Using HPC Clusters For Predictive Maintenance in Manufacturing

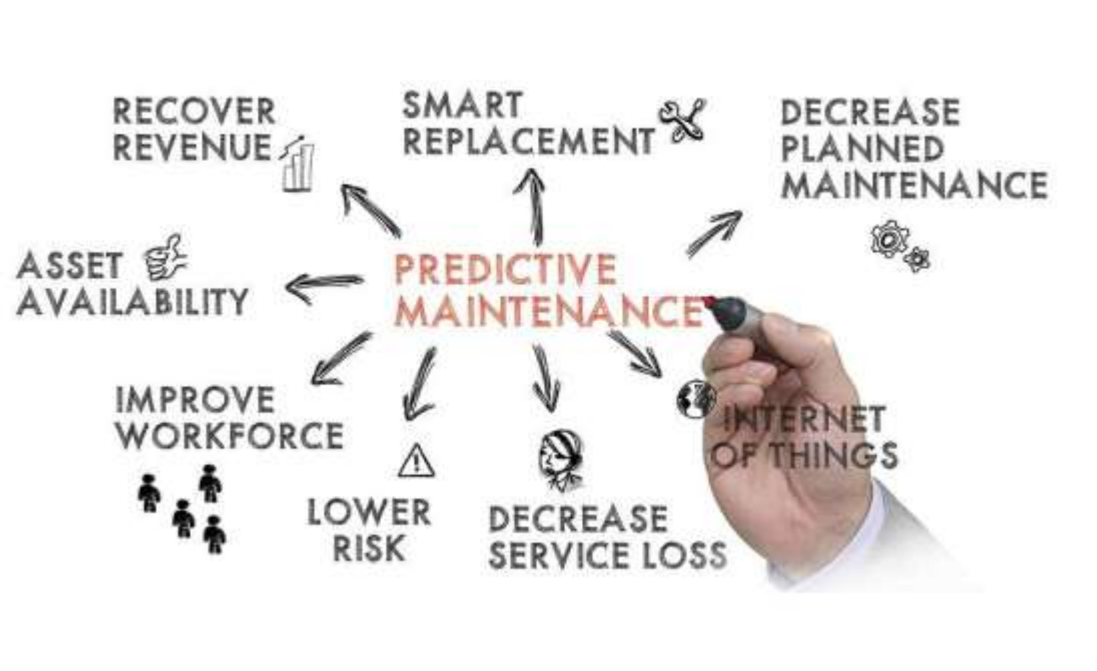
Berkay Demireller
ERSTE Software

Predictive Maintenance

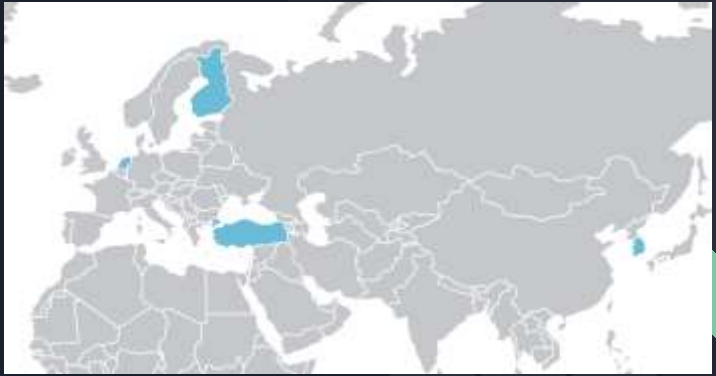
Predictive Maintenance

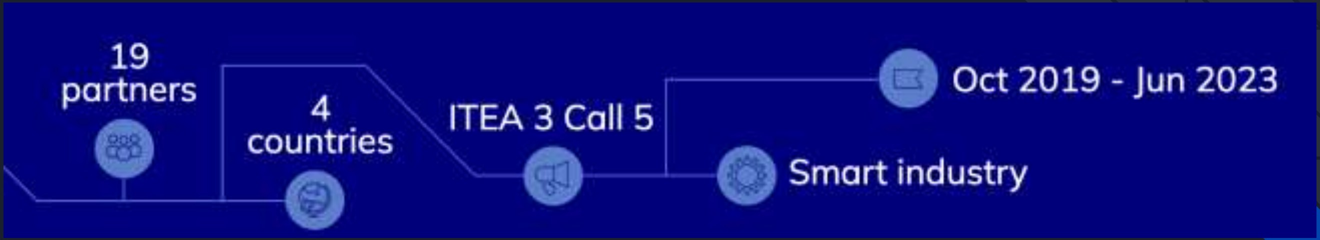
- A collection of techniques designed to monitor and act on equipment health

