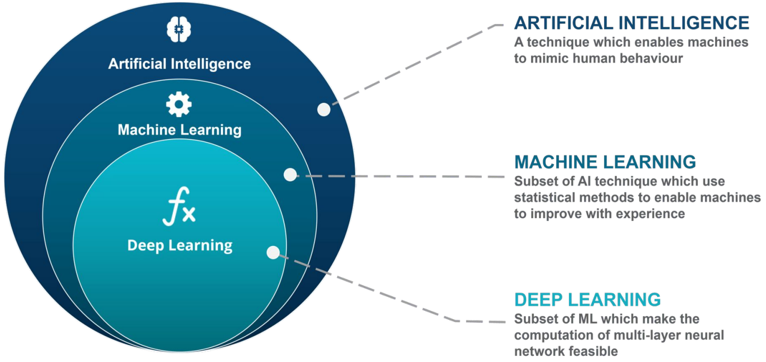
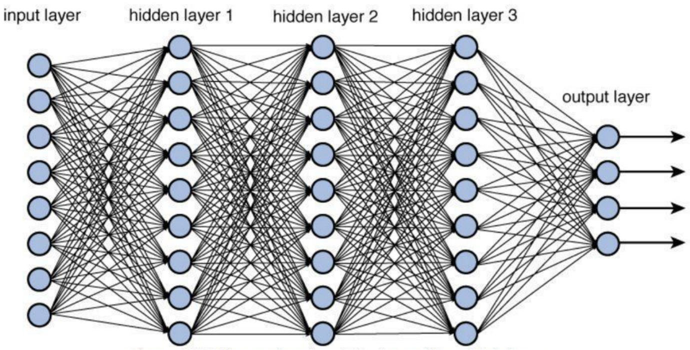
Deep Learning at NERSC

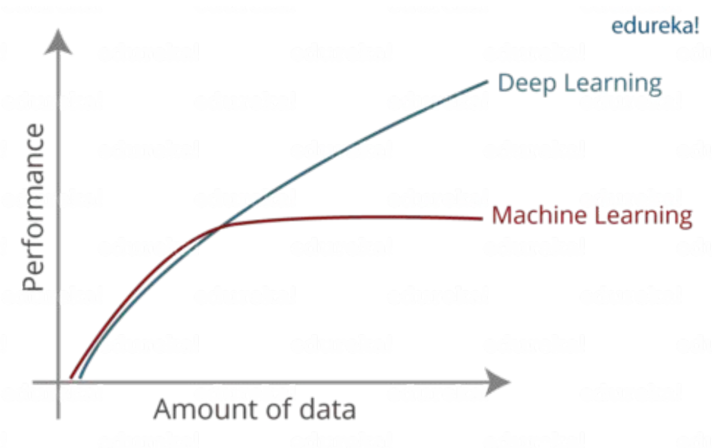
Grads@NERSC: How to Do Deep Learning with Jupyter Notebooks and Beyond
April 11, 2024

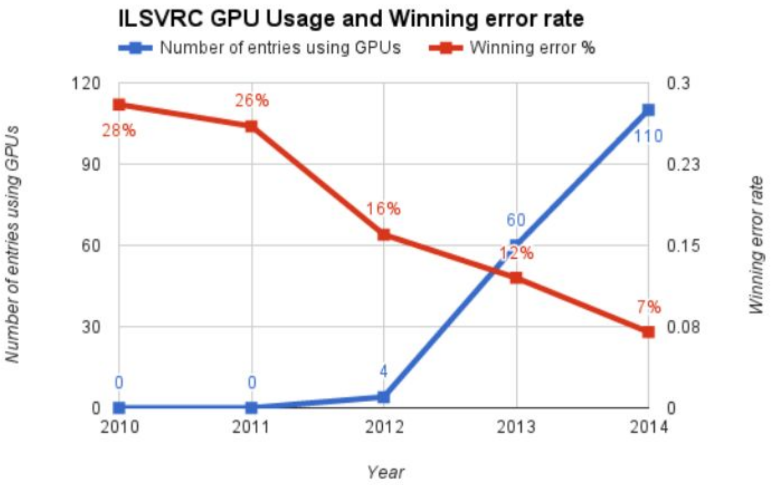
Steven Farrell
Shashank Subramanian
Data, AI, and Analytics Services

The Deep Learning revolution









AI is transforming science

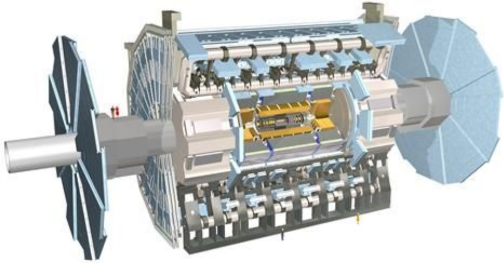
Across all domains

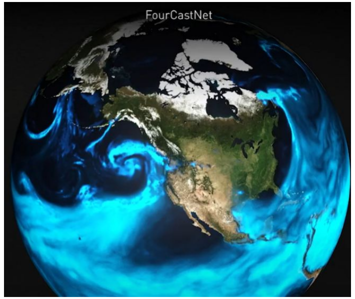
- Especially those with Big Data

Across *many* application areas

- Analyzing data better, faster
- Accelerating expensive simulations
- Control + design of complex systems

