

# Throughput Machine Learning in CHTC

Ian Ross

Data Engineer, Center for High Throughput Computing

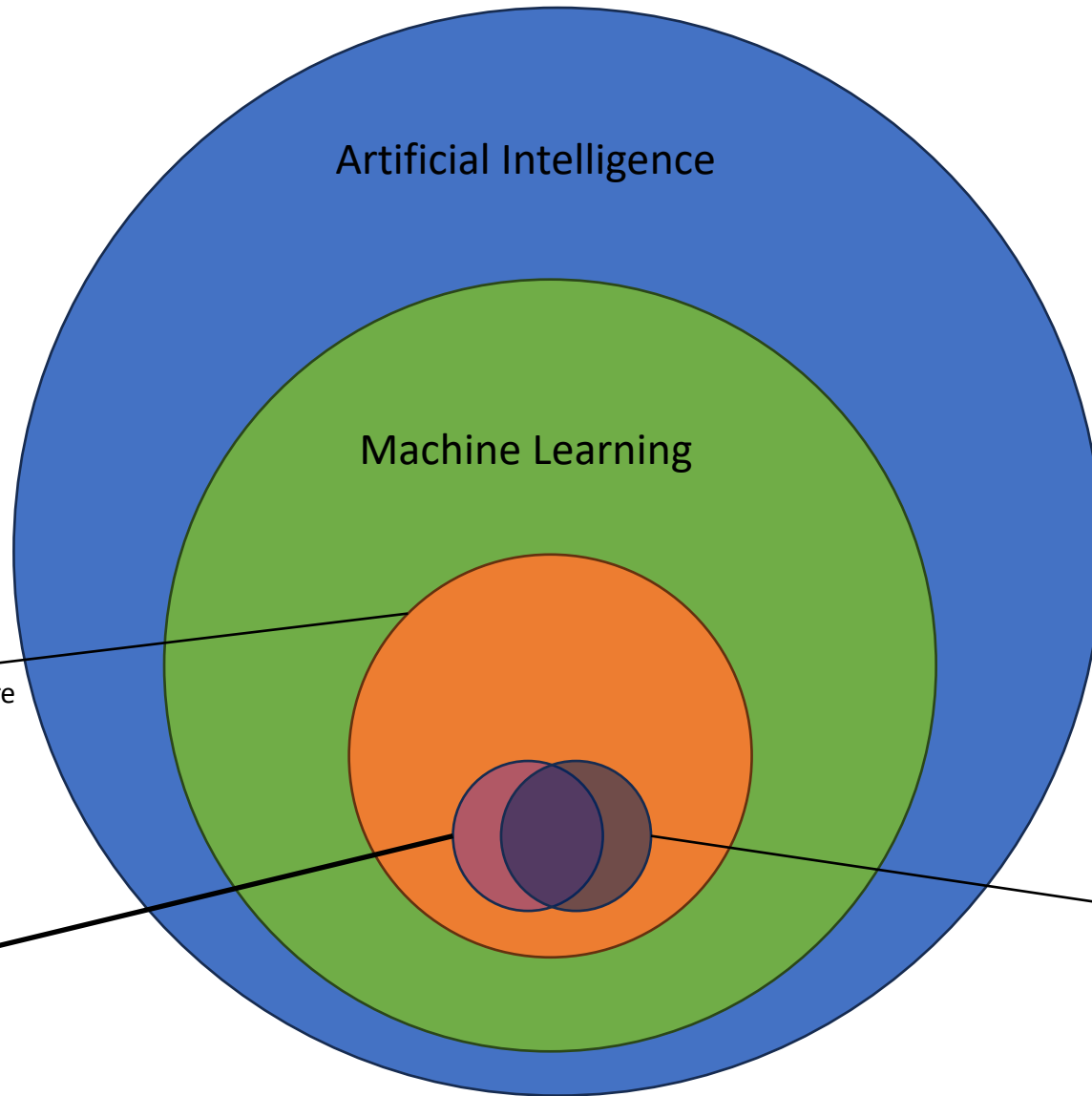
# Outline

- Artificial Intelligence and Machine Learning – a too-brief overview
- Throughput Machine Learning
- Example use cases
- ML workflows and usage in CHTC
- Ongoing work
- Future plans

