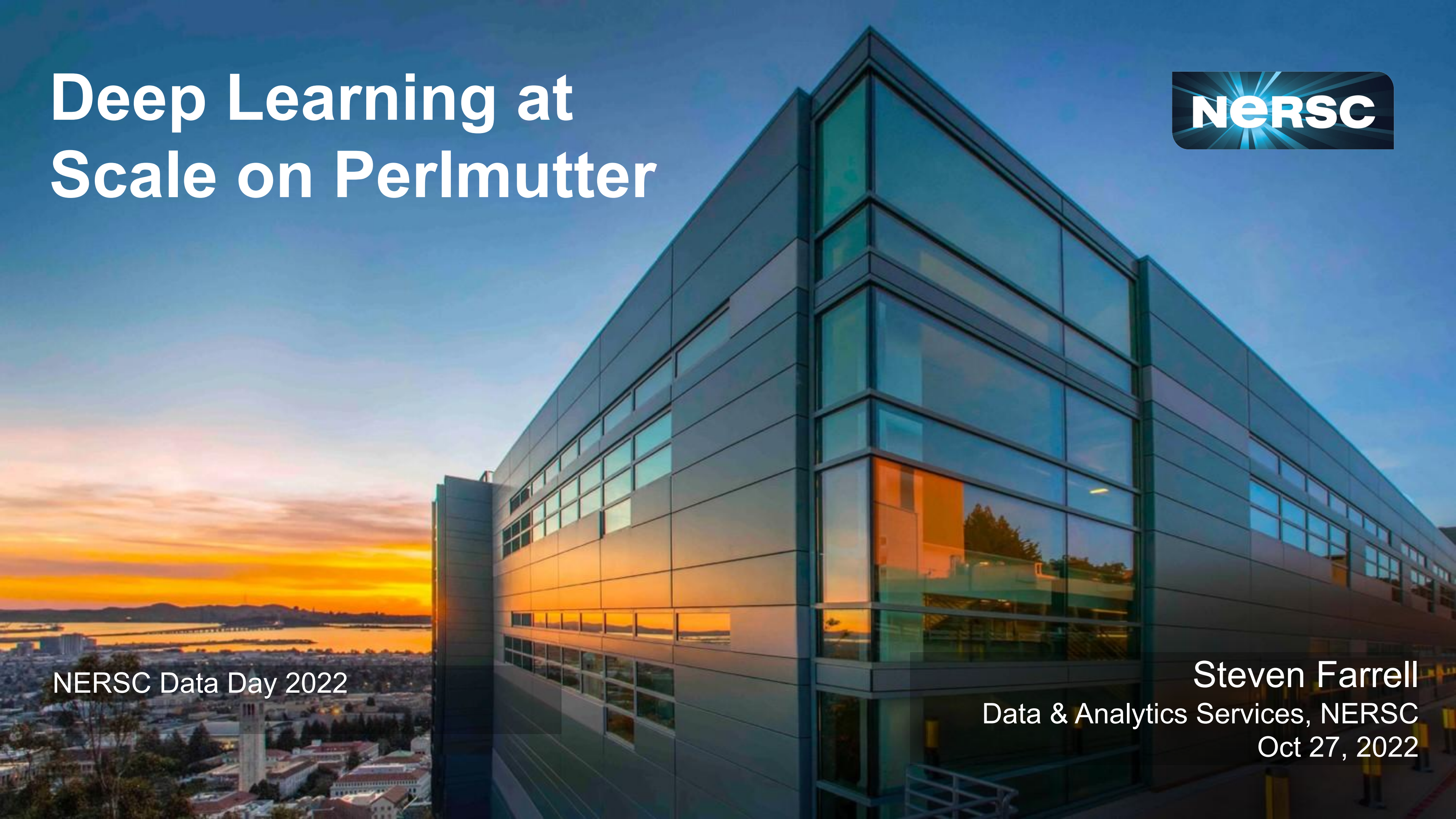
Deep Learning at Scale on Perlmutter



NERSC Data Day 2022

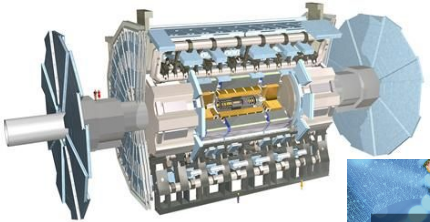
Steven Farrell
Data & Analytics Services, NERSC

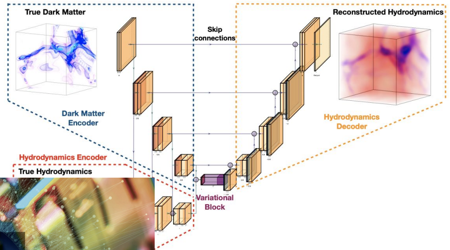
Oct 27, 2022

AI is transforming science

AI/ML/DL have powerful capabilities for scientific workflows

- Analysis of large datasets
- Acceleration of expensive simulations
- Control of complex experiments





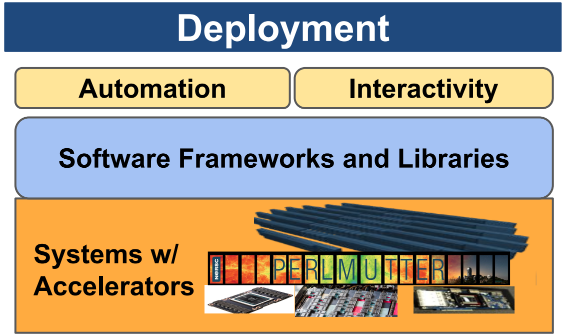
Scientists (and the DOE) are enthusiastic about AI

- Lots of R&D, methods and tools rapidly evolving
- Anticipation for a future DOE AI4Science project
- Some areas moving into maturity

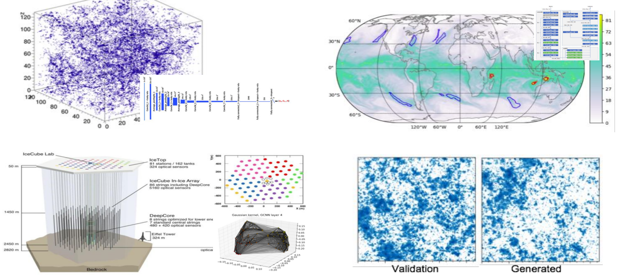
AI4Science workloads increasingly need large computational resources

- Problems, datasets, models growing in size and complexity
- HPC centers like NERSC can play an important role

NERSC AI Strategy



Methods and Applications



Empowerment

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

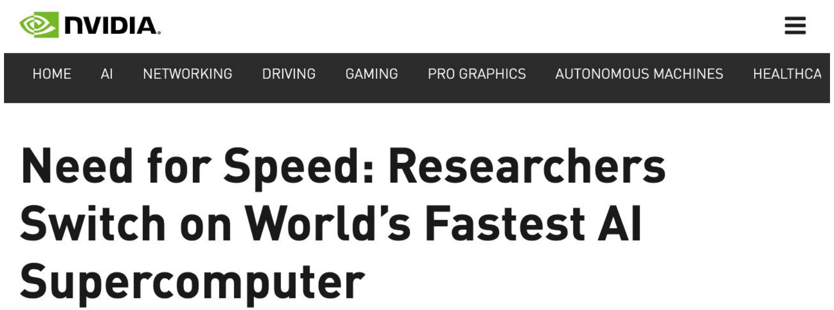
Perlmutter: A Scientific AI Supercomputer