Embraced by the DOE and other funding agencies

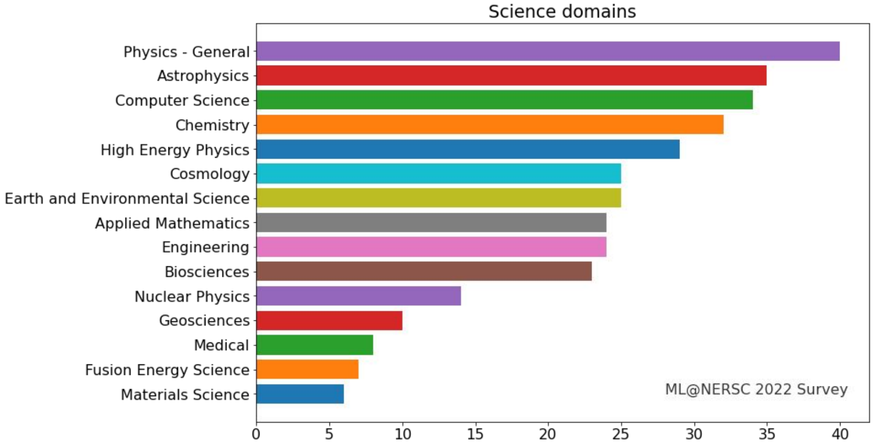


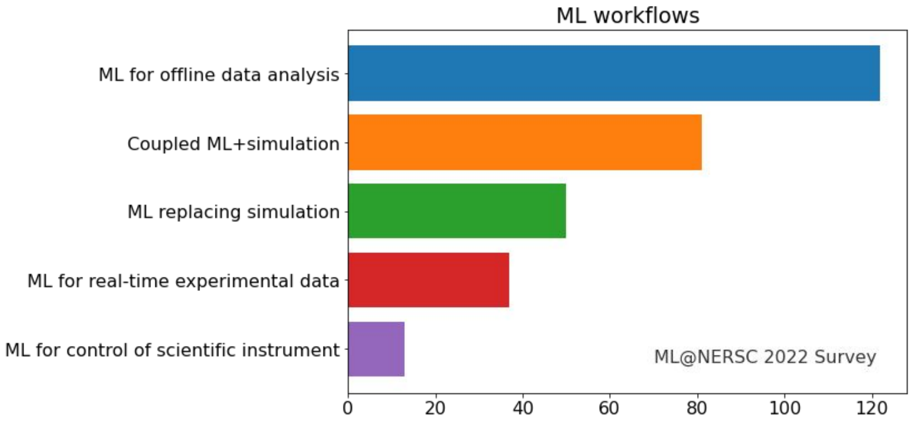


National Science Foundation

National Artificial Intelligence Research Institutes

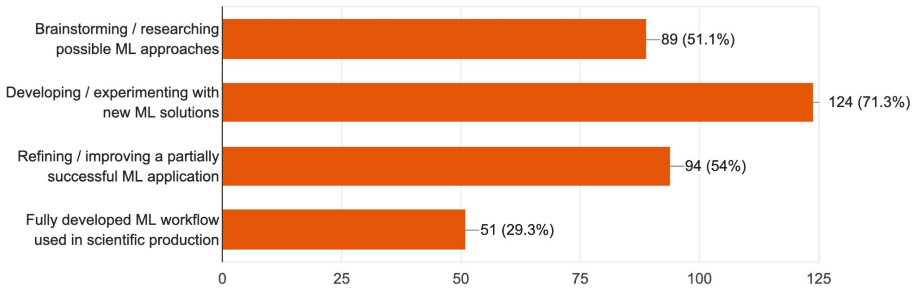
Scientific AI users

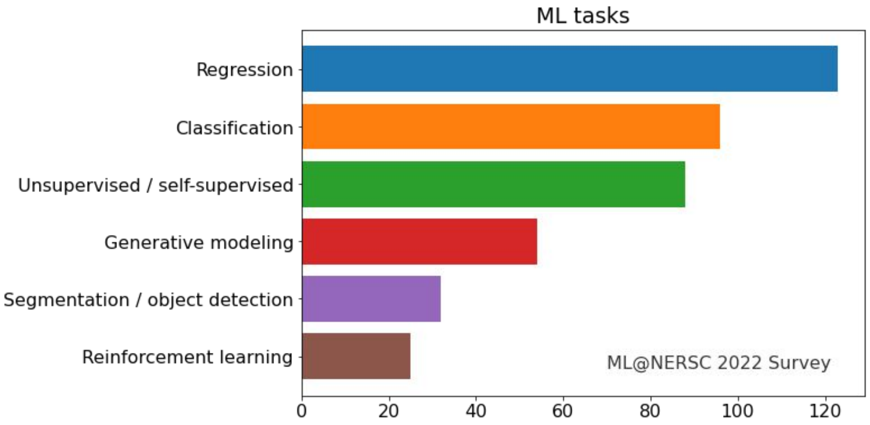




What is the level of maturity of ML in your research? (mark all that apply to your projects)

174 responses





BERKELEY LAB
Bringing Science Solutions to the World
U.S. DEPARTMENT OF ENERGY
Office of Science

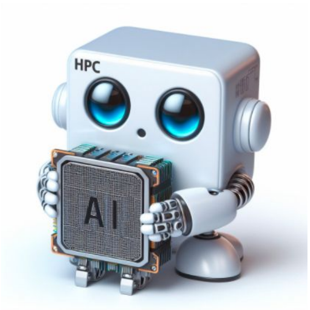
The need for HPC

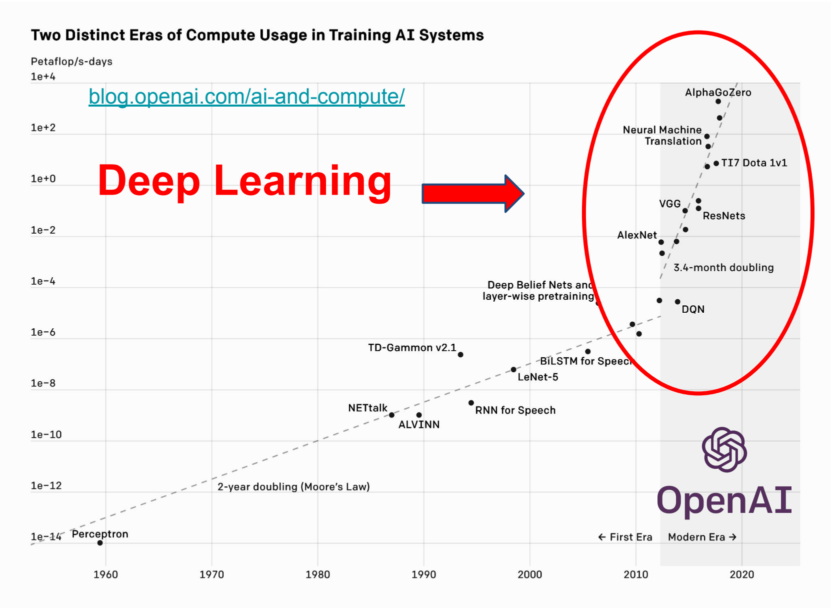
Growing computational cost of training AI models

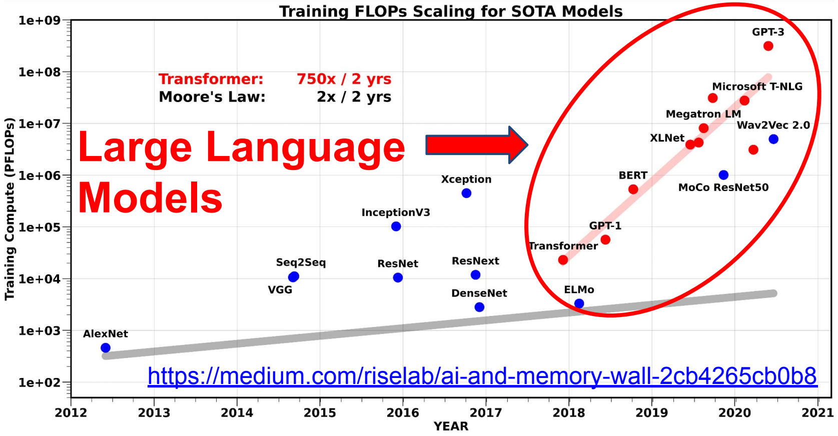
- bigger datasets + models, more complexity