- Aim is to minimize unplanned stoppage of manufacturing operations due to failure



MACHINAIDE





19 partners

4 countries

ITEA 3 Call 5

Oct 2019 - Jun 2023

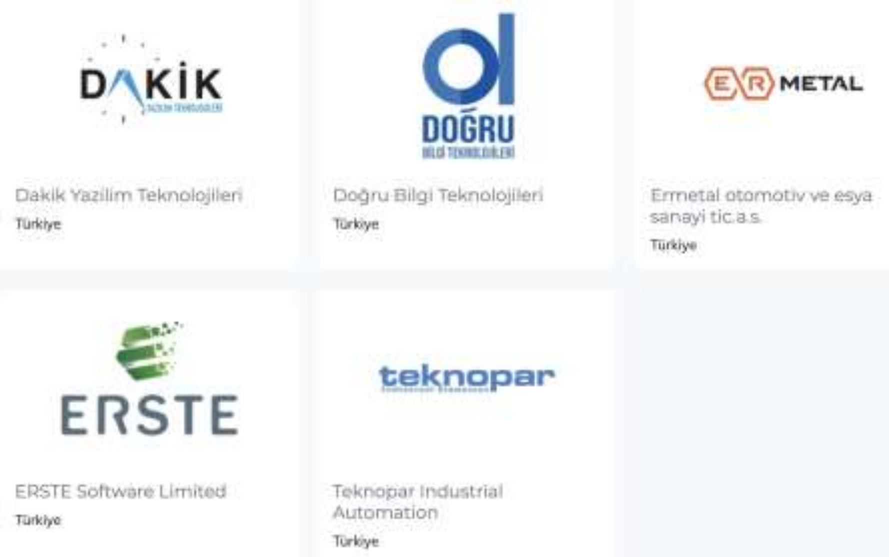
Smart industry

machinaide

- Machinaide aims to provide innovative ways of engaging with Digital Twin data

- One of the tasks the Turkey use case focuses is Predictive Maintenance through ML and Deep Learning