HPE/Cray Shasta system

Phase 1 (in early science phase):
- 12 GPU cabinets with 4x NVIDIA [Ampere]() GPU nodes; Total >6000 GPUs
- 35 PB of All-Flash storage

Phase 2 (2022):
- 12 AMD CPU-only cabinets
- HPE/Cray Slingshot high performance network

Optimized software stack for AI
Application readiness program (NESAP)


NVIDIA.
HOME AI NETWORKING DRIVING GAMING PRO GRAPHICS AUTONOMOUS MACHINES HEALTHCA

Need for Speed: Researchers Switch on World's Fastest AI Supercomputer

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

NERSC AI software

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

We build optimized modules for

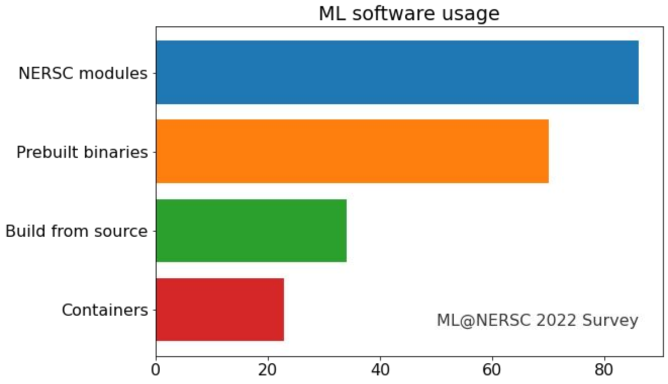
- Python
- PyTorch (pytorch-distributed + NCCL)
- TensorFlow (horovod + NCCL)

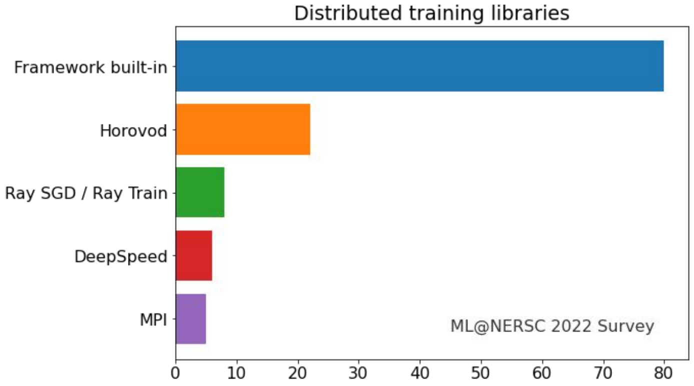
We support optimized containers via Shifter

- NGC DL images
- User images

Users can use their own environments

- conda, etc.





NERSC AI software

Hyperparameter optimization

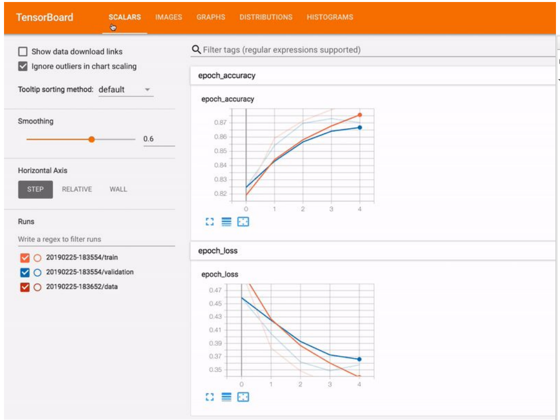
- Most tools should work
- We use Ray Tune, Weights & Biases, etc.

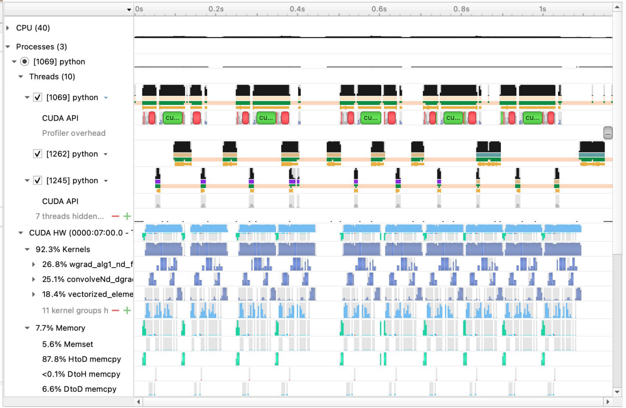
Jupyter

- Popular for developing and training models

Profiling and visualization

- NVIDIA profiler (nsight)
- Tensorboard
- Weights & Biases

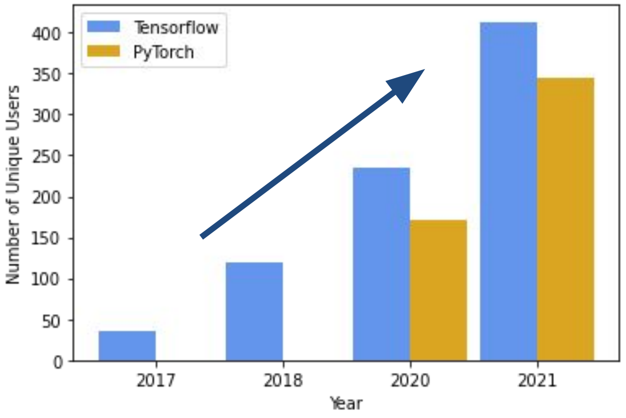




Growing scientific AI workload at NERSC

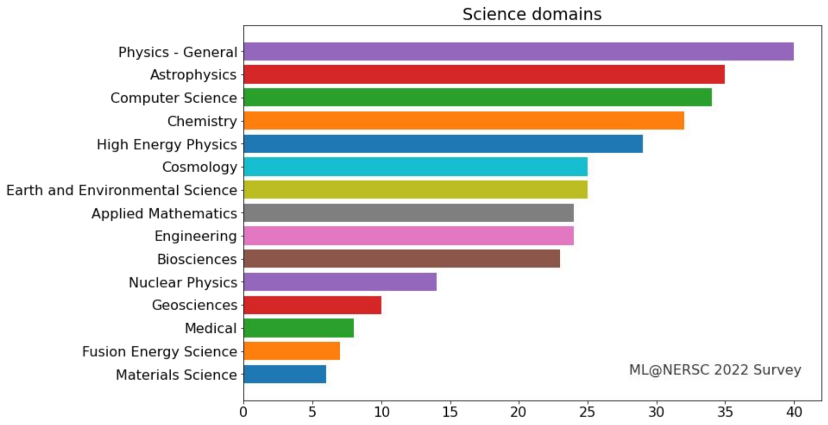
We track ML software usage

- Module loads and python imports
- Users of DL frameworks increased more than 6x from 2018 to 2021