Researchers need large scale resources

- Rapid iteration, reduce time to discovery

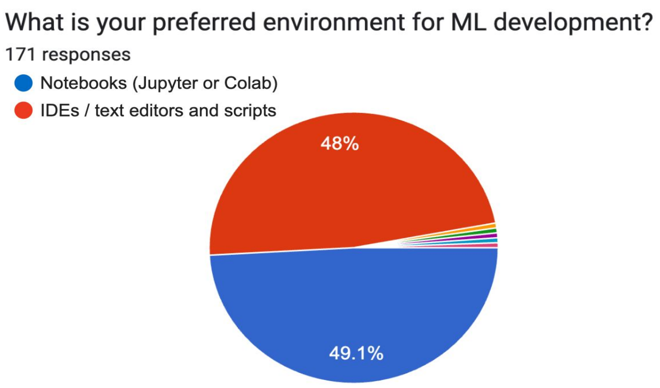


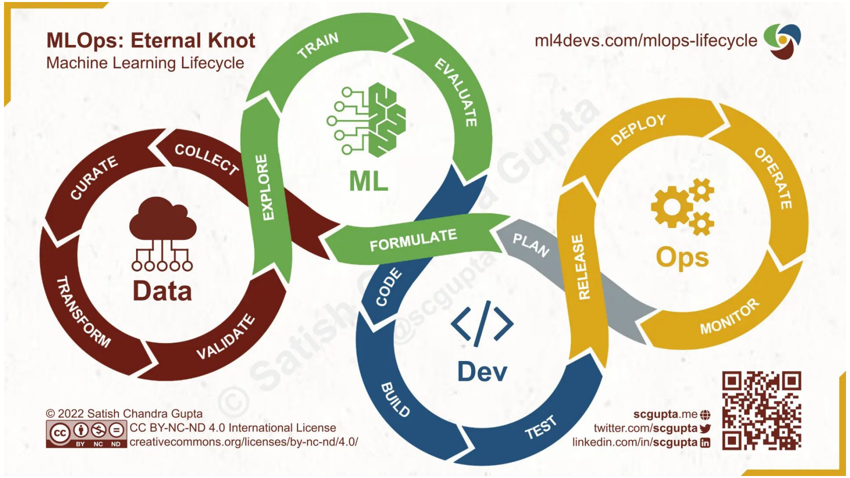




The AI for Science lifecycle

- **Experimentation**
  - Jupyter, interactive sessions
  - Data engineering
  - Testing architecture types
- **Full scale training, hyperparameter tuning, validation**
  - Batch jobs
  - Parallelism
- **Deployment**
  - Offline/online data processing
  - Streaming, as-a-service





NERSC AI Strategy

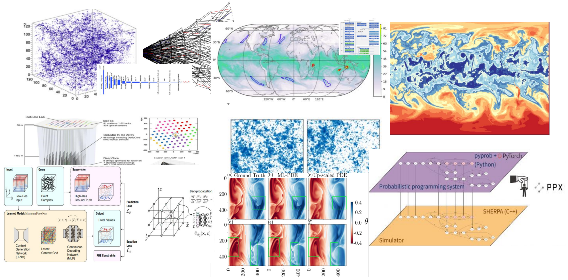
Deployment

Automation | Interactivity

Software Frameworks and Libraries

Systems w/ Accelerators

Methods and Applications



Empowerment

SC21 St. Louis, MO science & beyond.

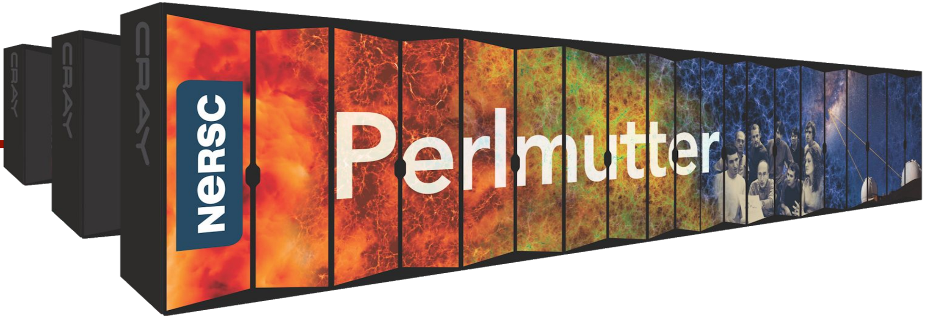
ISC

- ***Deploy*** optimized hardware and software systems
- ***Apply*** AI for science using cutting-edge methods
- ***Empower*** through seminars, workshops, training and schools

Perlmutter

35 PB
All-Flash
Scratch

>5 TB/s



#12, 113PF Peak

Off Platform Storage

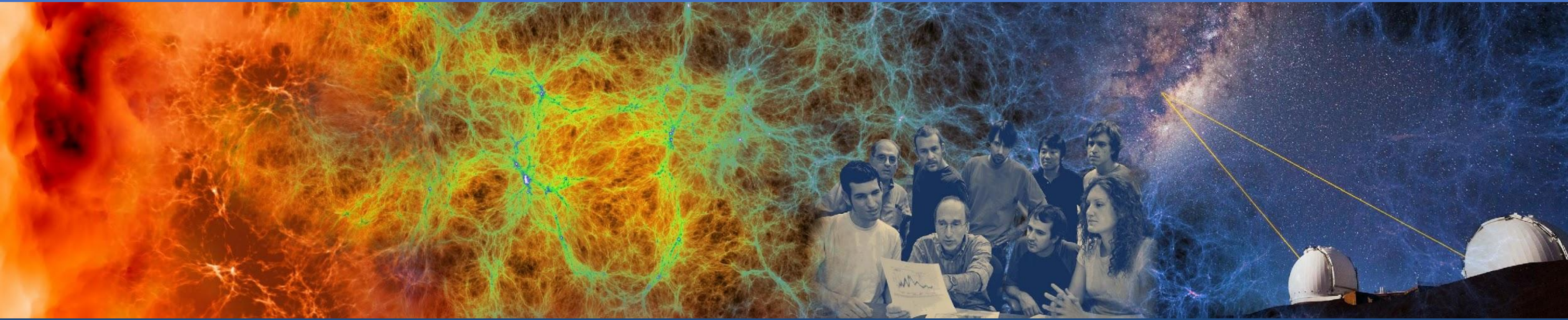
HPSS Tape Archive >1 EB
Community File System 240 PB
/home 450 TB

1,792 GPU-accelerated nodes
- 4 NVIDIA A100 GPUs + 1 AMD "Milan" CPU
- 448 TB (CPU) + 320 TB (GPU) memory

3,072 CPU-only nodes
- 2 AMD "Milan" CPUs
- 1,536 TB CPU memory

HPE Slingshot 11 ethernet-compatible interconnect
4 NICs/GPU node,
1 NIC/CPU node

Deep Learning on Perlmutter: Software stack and best practices



NERSC

BERKELEY LAB

U.S. DEPARTMENT OF ENERGY | Office of Science

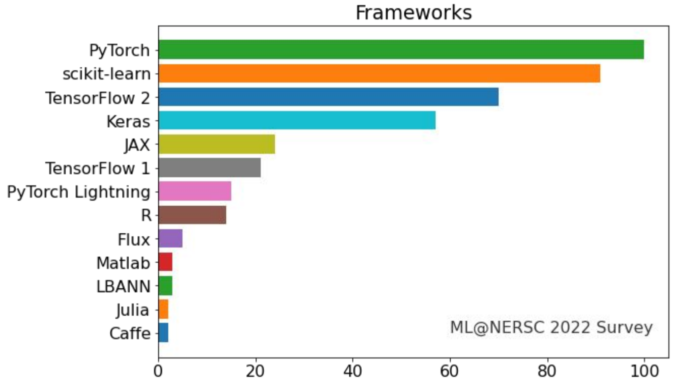
Perlmutter deep learning software stack overview