- Unfortunately, models and/or training data can be very large



Dakik Yazilim Teknolojileri
Türkiye

Doğru Bilgi Teknolojileri
Türkiye

Ermetal otomotiv ve esya sanayi tic.a.s.
Türkiye

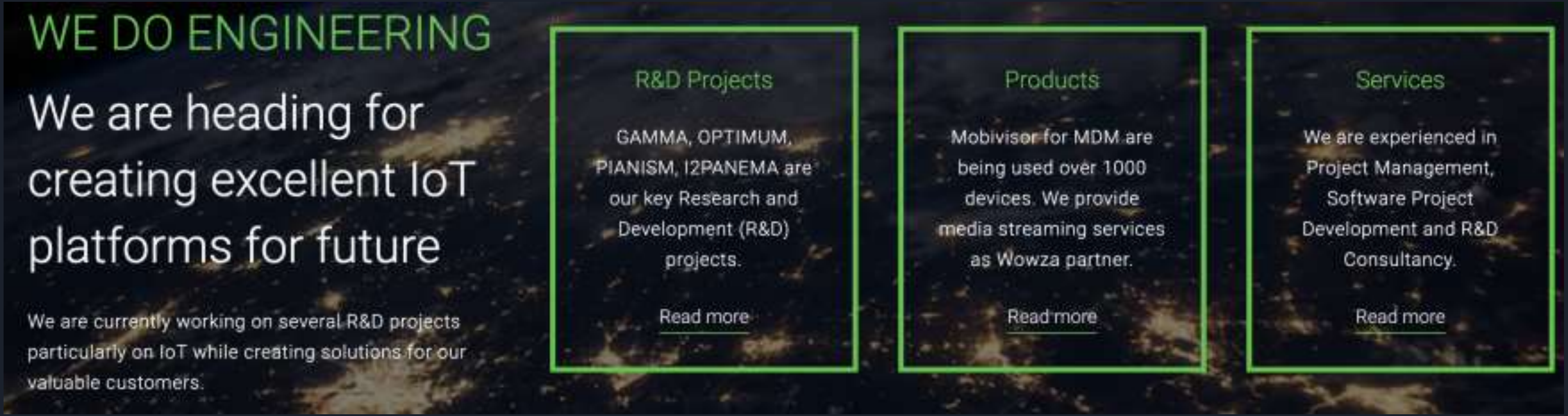
ERSTE Software Limited
Türkiye

Teknopar Industrial Automation
Türkiye

- Models work on real time data -> need to re-train models as new information becomes available.
- Large data/models mean resource heavy training process.
- Servers affordable by the end user usually are not very powerful, resources are mostly occupied by the service/application itself.
- That is training cannot be performed on the application server which is being used for visualization, dashboarding, rule-based data processing and inference.

Enabling HPC for ML-based Manufacturing

Enabling HPC - Erste



WE DO ENGINEERING

We are heading for creating excellent IoT platforms for future

We are currently working on several R&D projects particularly on IoT while creating solutions for our valuable customers.

R&D Projects

GAMMA, OPTIMUM, PIANISM, I2PANEMA are our key Research and Development (R&D) projects.

Read more

Products

Mobivisor for MDM are being used over 1000 devices. We provide media streaming services as Wowza partner.

Read more

Services

We are experienced in Project Management, Software Project Development and R&D Consultancy.