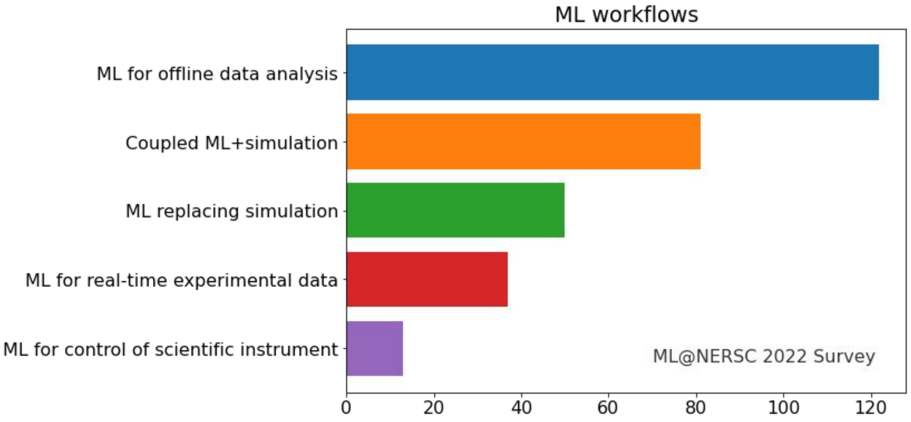


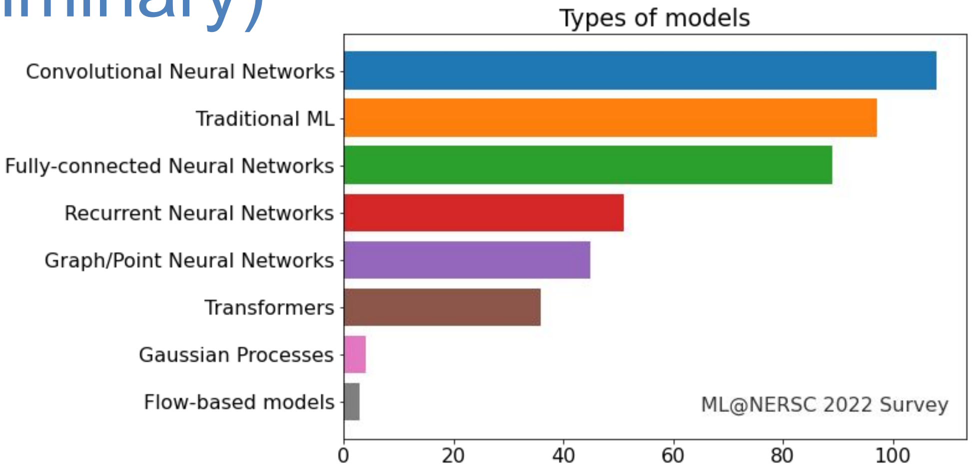
We track ML trends through 2-yearly survey

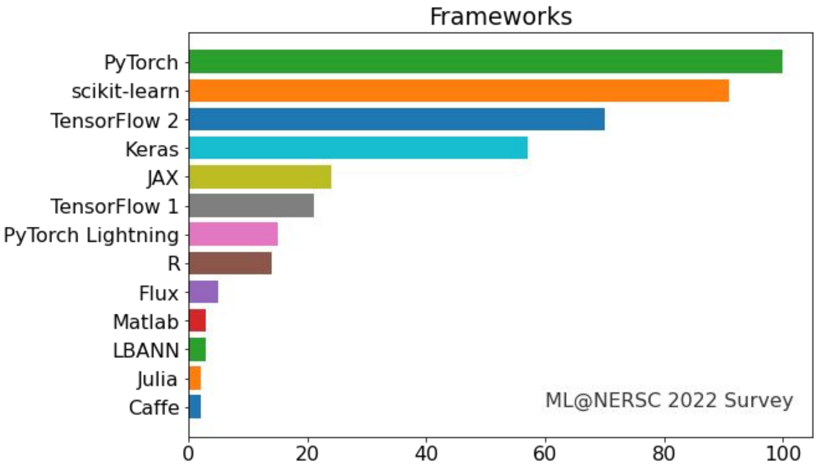
- Targets scientific communities potentially using HPC resources (not just NERSC)
- Covers problem type, workload, model architectures, scaling, hardware, software, and usage of NERSC software/resources
- ***Help us out by filling the 2022 survey:*** https://forms.gle/1CJ9x2ndXTfjsYfx9

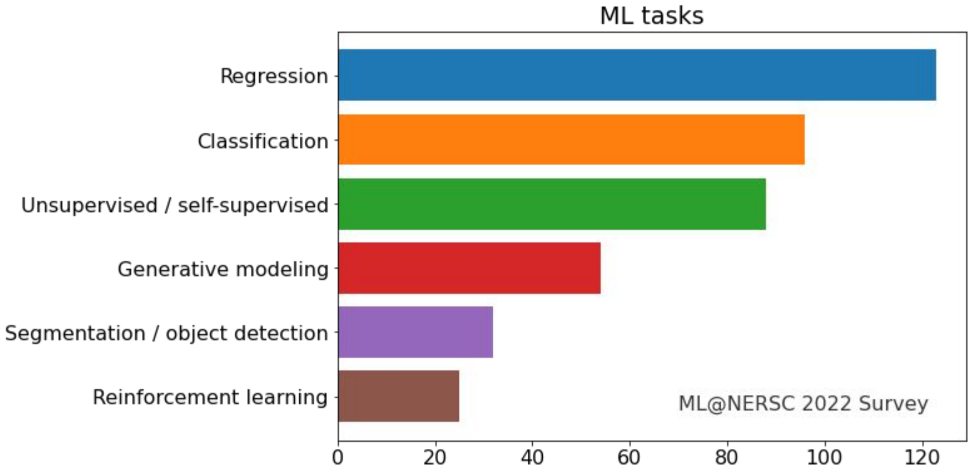


ML@NERSC Survey (preliminary)