General strategy:

- Provide functional, performant installations of the most popular frameworks and libraries
- Enable flexibility for users to customize and deploy their own solutions

Frameworks:

PyTorch Keras TensorFlow



Distributed training libraries:

- PyTorch distributed
- NCCL, MPI
- Horovod

https://docs.nersc.gov/machinelearning/

Productive tools and services:

- Jupyter, Shifter

How to use the Perlmutter DL software stack

We have modules you can load which contain python and DL libraries:

```
module load pytorch/2.1.0-cu12
module load tensorflow/2.15.0
```

Check which software versions are available with:

```
module spider pytorch
```

You can install your own packages on top to customize:

```
pip install --user MY-PACKAGE
```

Or, clone a conda environment from our modules:

```
conda create -n my-env --clone /path/to/module/installation
```

Or, create custom conda environments from scratch:

```
conda create -n my-env MY-PACKAGES
```

More on how to customize your setup can be found in the docs (PyTorch, TensorFlow).

Containerized DL: using Shifter on Perlmutter

NERSC currently supports [containers with Perlmutter via Shifter](#)

- Easy, performant: our top500 entry used a container!

To see images currently available:

```
shifterimg images | grep pytorch
```

To pull desired docker images onto Perlmutter:

```
shifterimg pull <dockerhub_image_tag>
```

To use interactively:

```
shifter --module gpu --image=nersc/pytorch:ngc-23.07-v1
```

Use Slurm image shifter options for best performance in batch jobs:

```
#SBATCH --image=nersc/pytorch:ngc-23.07-v1
#SBATCH --module=gpu,nccl-2.18
srun shifter python my_python_script.py
```

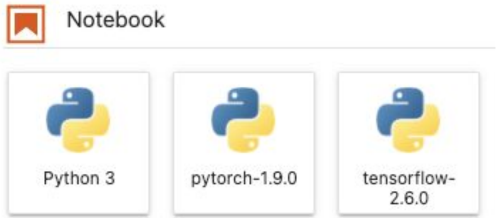
Jupyter for deep learning

JupyterHub service provides a rich, interactive notebook ecosystem on Cori

- Very popular service with thousands of users
- A favorite way for users to develop ML code

Users can run their deep learning workloads

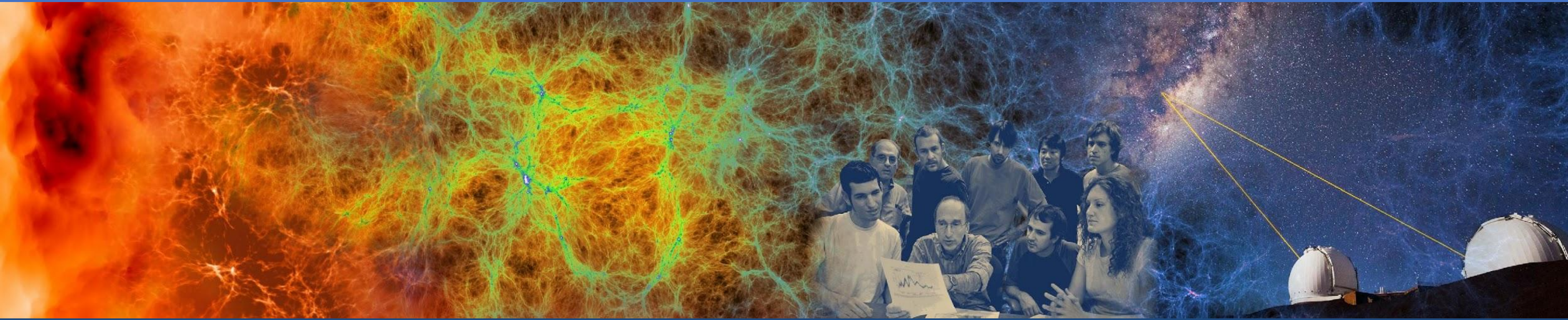
- on dedicated Perlmutter GPU nodes
- using our pre-installed DL software kernels
- [using their own custom kernels]()



	Shared CPU Node	Shared GPU Node	Exclusive GPU Node	Exclusive Large Memory Node	Configurable GPU	Configurable DGX
Perlmutter	start		start		start	
Cori	start	start		start	start	
Resources	Use a node shared with other users' notebooks but outside the batch queues.		Use your own node within a job allocation using defaults.		Use multiple compute nodes with s	
Use Cases	Visualization and analytics that are not memory intensive and can run on just a few cores.		Visualization, analytics, machine learning that is compute or memory intensive but can be done on a single node.		Multi-node analytics jobs, jobs in reservations, custom project charging, and more.	

Distributed Deep Learning

Reference material: [SC23 Deep Learning at Scale Tutorial]



NERSC

BERKELEY LAB

U.S. DEPARTMENT OF ENERGY | Office of Science

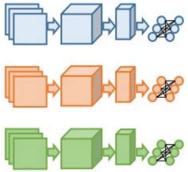
General strategy for optimizing deep learning at NERSC

Start with an appropriate model which trains on a single CPU or GPU

Optimize the single-node / single-GPU performance


- Using performance analysis tools
- Tuning and optimizing the data pipeline
- Make effective use of the hardware (e.g. mixed precision)

Distribute the training across multiple processors

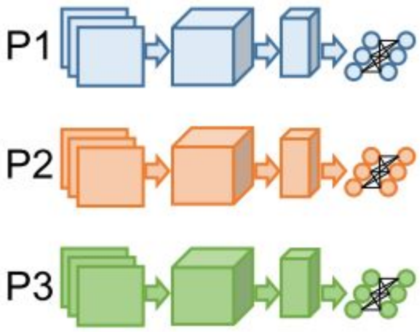


- Multi-GPU, multi-node training: data and/or model parallel
- Use best practices for large scale training and convergence
- Use best optimized libraries for communication, tune settings

Advanced parallelism

- Model/hybrid parallelism design considerations
- Implementation & analysis

Parallel training strategies



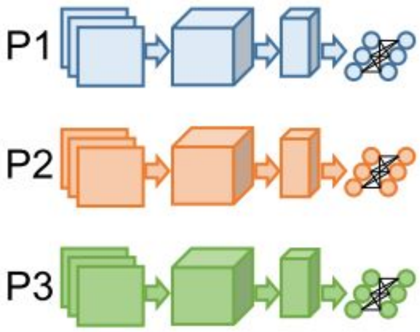
Data Parallelism

- Distribute input samples
- Model replicated across devices
- Most common

Model Parallelism

- Distribute network structure, within or across layers
- Needed for massive models that don't fit in device memory
- Becoming more common

Parallel training strategies



Data Parallelism

- Distribute input samples
- Model replicated across devices
- Most common

✅ Conceptually simple
✅ Easy implementation
- PyTorch, TensorFlow have built-in functionality

⚠️ Some additional considerations
- Data loading at scale
- Modified hyperparameters