Read more

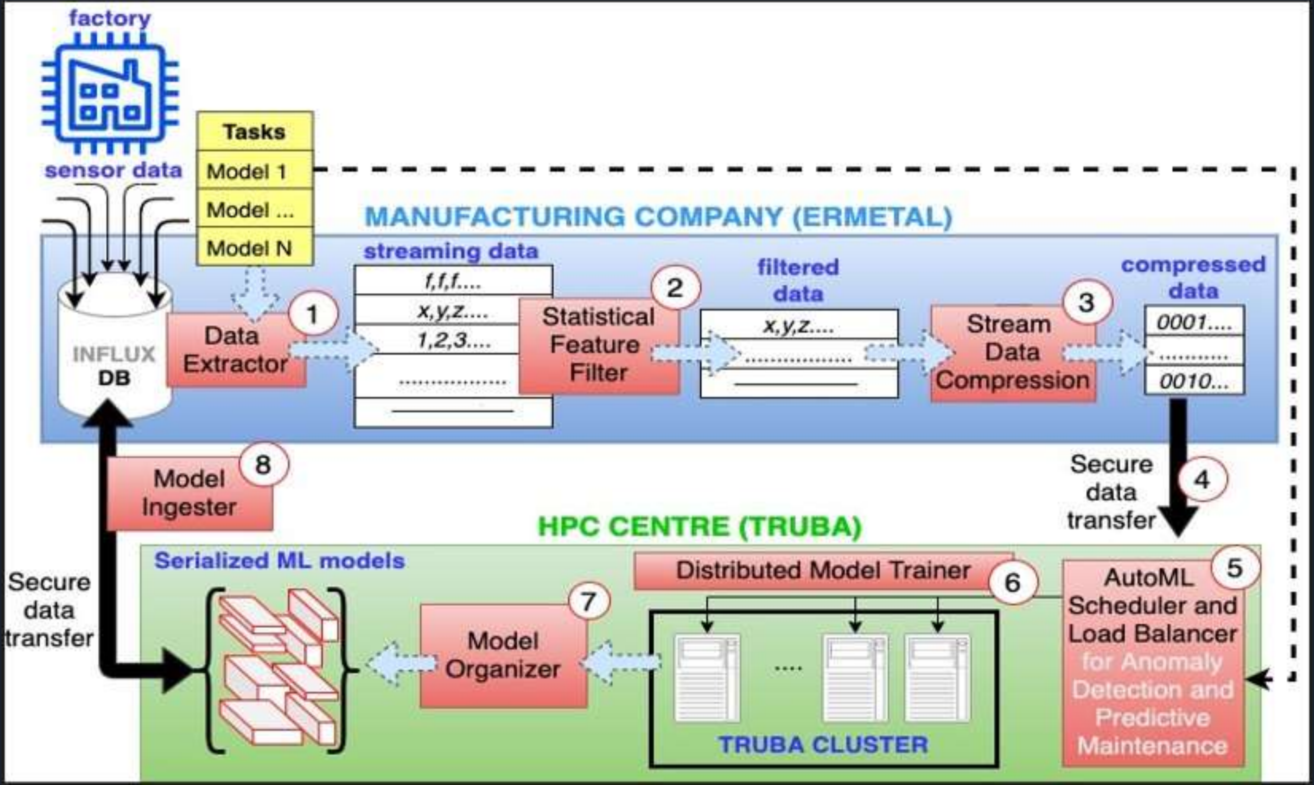
Enabling HPC - Erste

- Although we are used to work with streaming data and Machine Learning, before this EuroCC use-case we did not have much HPC experience.

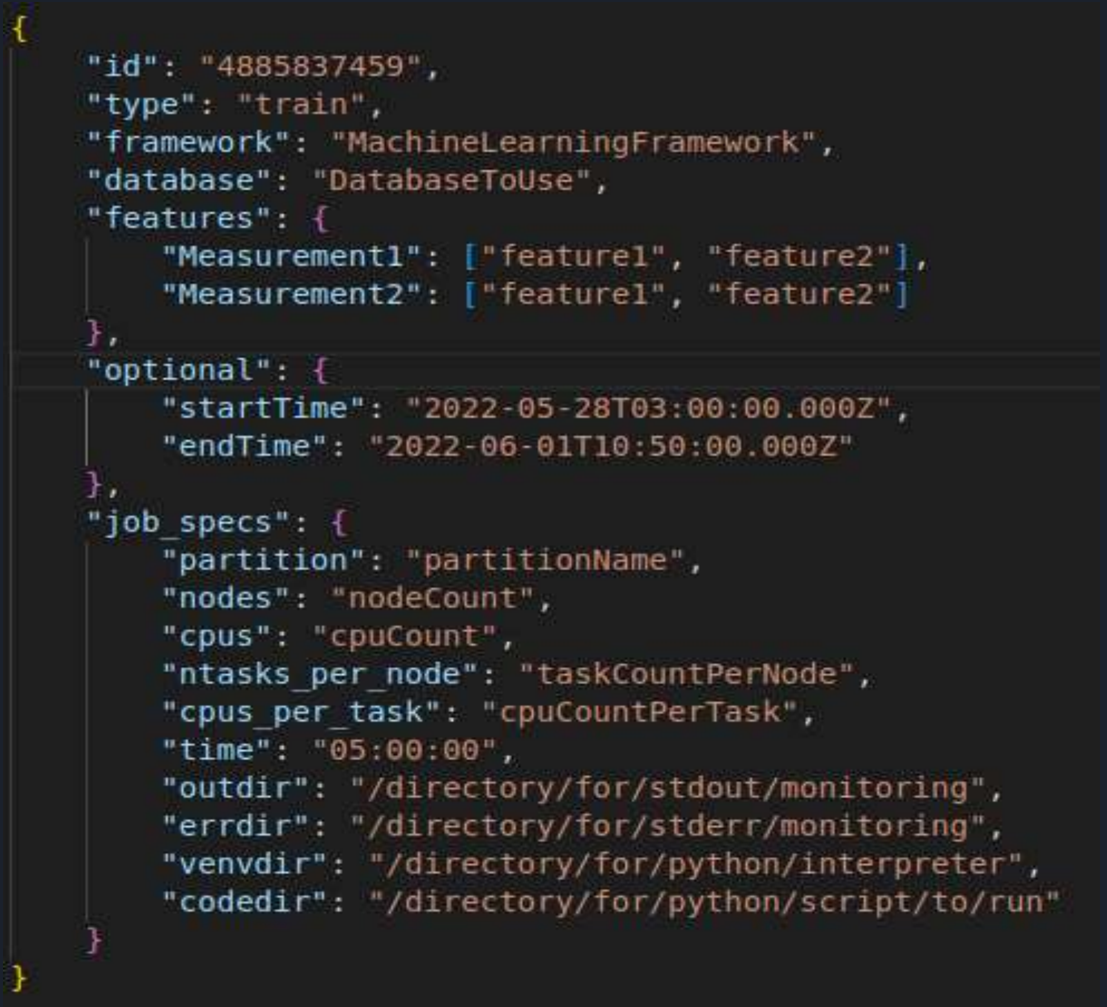
- Having access to an HPC cluster enables us to perform resource heavy training operations.