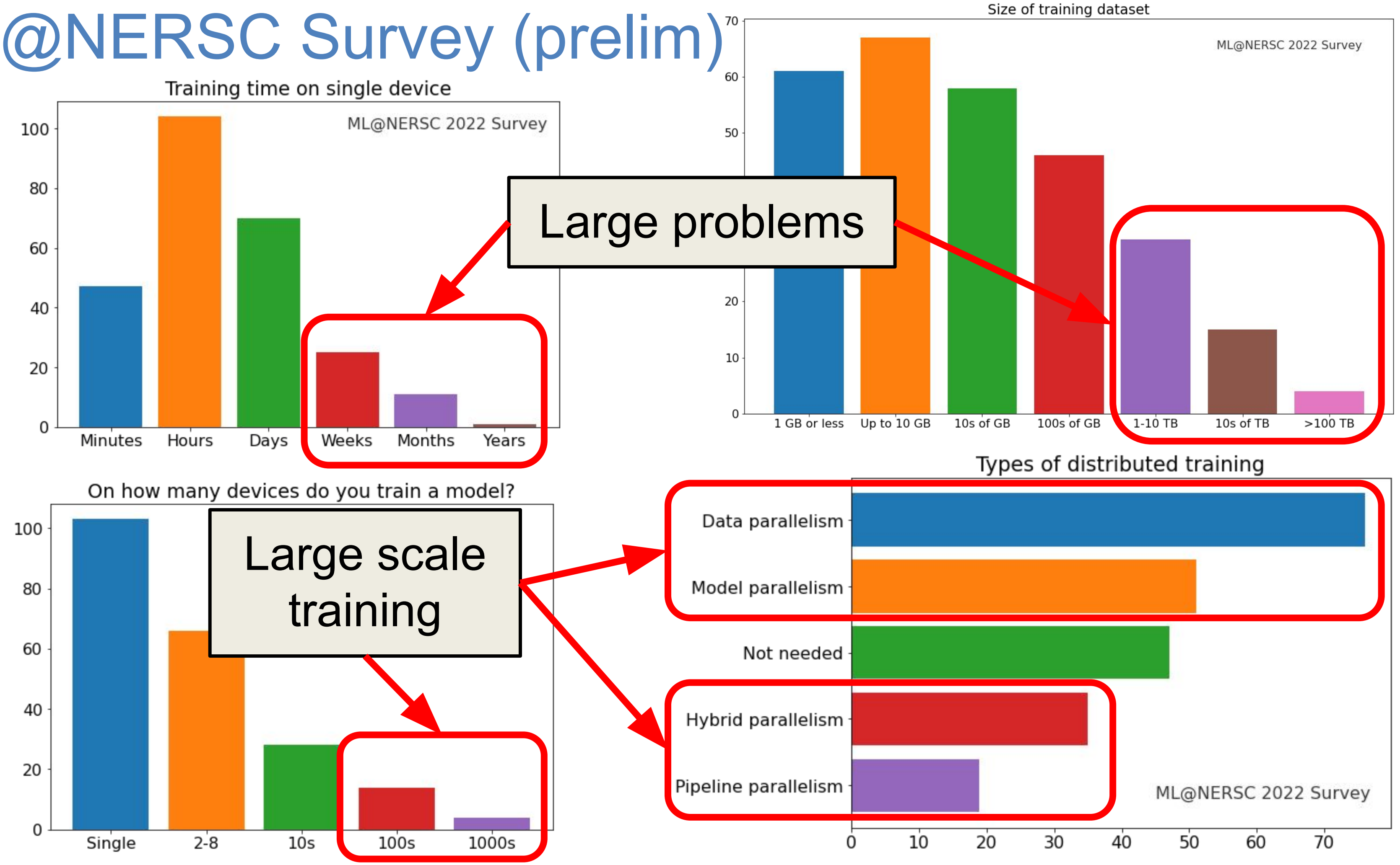








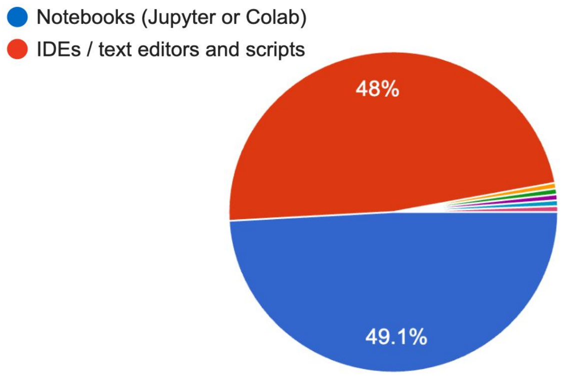
ML@NERSC Survey (prelim)



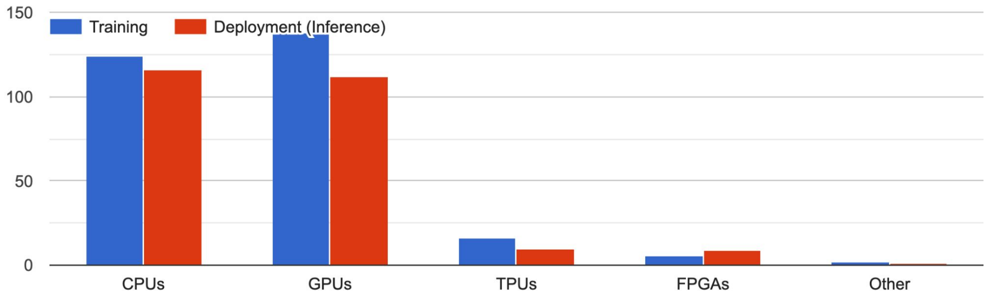
ML@NERSC Survey (preliminary)

What is your preferred environment for ML development?

171 responses

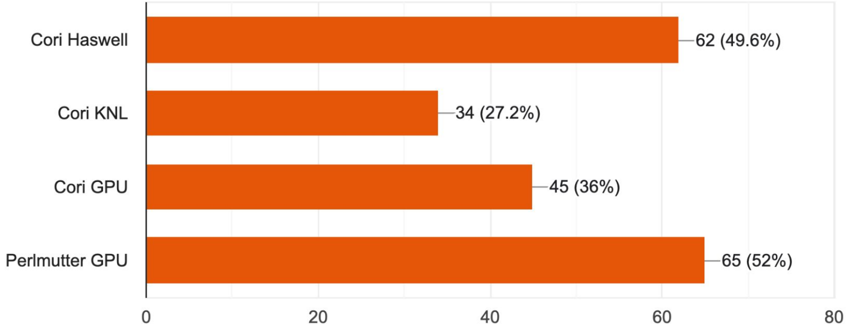


What hardware do you run your models on (include future plans)?



Which NERSC system(s) are you using for ML?

125 responses



- Jupyter very popular
- CPUs still used by many
- Trends in training vs. inference

Empowerment and training resources

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

- 2019 in-person lectures, demos, hands-on sessions, posters (videos, slides, code)
- 2020 summer webinar series. Recorded talks: https://dl4sci-school.lbl.gov/agenda

The Deep Learning at Scale Tutorial

- Since 2018, and with NVIDIA in 2020/21
- 2021 was first training event to use Perlmutter Phase 1 with hands-on material for distributed training
- See the full SC21 material here and videos
- Accepted again for SC22!

NVIDIA AI for Science Bootcamp - Aug 25-26, 2022

- View the agenda and slides

Other NERSC trainings

- New User Training, Data Day (***now!***), etc.

How to optimize DL workloads on HPC

Scientists need *fast* and *efficient* DL methods and tools

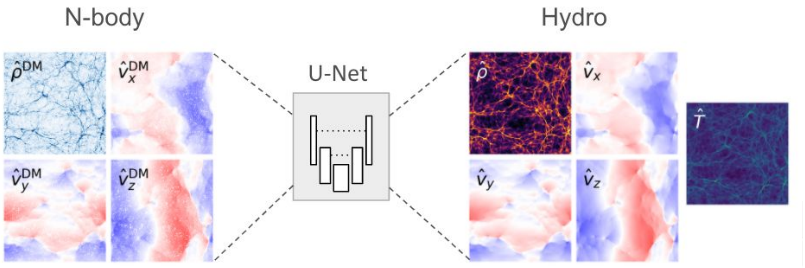
- to enable rapid development and testing of ideas
- for production workloads with computational constraints (e.g. realtime)
- to optimize overall system throughput for all NERSC users

Effective use of modern HPC systems can greatly accelerate DL workflows

- It’s getting easier, but can still be non-trivial

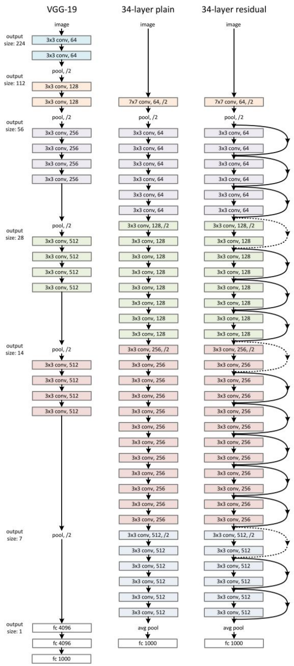
This material comes mostly from our Deep Learning at Scale Tutorial

