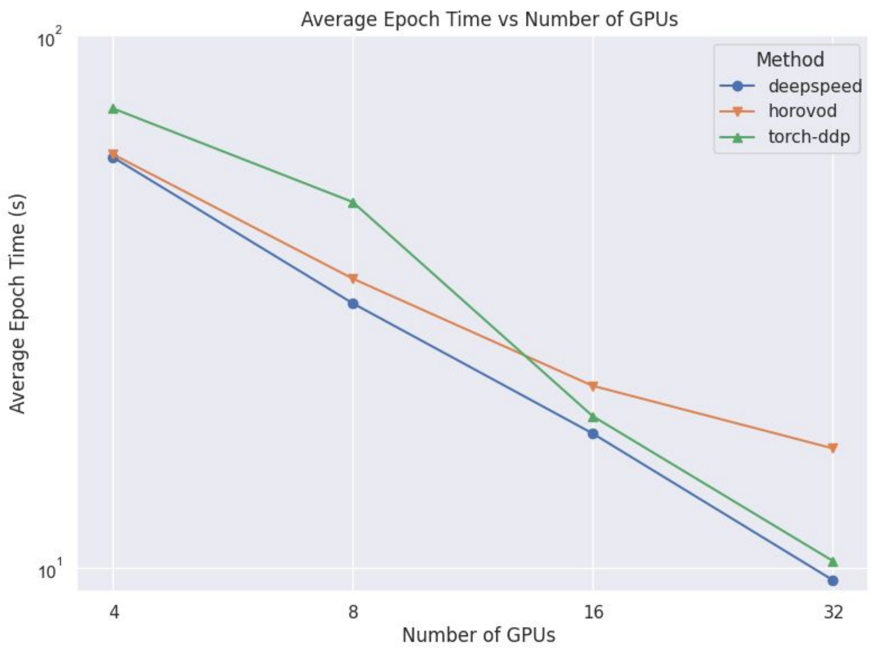
- Average time per epoch
- Relative speedup of time per epoch
- GPU Utilization (0–100%)
- GPU Power Consumption (watt-hours)
- Communication overhead (0–100%)

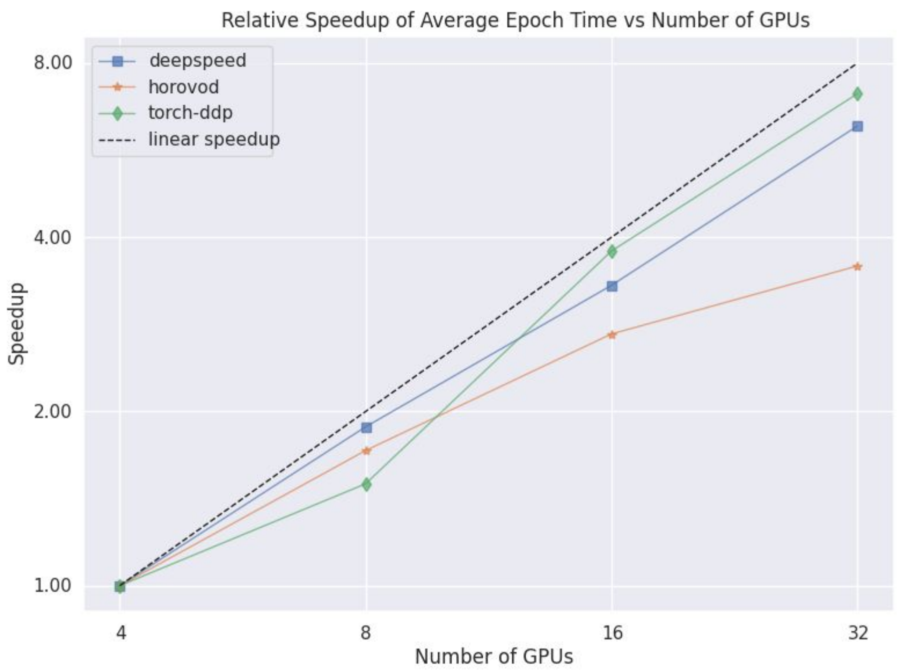
The itwinai Scalability Report

Combination of tools:

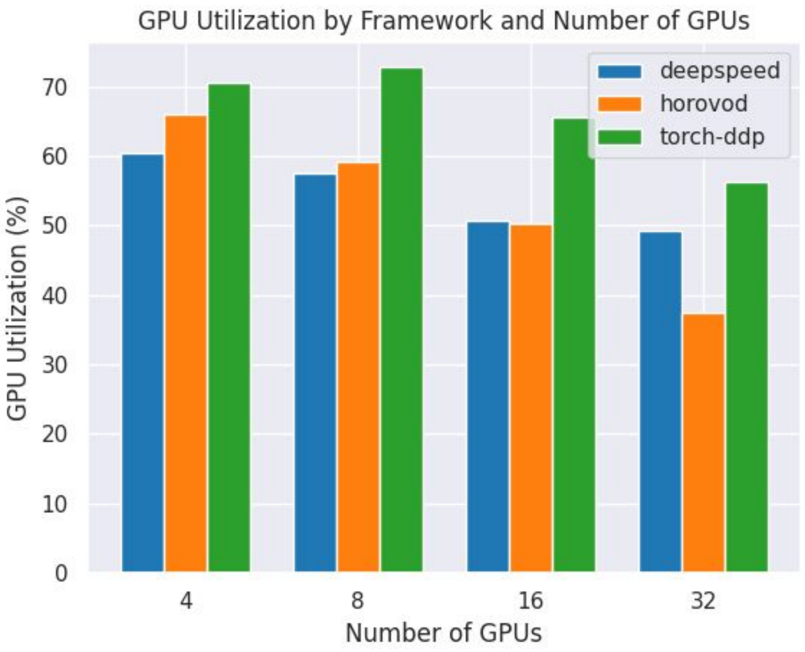
- ML Profiler for measuring communication overhead
  - Looks at e.g. CUDA communication calls
  - Compares it to PyTorch's ATen library
- GPU monitoring
  - Probing GPUs at regular intervals
- And more