- We believe, this experience and know-how obtained will be useful in our other projects.

Enabling HPC – A generic picture

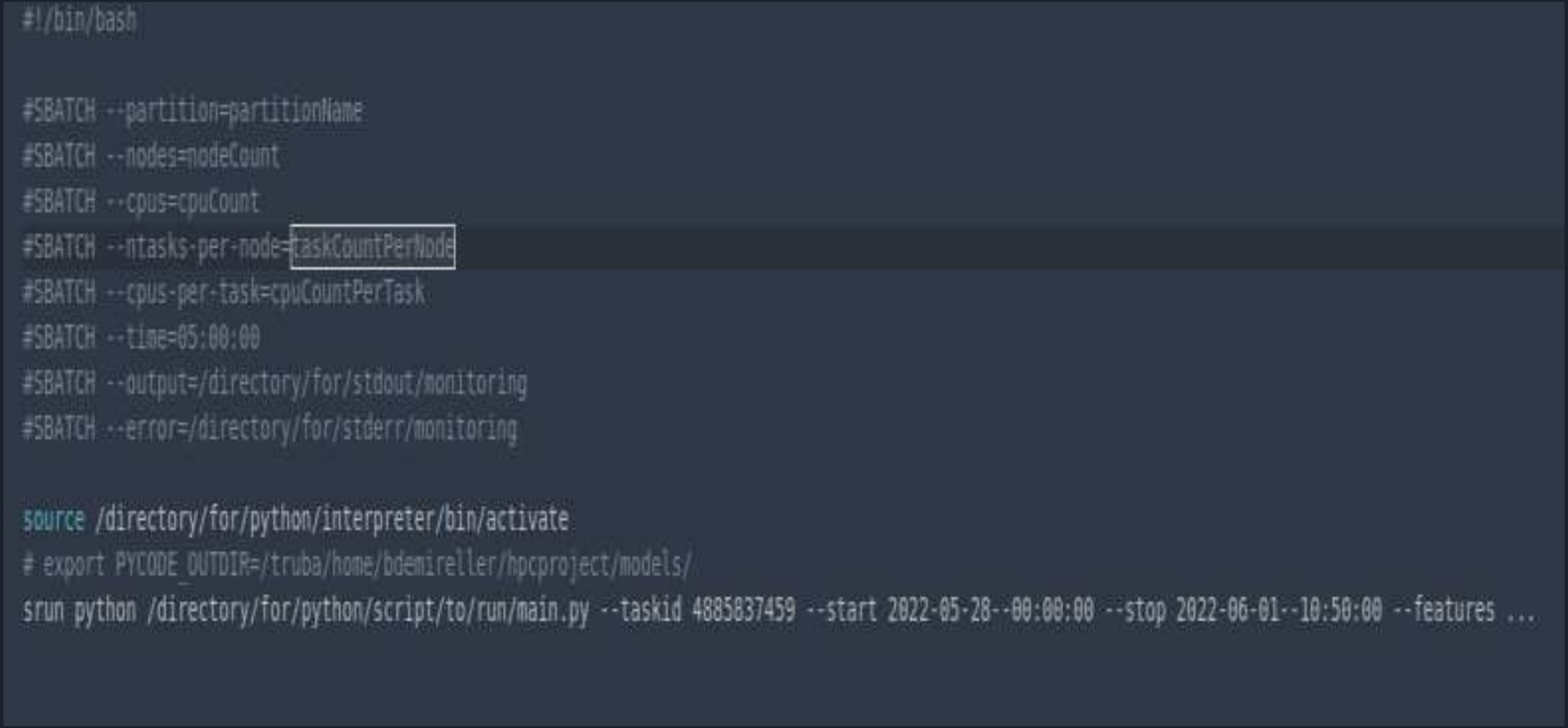


HPC Access - Task Creation



```
{
    "id": "4885837459",
    "type": "train",
    "framework": "MachineLearningFramework",
    "database": "DatabaseToUse",
    "features": {
        "Measurement1": ["feature1", "feature2"],
        "Measurement2": ["feature1", "feature2"]
    },
    "optional": {
        "startTime": "2022-05-28T03:00:00.000Z",
        "endTime": "2022-06-01T10:50:00.000Z"
    },
    "job_specs": {
        "partition": "partitionName",
        "nodes": "nodeCount",
        "cpus": "cpuCount",
        "ntasks_per_node": "taskCountPerNode",
        "cpus_per_task": "cpuCountPerTask",
        "time": "05:00:00",
        "outdir": "/directory/for/stdout/monitoring",