Data parallelism

Batches are sharded across GPUs

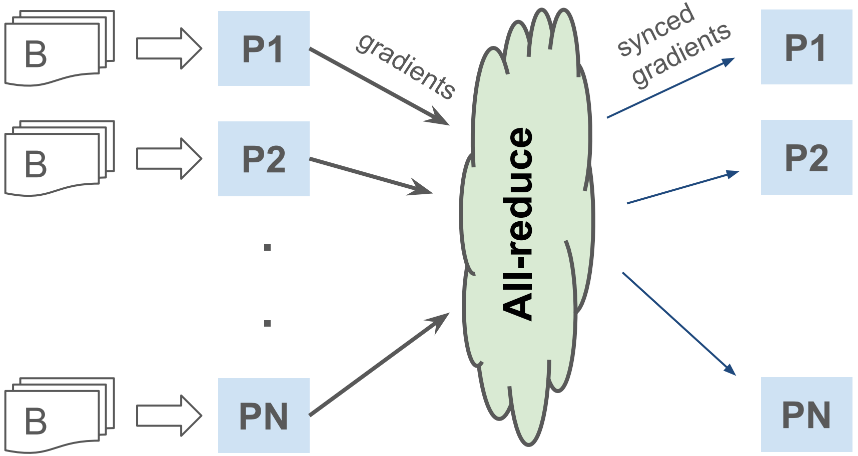
- Local batch-size = B
- Global batch-size = N * B

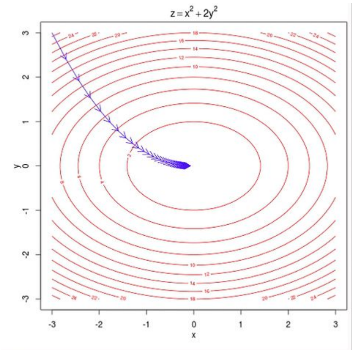
Gradients averaged across GPUs via all-reduce calls

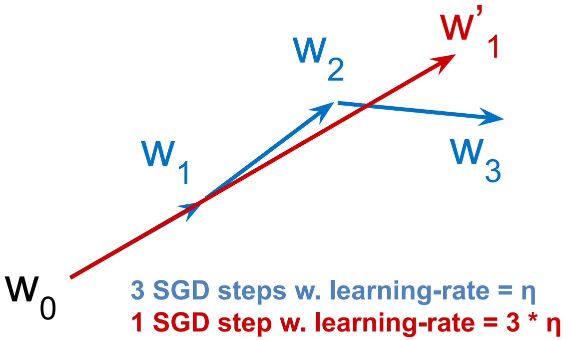
- Incurs communication cost
- Can be partially overlapped (hidden) by computation

Speed up model training by scaling

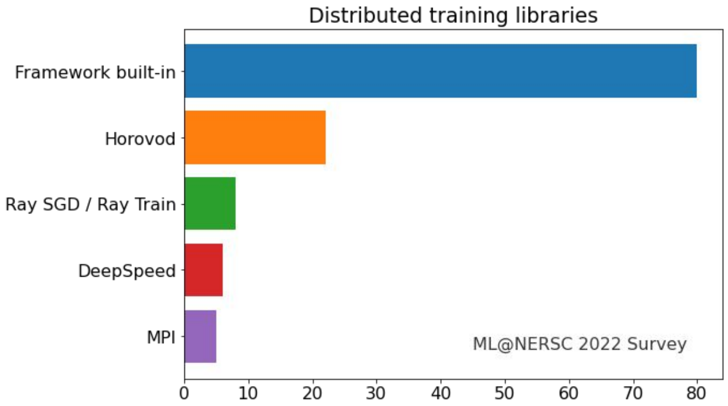
- More GPUs => larger batch size
- Increase learning rates for larger, faster steps to convergence







Distributed Training Tools



Framework built-in

- PyTorch DistributedDataParallel (DDP)
- TensorFlow Distribution Strategies

Other popular libraries

- **Lightning:** DDP + convenient features
- **DeepSpeed:** ZeRO optimizations, 3D parallelism
- **HuggingFace accelerate**: DDP + features
- **Ray:** DDP + HPO
- **Horovod:** MPI+NCCL, easy to use, examples
- **LBANN:** multi-level parallelism, ensemble learning, etc., docs

Communication backends

- NCCL is the backend of choice for GPU nodes on Perlmutter
- The NCCL OFI plugin (from AWS) enables RDMA performance on the libfabric-based Perlmutter Slingshot network (see our docs)

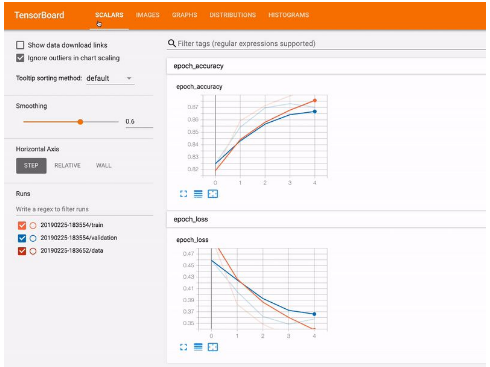
Workflow tools

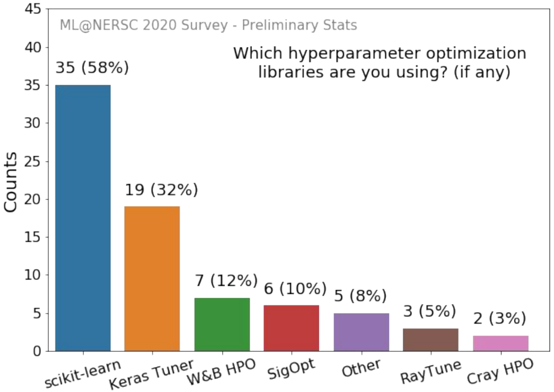
Some high level tools will be vital to your success as you scale up

- Hyper-parameter optimization (HPO) is critical for getting the most out of your models and data, but can be complex and computationally expensive
- Experiment tracking and visualization tools make your work reproducible, shareable, and more interpretable

Helpers / examples / docs

- NERSC HPO docs
- W&B template (new)
- Ray cluster helper (new)
- Tensorboard jupyter launcher





RAY

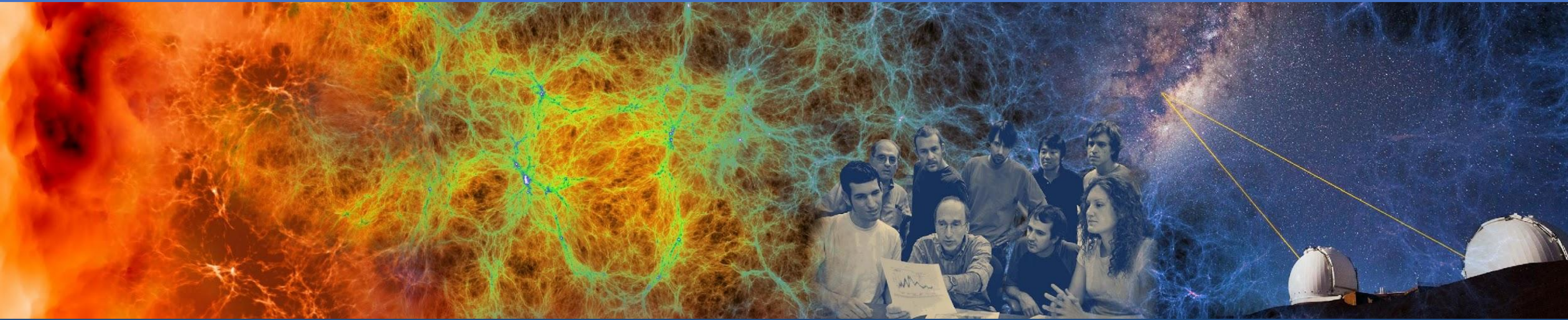
tune

Weights & Biases

OPTUNA

SIGOPT

Outreach & additional resources



Training events

The Deep Learning for Science School at Berkeley Lab (https://dl4sci-school.lbl.gov/)