- Most recently shown at SC21: https://github.com/NERSC/sc21-dl-tutorial
- Accepted at SC22 this year

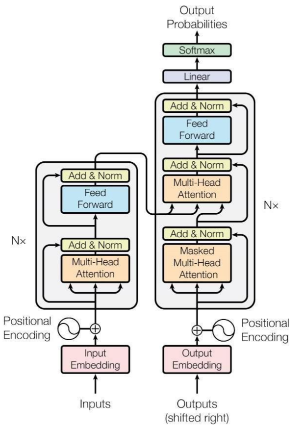


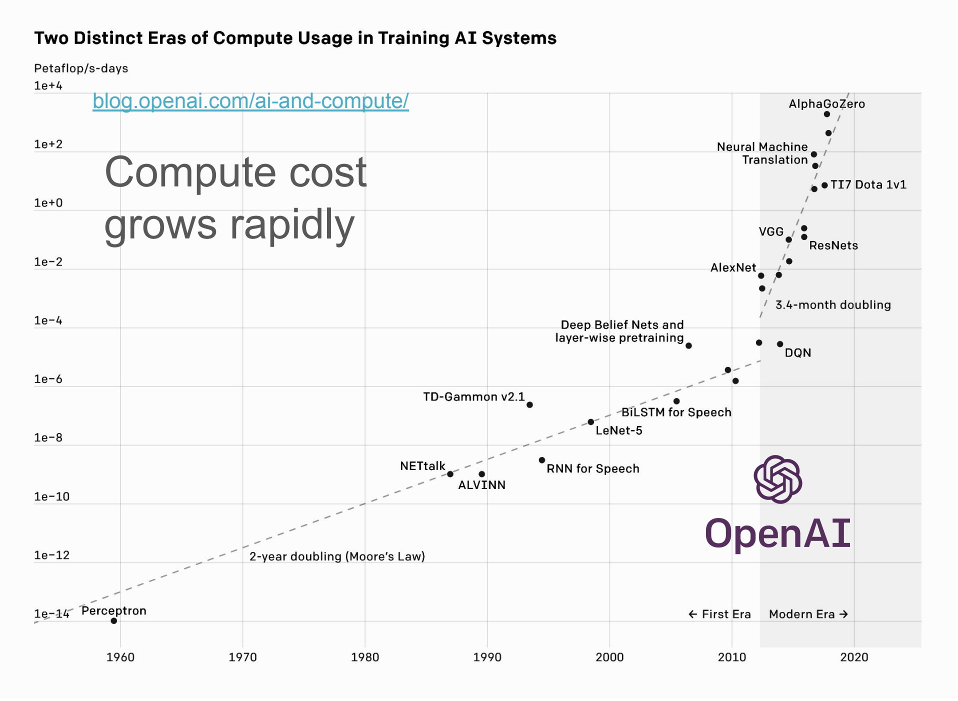
The need for HPC-scale resources

- Deep Learning (DL) is a powerful tool
- Deep Learning is computationally intensive (especially training)
- The compute requirements of DL are growing



Models get bigger and more complex





blog.openai.com/ai-and-compute/

Compute cost grows rapidly

How do we make effective use of HPC for Deep Learning training?

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, data-parallel and/or model-parallel training
- Use best practices for large scale training and convergence

Optimize distributed performance

- Use best optimized libraries for communication
- Tune communication settings

PROFILING CODE

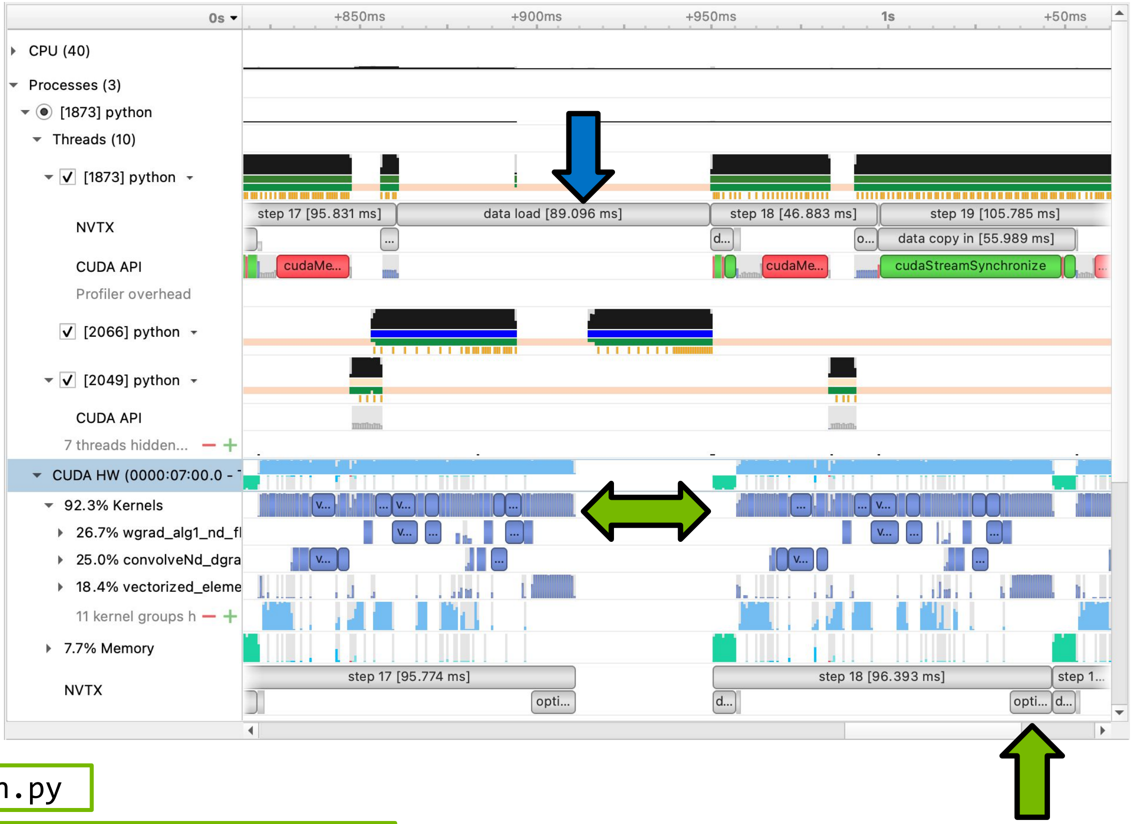
Using NVIDIA Nsight Systems

Using a profiler is an essential step in optimizing any code

Nsight Systems timeline provides a high-level view of your workload and helps you identify bottlenecks:

- I/O, data input pipeline
- Compute
- Scheduling (e.g. unexpected synchronization)

Can use NVTX ranges to annotate profiles



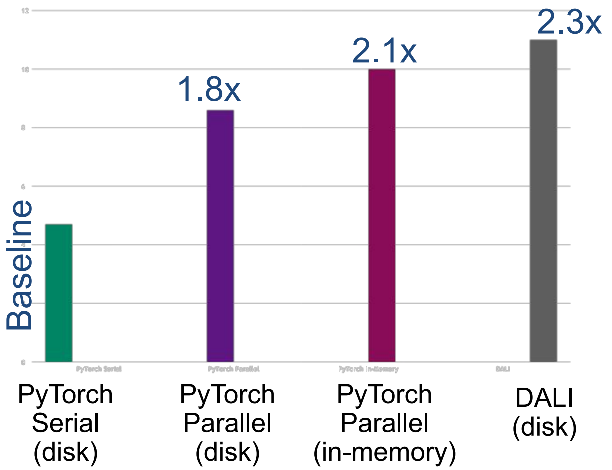
To generate a profile:

```
nsys profile -o myprofile python train.py
```

```
nsys profile -o myprofile -t cuda,nvtx python train.py
```

credit: Josh Romero, Thorsten Kurth (NVIDIA)