        "errdir": "/directory/for/stderr/monitoring",
        "venvdir": "/directory/for/python/interpreter",
        "codedir": "/directory/for/python/script/to/run"
    }
}
```

HPC Access - Task Creation



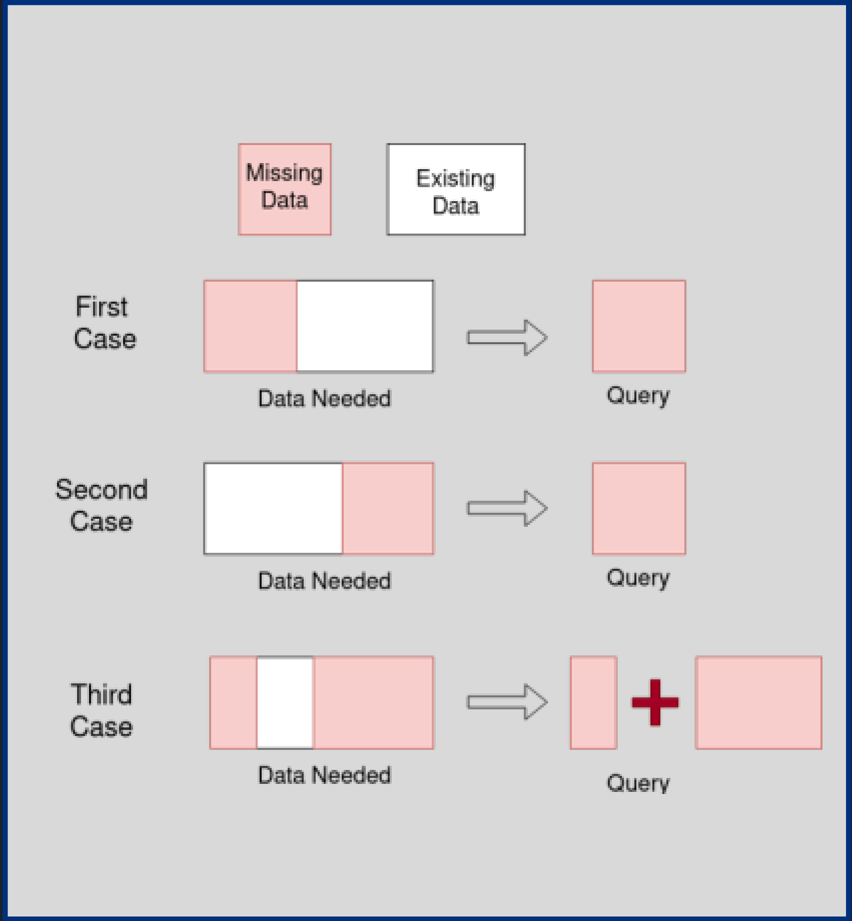
```
#!/bin/bash

#SBATCH --partition=partitionName
#SBATCH --nodes=nodeCount
#SBATCH --cpus=cpuCount
#SBATCH --ntasks-per-node=taskCountPerNode
#SBATCH --cpus-per-task=cpuCountPerTask
#SBATCH --time=05:00:00
#SBATCH --output=/directory/for/stdout/monitoring
#SBATCH --error=/directory/for/stderr/monitoring

source /directory/for/python/interpreter/bin/activate
# export PYCODE_OUTDIR=/truba/home/bdemireller/hpcproject/models/
srun python /directory/for/python/script/to/run/main.py --taskid 4885837459 --start 2022-05-28--00:00:00 --stop 2022-06-01--10:50:00 --features ...
```