- Comprehensive program with lectures, demos, hands-on sessions, posters
- 2019 material (videos, slides, code) online: https://sites.google.com/lbl.gov/dl4sci2019
- 2020 webinar series material: https://dl4sci-school.lbl.gov/agenda

The Deep Learning at Scale Tutorial

- Jointly organized with NVIDIA (+ previously Cray, ORNL)
- Presented at SC18-23, ECP Annual 2019, ISC19
- Detailed lectures + hands-on material covering distributed training, scaling, profiling, and optimization on Perlmutter
- See the full SC23 material here

NERSC training events

- NERSC-NVIDIA LLM Bootcamp 2024 (Apply now!)
- NVIDIA AI for Science Bootcamp 2023
- Data Day 2024, New User Training Sep 2023

NERSC Data Seminar Series:

- https://github.com/NERSC/data-seminars
- https://www.youtube.com/playlist?list=PL20S5EeApOSvkewFIuz2scAEkbnBIIzYy

Conclusions

Deep learning for science is here and growing

- Powerful capabilities; enthusiastic community
- We're excited to see what you accomplish with it!

Perlmutter has a productive, performant software stack for deep learning

- Optimized frameworks and solutions for small to large scale DL workloads
- Support for productive workflows (Jupyter, HPO)

Join the NERSC Users Slack

Take the ML@NERSC 2024 Survey!!!

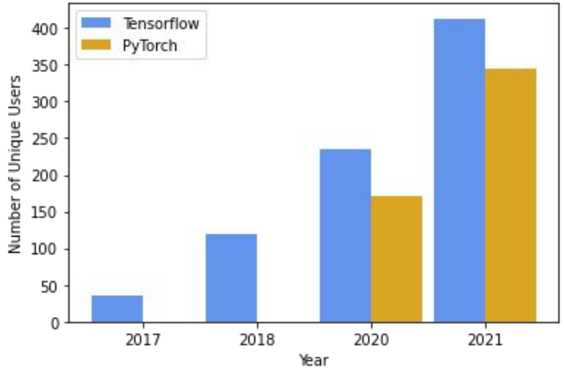
Thank You!
Next: run through of [GitHub material]

Growing scientific AI workload at NERSC

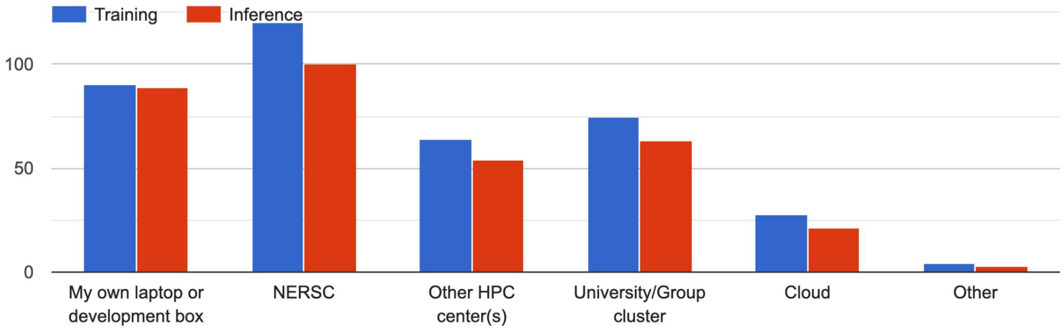
We track ML software usage

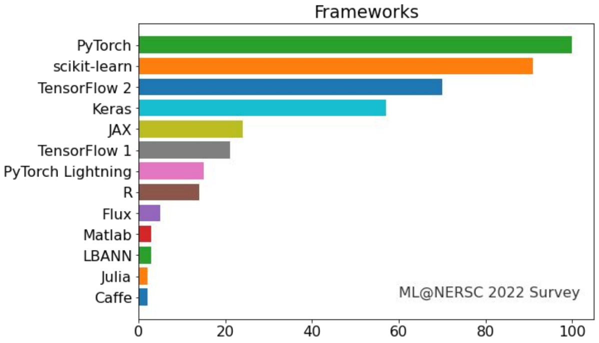
- Instrument user [python imports]
- DL users >10x from 2017 to 2021

Also track ML trends through 2-yearly survey



Where do you run your ML models (include future plans)?





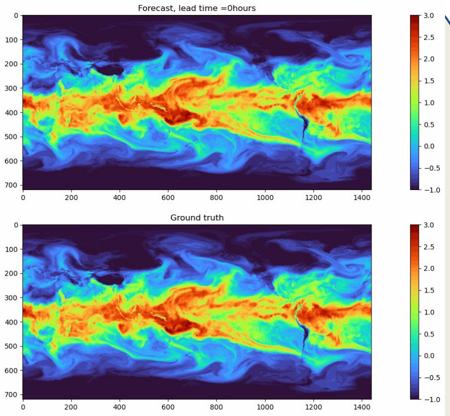
BERKELEY LAB
Bringing Science Solutions to the World
U.S. DEPARTMENT OF ENERGY
Office of Science

NESAP and Perlmutter are Enabling Adoption of Large-scale and Groundbreaking AI

FourCastNet

Pathak et al. 2022 **arXiv:2202.11214**
Collab with Nvidia, Caltech, … (+ now LBL EESA)

- Forecasts global weather at high-resolution.
- Prediction skill of numerical model; 10000s times faster



Jaideep Pathak
former NERSC
Postdoc now NVIDIA

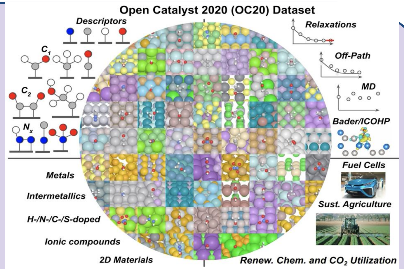
Shashank
Subramanian
Former NERSC
Postdoc now Staff

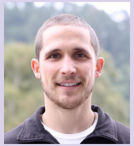
Jared Willard
NERSC Postdoc

CatalysisDL

Chanussot et al. 2021
Collab with CMU, MetaAI, …
arXiv:2010.09990

- NeurIPS 2021-23 Competitions
- Pre-trained models now used with DFT - e.g. FineTuna; AdsorbML