Optimizing GPU performance



Data loading

- Frequent cause of performance loss for users
- Parallelize your I/O
- Consider NVIDIA DALI

Mixed precision (FP32 + FP16)

- Can speed up training, leverage tensor cores, reduce memory
- Frameworks provide capabilities for automatically using FP16 where appropriate and for scaling gradients to prevent numerical underflow

JIT compilation, APEX fused operators, CUDA Graphs

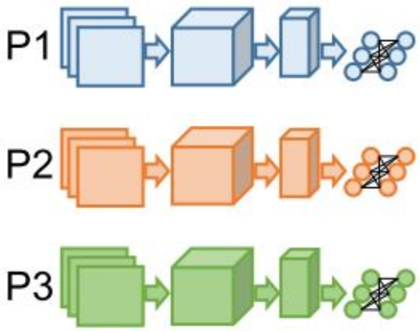
- Fuses kernels (+launches) together to increase GPU utilization

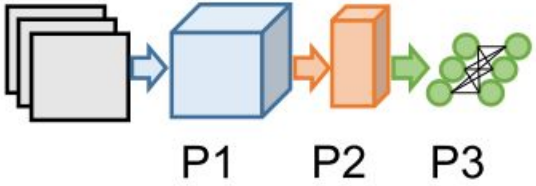
Other tricks

- Check out our tutorial for more

Full set of optimizations in tutorial => 6x speedup!

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

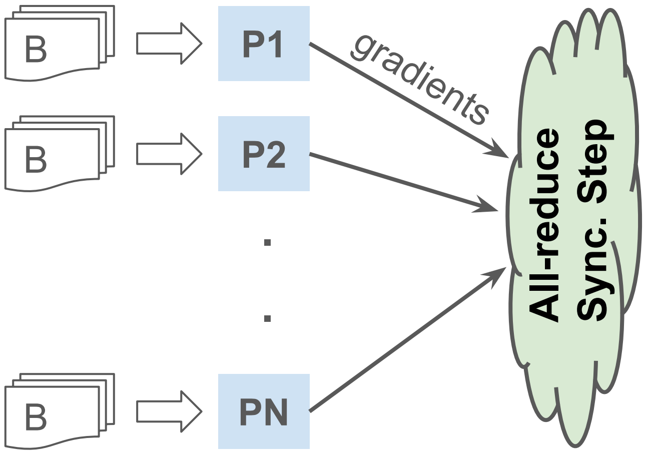
Synchronous data parallel scaling

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

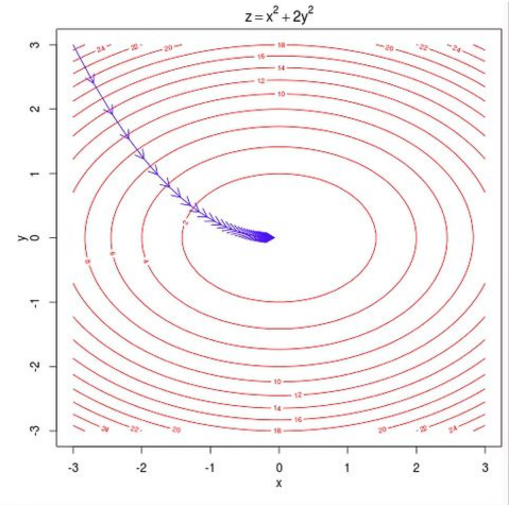
How do we accelerate learning?

Recall batched stochastic gradient descent:

$$w_{t+1} \leftarrow w_t - \frac{\eta}{B} \sum_{i=1}^{B} \nabla L(x_i, w_t)$$

B is batch-size

η is learning rate



We can converge faster by taking fewer, bigger, faster steps

- i.e., larger batch sizes, larger learning rates, more processors
- *Not a free lunch!*

Learning rate scaling

3 SGD steps vs. 1 SGD step with 3x learning rate

Some rules of thumb may work for you

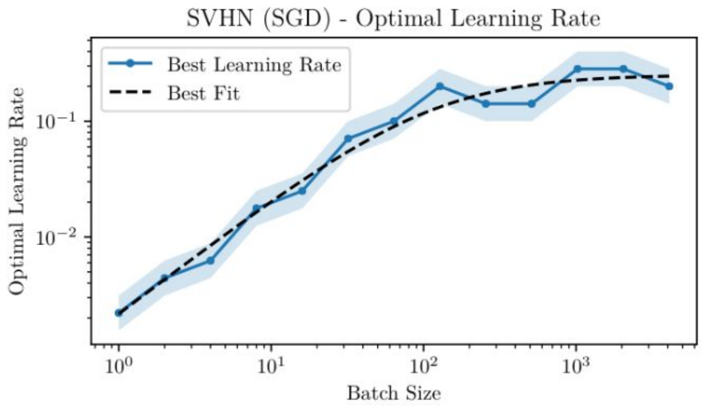
- Linear learning rate scaling:
  η → N * η
- Square-root learning rate scaling:
  η → sqrt(N) * η

Optimal learning rate can be more complex

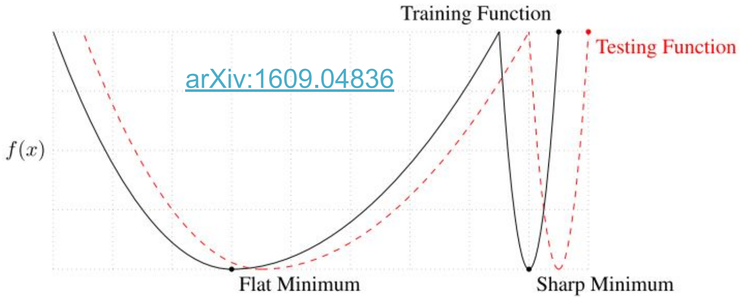
- See OpenAI (arXiv:1812.06162) study of dependence on batchsize

Large learning rates unstable in early training

- You may need a gradual LR “warm up”