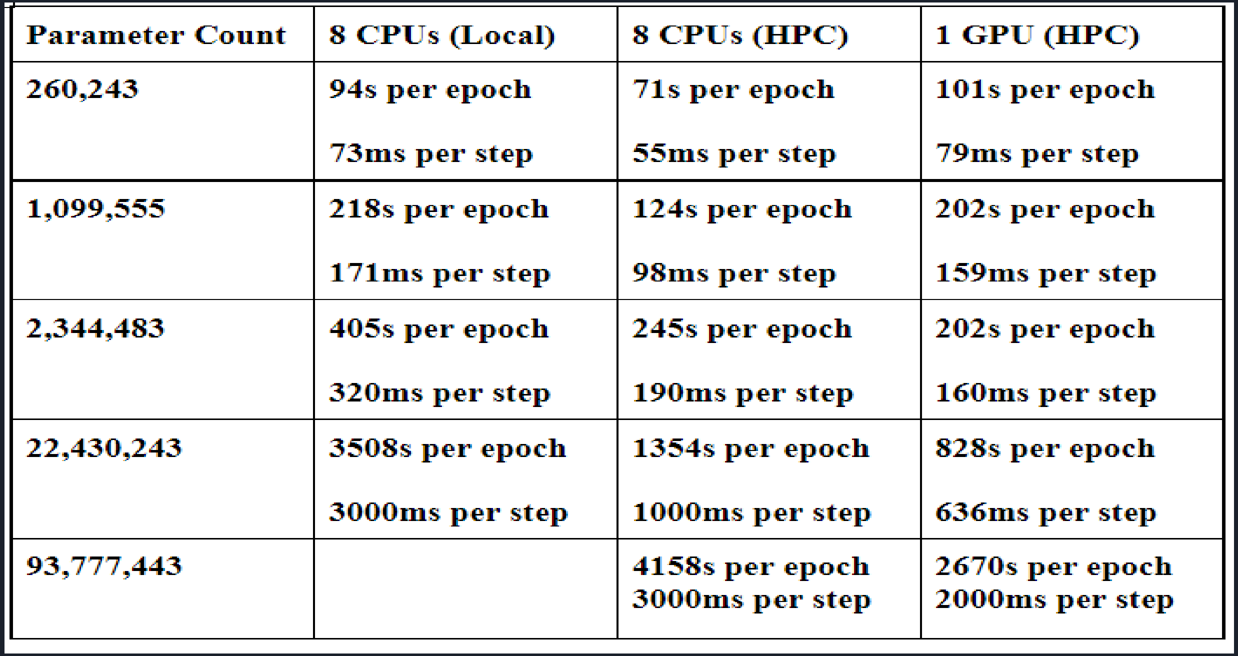
HPC Access - Data

- Data acquisition starts on the host machine
- There are special cases to consider



HPC Access - Results

- Setting: 4.5MB Data -> ~4 seconds of data/model/script transfer offset between the host machine and the cluster.



Parameter Count	8 CPUs (Local)	8 CPUs (HPC)	1 GPU (HPC)
260,243	94s per epoch 73ms per step	71s per epoch 55ms per step	101s per epoch 79ms per step
1,099,555	218s per epoch 171ms per step	124s per epoch 98ms per step	202s per epoch 159ms per step
2,344,483	405s per epoch 320ms per step	245s per epoch 190ms per step	202s per epoch 160ms per step
22,430,243	3508s per epoch 3000ms per step	1354s per epoch 1000ms per step	828s per epoch 636ms per step
93,777,443		4158s per epoch 3000ms per step	2670s per epoch 2000ms per step

Questions