Brandon Wood
former NERSC
Postdoc now Meta AI

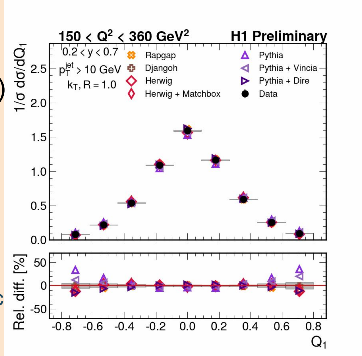
Wenbin Xu
NERSC postdoc

HEP-ML

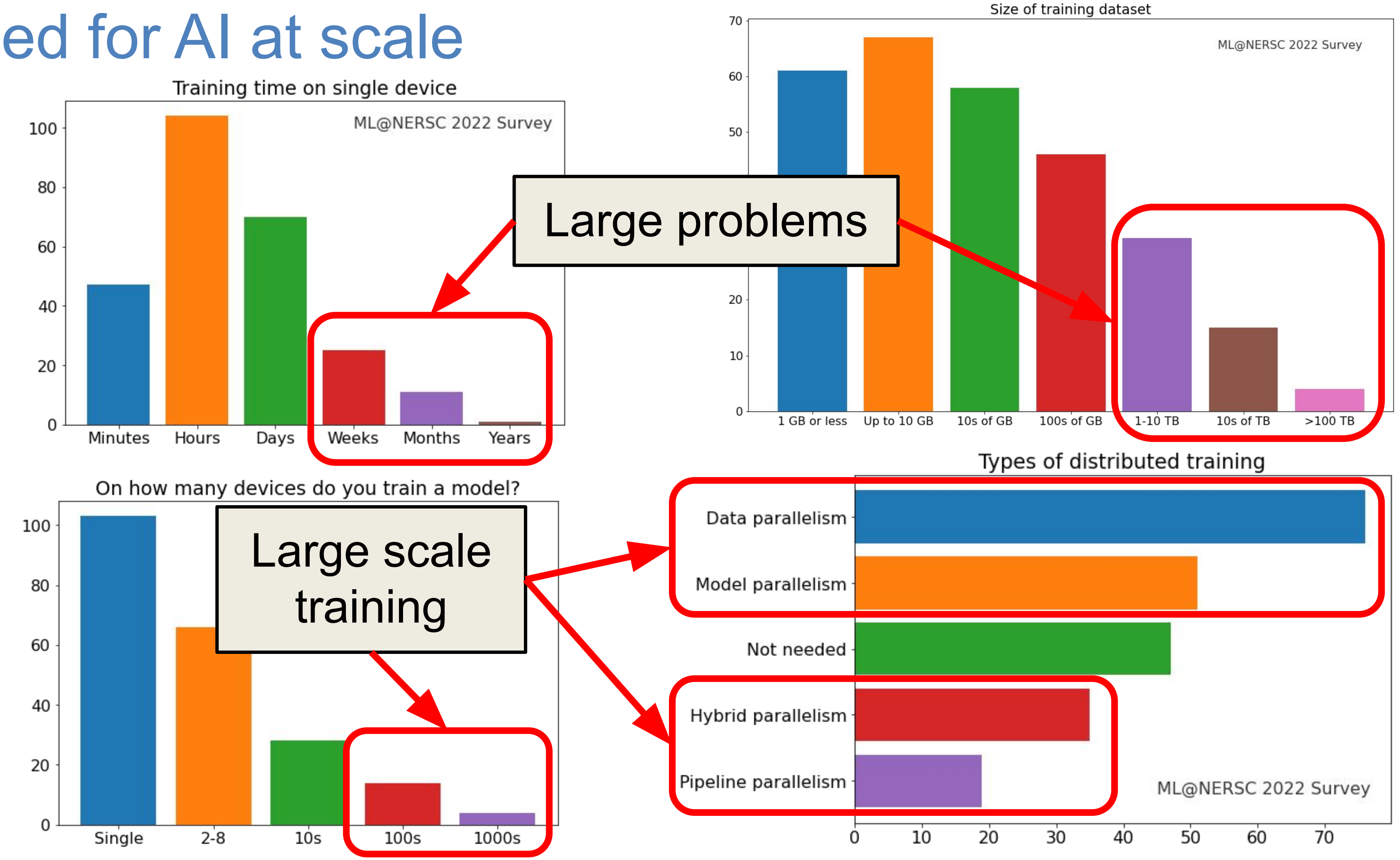
Collab with LBL Physics division (and H1 Collaboration)

- AI “Unfolding” extracts new physics insights from data
  - Requires Perlmutter for 1000s of UQ runs

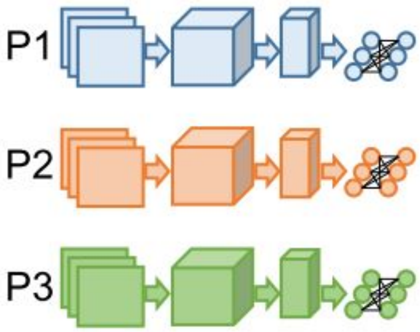
Vinicius Mikuni
NERSC Postdoc

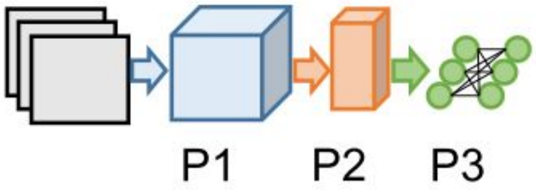


Need for AI at scale



Deep Learning parallelization strategies





Data Parallelism

Distribute input samples.

Model (tensor) Parallelism

Distribute network structure (layers).

Layer Pipelining

Partition by layer.

Fig. credit: arXiv:1802.09941

Hybrid parallelism example: Megatron-Turing NLG 530B

Best Practices for DL + Shifter on Perlmutter

NVIDIA provides containers optimized for deep learning on GPUs with

- Pytorch or TensorFlow+Horovod
- Optimized drivers, CUDA, NCCL, cuDNN, etc
- Many different versions available

NVIDIA. NGC

We also provide images based on NVIDIA's, which have a few useful extras

You can also build your own custom containers (easy to build on top of NVIDIA's)

Notes

- Customization: from inside the container, do `pip install --user MY-PACKAGE` (make sure to set `$PYTHONUSERBASE` to a custom path for the desired container)

- NVIDIA NGC containers use OpenMPI, which requires specific options if you require MPI. Instructions: https://docs.nersc.gov/development/shifter/how-to-use/#shifter-mpich-module

General guidelines for deep learning at NERSC

NERSC documentation: https://docs.nersc.gov/analytics/machinelearning/overview/

Use our provided modules/containers if appropriate

- They have the recommended builds and libraries tested for functionality and performance
- We can track usage which informs our software support strategy