Limits of large batch training



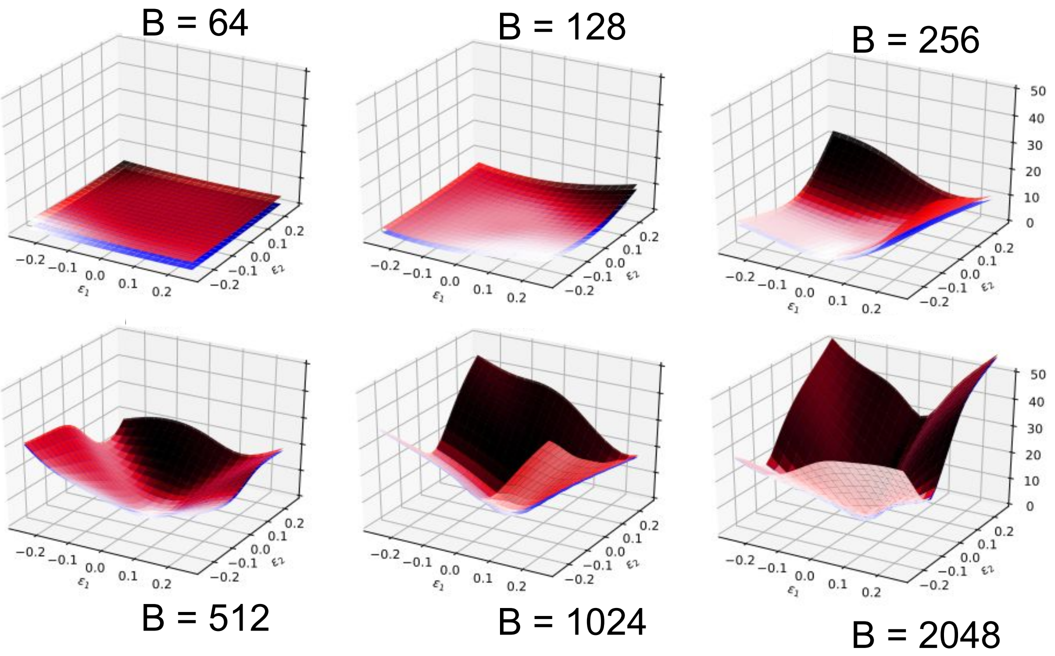
arXiv:1609.04836

Larger batches can result in sharper minima

- Poor generalization, overfitting

Loss at the end of training CIFAR-10 (axes are dominant eigenvectors of the Hessian)

Z. Yao et al. arXiv:1802.08241



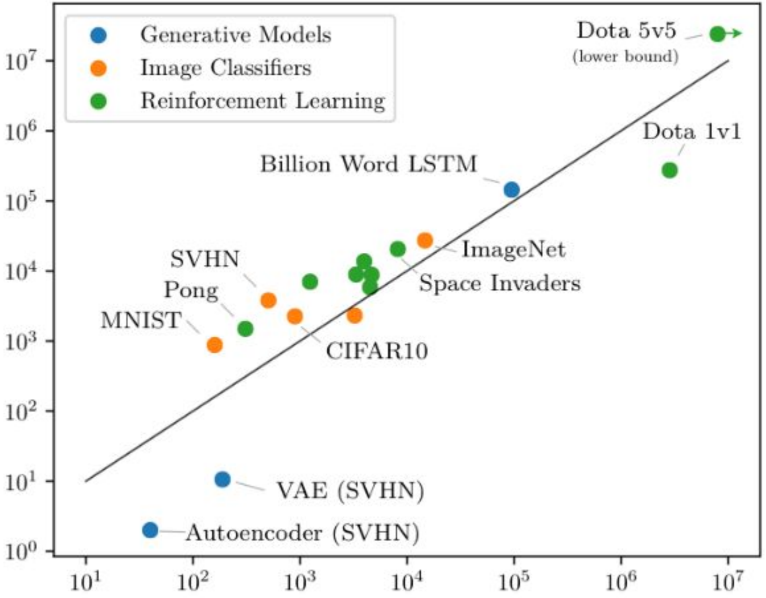
Limits of large batch training

Empirical studies by OpenAI ([arXiv:1812.06162](https://arxiv.org/abs/1812.06162)) and Google Brain ([arXiv:1811.03600](https://arxiv.org/abs/1811.03600)) show

- Relationship between critical batch size and *gradient noise scale*
- More complex datasets/tasks have higher gradient noise, thus can benefit from training with larger batch-sizes

McCandlish, Kaplan and Amodei [arXiv:1812.06162](https://arxiv.org/abs/1812.06162)

Gradient noise scale measures the variation of the gradients between different training examples



Critical batch size is the maximum batch size above which scaling efficiency decreases significantly

Some other tricks

Adaptive batch size

- Don't Decay the Learning Rate, Increase the Batch Size: https://arxiv.org/abs/1711.00489
- Adaptive batch-size scaling with 2nd-order information (ABSA): https://arxiv.org/abs/1810.01021

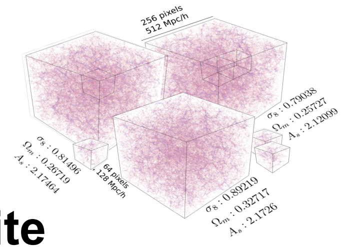
Large-batch optimizers

- LARS: https://arxiv.org/abs/1708.03888
  - layer-wise adaptive rate scaling
  - uses “trust ratio” to regularize update size to param size
- LAMB: https://arxiv.org/abs/1904.00962
  - some extensions to LARS, e.g. based on Adam
  - gave SOTA performance on language models

MLPerf™ Performance Benchmarks

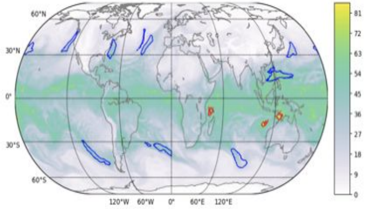
MLCommons™ publishes the MLPerf benchmarks which are driving performance innovation in ML training and inference workloads

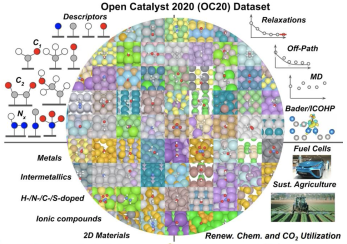
- Latest MLPerf Training results scale to 4k TPUs and GPUs; ResNet50 now trains in ~12s.



NERSC played active role to develop MLPerf HPC benchmark suite

- Scientific applications that push on HPC systems:
  - CosmoFlow - 3D CNN predicting cosmological parameters
  - DeepCAM - segmentation of phenomena in climate sims
  - OpenCatalyst - GNN modeling atomic catalyst systems
- MLPerf HPC v1.0 release at SC21 conference:
  - Time-to-train and “Weak-scaling” throughput metrics
  - Competitive results with Perlmutter, very useful experience for NERSC





Megatron-Turing NLG 530B

Currently the world's largest and most powerful generative language model

- 530 billion parameters
- SOTA performance in several NLP tasks

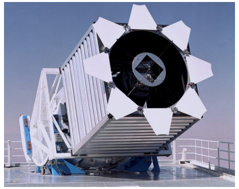
Uses multiple forms of parallelism

- 8-way tensor-parallelism within a node
- 35-way pipeline parallelism across nodes
- data-parallelism up to thousands of GPUs