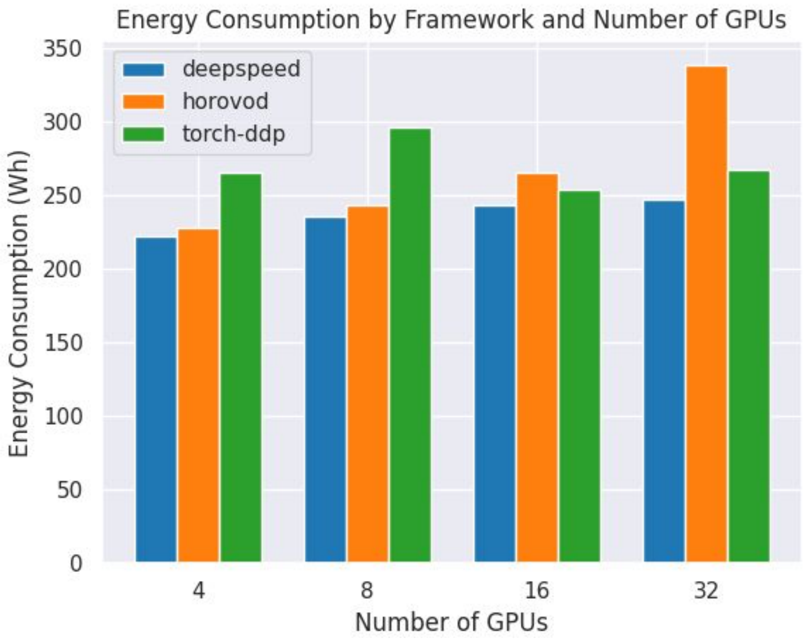
Example Report (Virgo)



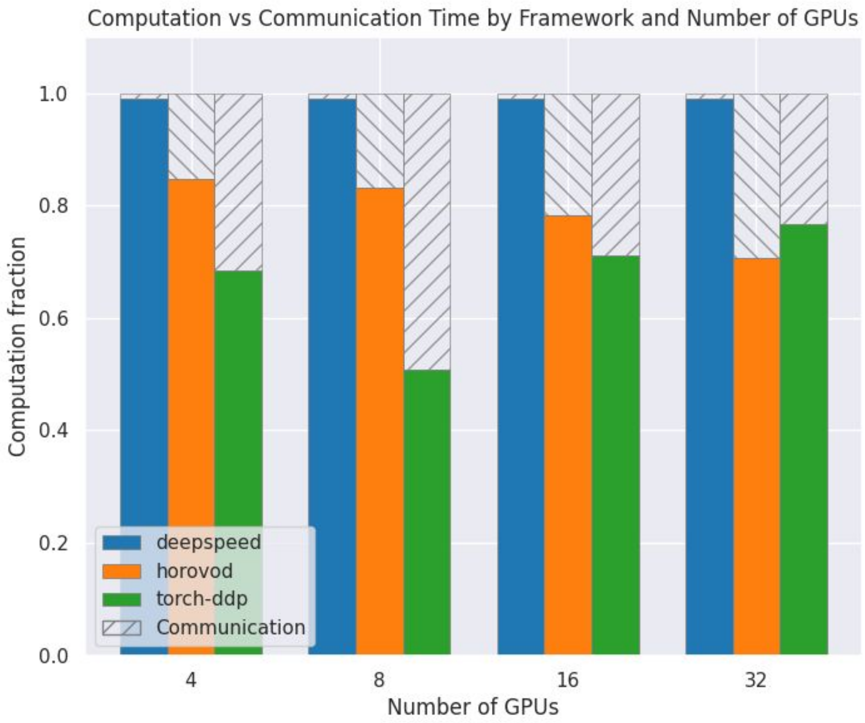


Example Report





Example Report



[WIP] CodeCarbon Integration—Dashboard for sustainability

Carbon Footprint

Measure Compute Emissions

Across All Projects

Net Power Consumption : 2.4 kWh

Net Carbon Equivalent : 0.5 kg

Select a Project

codecarbon

Infrastructure Hosted at nan, Germany

Power Consumption Across All Experiments : 2.4 kWh

Carbon Equivalent Across All Experiments : **0.5 kg**

Last Run Power Consumption : 0.3 kWh

Last Run Carbon Equivalent : 0.1 kg

Exemplary Equivalents

0.30 % of weekly American household emissions

1 miles driven

5 hours of 32-inch LCD TV watched

Thank you! Questions?

www.intertwin.eu

info@intertwin.eu

intertwin_eu

intertwin