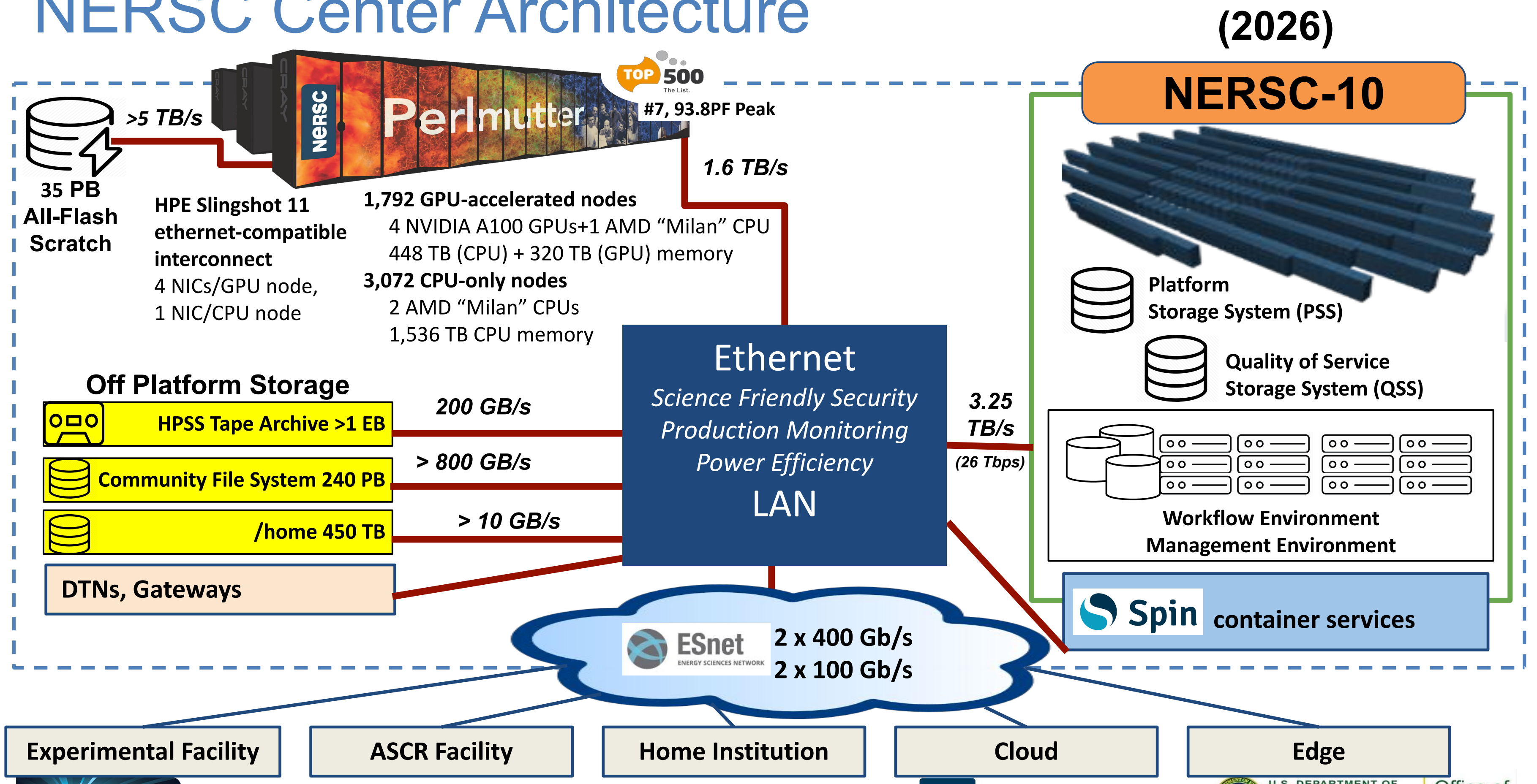
For developing and testing your ML workflows

- Use interactive QOS or Jupyter for on-demand compute resources
- Visualize your models and results with TensorBoard or Weights & Biases

For performance tuning

- Check cpu/gpu utilization to indicate bottlenecks (e.g. with top, nvidia-smi)
- Data pipeline is the most common source of bottlenecks
    - Use framework-recommended APIs/formats for data loading
    - Use multi-threaded data loaders and stage data if possible
- Profile your code, e.g. with Nvidia Nsight Systems or TensorBoard Profiler

NERSC Center Architecture



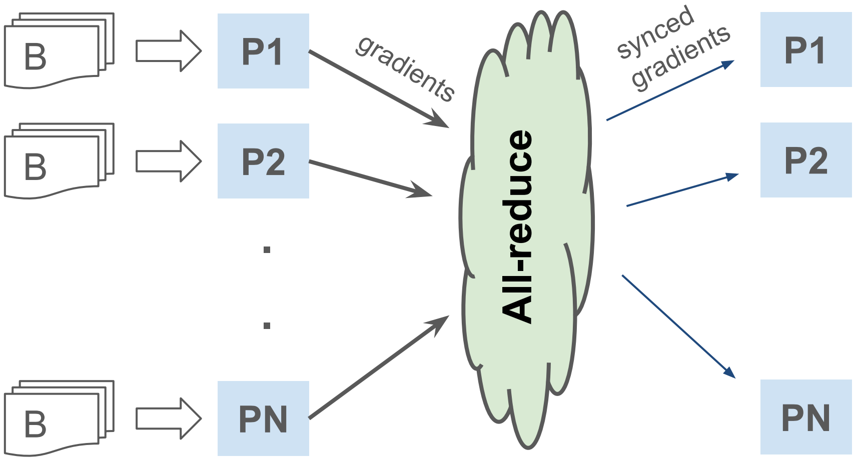
Synchronous data parallel scaling

Weak scaling (fixed local batch size)

- Global batch size grows with number of workers
- Computation grows with communication; good scalability
- Large batch sizes can negatively affect convergence

Strong scaling (fixed global batch size)

- Local batch size decreases with number of workers
- Convergence behavior unaffected
- Communication can become a bottleneck



Local batch-size = B

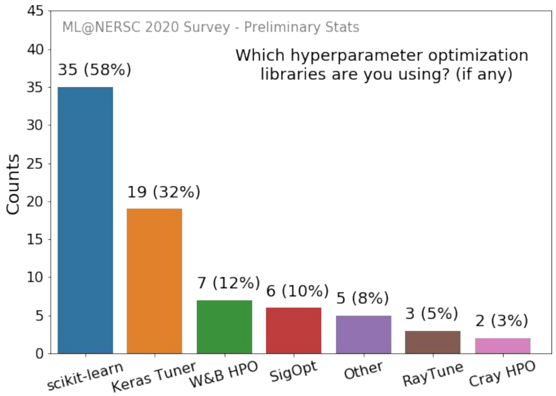
Global batch-size = N * B

Hyper-parameter optimization (HPO) solutions

https://docs.nersc.gov/machinelearning/hpo/

Model selection/tuning are critical for getting the most out of deep learning

- Many methods and libraries exist for tuning your model hyper-parameters
- Usually very computationally expensive because you need to train many models => Good for large HPC resources



Helpers / examples

- W&B template (new)
- Ray cluster helper (new)

Users can use whatever tools work best for them

- Ask us for help if needed!

RAY tune Weights & Biases OPTUNA SIGOPT

TensorBoard at NERSC

TensorBoard is the most popular tool for visualizing and monitoring DL experiments, widely adopted by TensorFlow and PyTorch communities.
We recommend running TensorBoard in Jupyter using nersc-tensorboard helper module.

```
import nersc_tensorboard_helper
%load_ext tensorboard
%tensorboard --logdir YOURLOGDIR --port 0
```

then get an address to your TensorBoard GUI:

```
nersc_tensorboard_helper.tb_address()
```