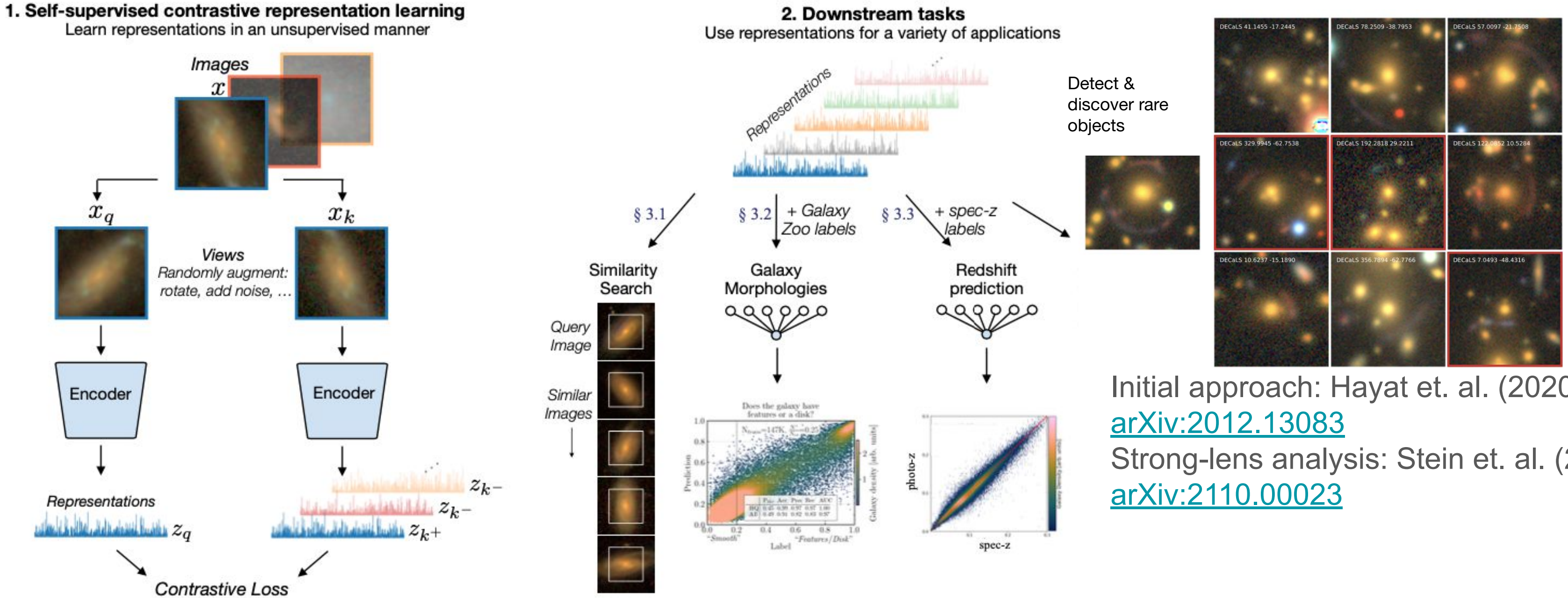
Press releases:

- [NVIDIA Technical Blog]
- [Microsoft Research]

Self-supervised sky surveys



- Sky surveys image billions of galaxies that need to be understood
- Limited “labels”, so can learn in *semi-supervised* way
- Pre-training on entire dataset on HPC, downstream task can be on laptop/edge
- Recently used to find > 1000 previously undiscovered strong-lens candidates

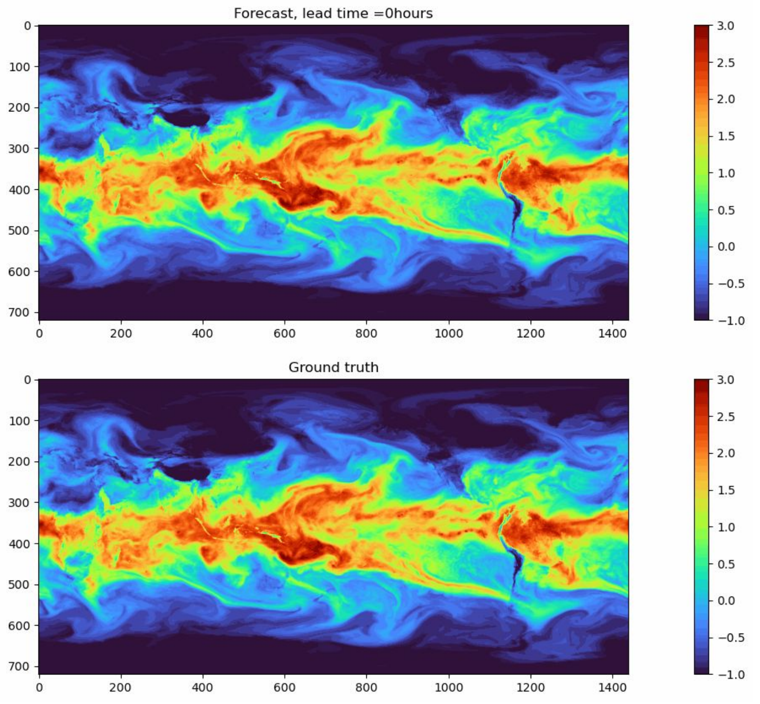


Initial approach: Hayat et. al. (2020) arXiv:2012.13083
Strong-lens analysis: Stein et. al. (2021) arXiv:2110.00023

FourCastNet: Data-driven atmospheric modeling

Pathak et al. 2022
arXiv:2202.11214

- Data-driven modeling of atmospheric flows using a state-of-the-art transformer-based FourCastNet
- Collaboration with NVIDIA, Caltech and others
- Forecasts global weather at 0.25° resolution
  - **Order of magnitude greater resolution** than state-of-the-art deep learning models
  - Forecasts wind speeds, precipitation and water vapor close to the skill of numerical weather prediction models up to 8 days
  - Produces a 24hr 100-member ensemble forecast in 7 seconds on a Perlmutter GPU node
  - Traditional NWP: 5 mins on *thousands of CPU nodes* for equivalent ensemble



Data-driven forecast of an atmospheric river


Jaideep Pathak
former NERSC
Postdoc now NVIDIA


Shashank
Subramanian
NERSC Postdoc


Peter Harrington
NERSC ML
Engineer

Conclusions

AI for science requires supercomputer-scale capabilities

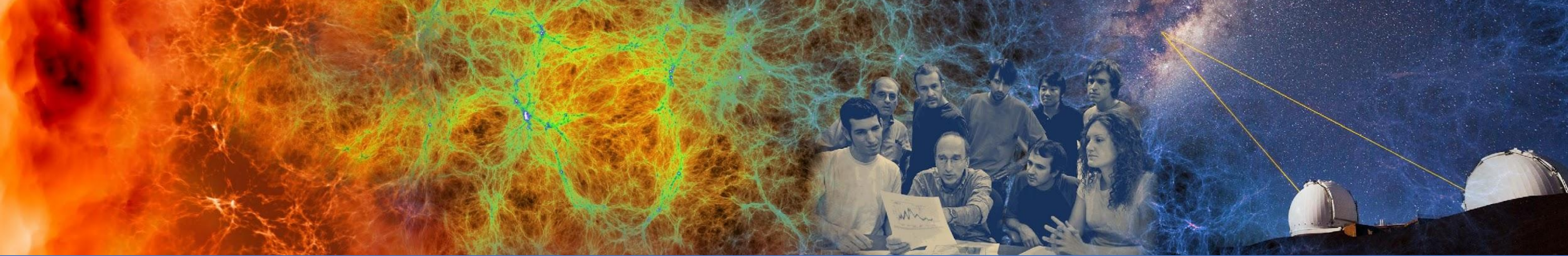
- Optimized, scalable hardware and software
- Flexibility, interactivity, and automation
- NERSC delivering this with Perlmutter

Scientific AI growing in sophistication and maturity

- Trend towards science-specific architectures and scale
- Examples running now on Perlmutter - much more to come
- We're excited to see what comes next

Questions? Collaborations? => sfarrell@lbl.gov

We're hiring postdocs, engineers, and staff:
https://lbl.referrals.selectminds.com/page/nersc-careers-85



Thank you!