# AI/ML – a too-brief overview

- Artificial intelligence – Methods and software to enable machines to *observe, identify, and react to stimuli to achieve a defined goal*
- Machine learning – Algorithms and practices to enable machines to recognize patterns in data and generalize to new data to achieve tasks *without instruction*
  - Subset of AI





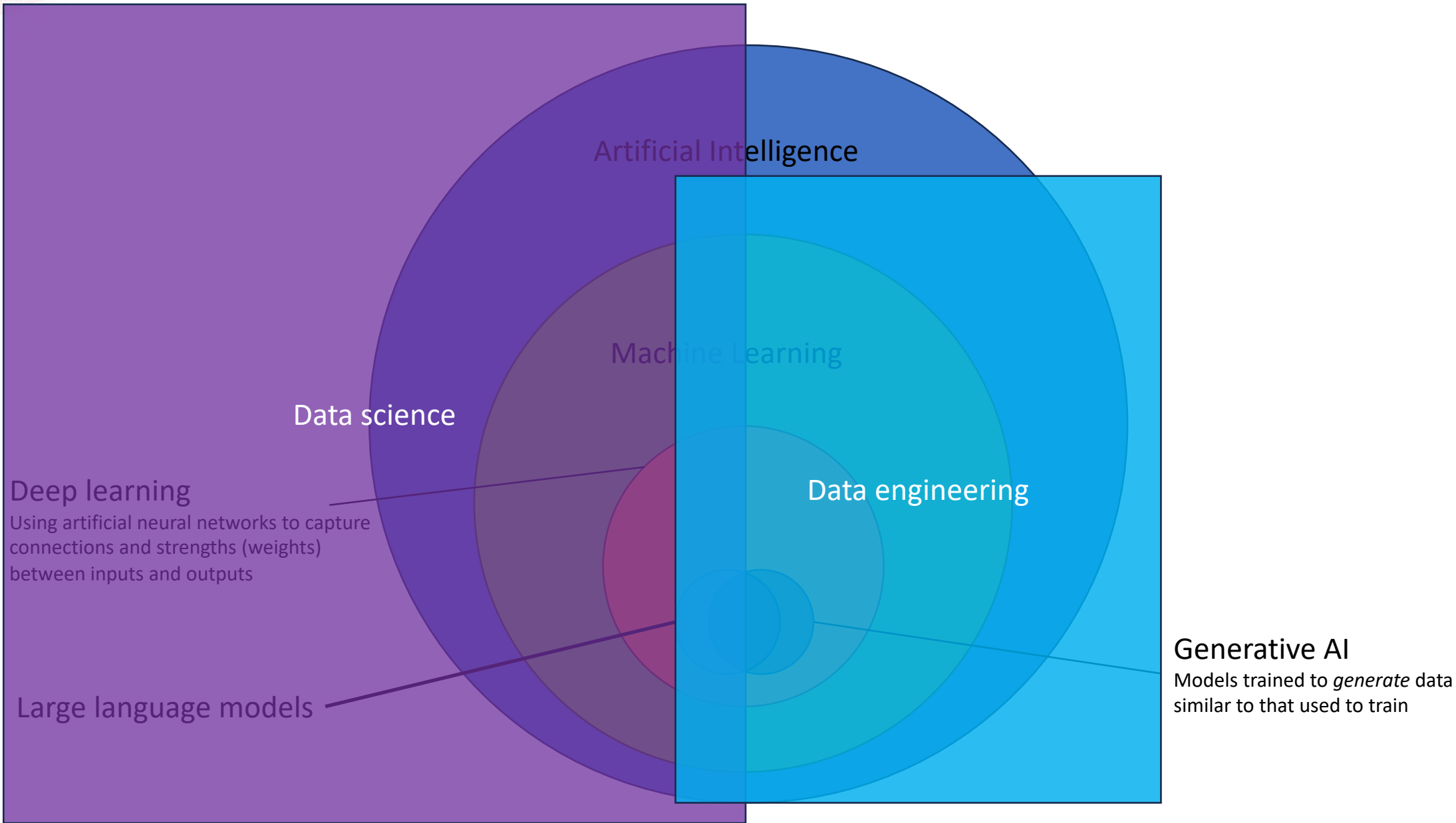
### Deep learning

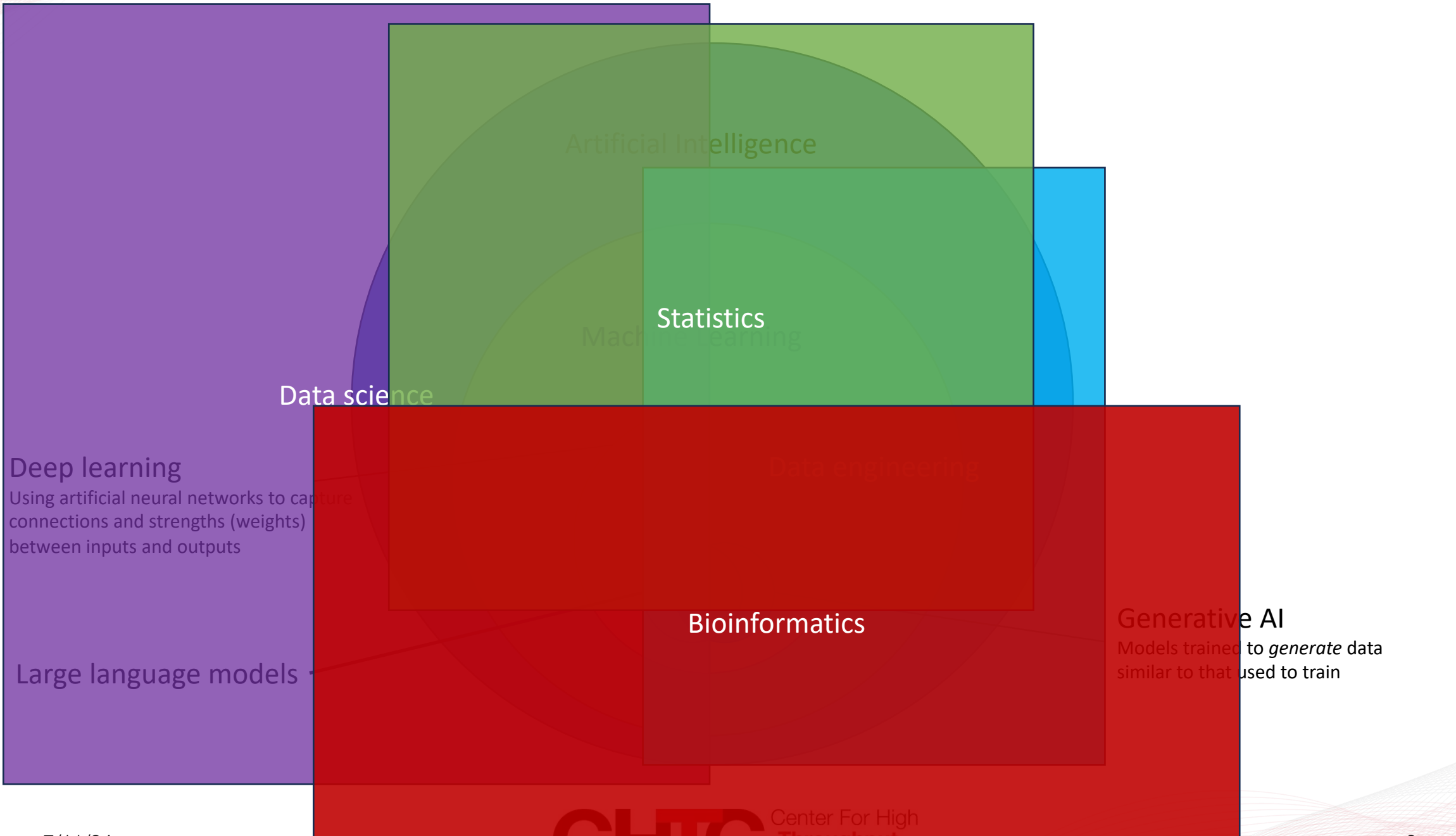
Using artificial neural networks to capture connections and strengths (weights) between inputs and outputs

### Large language models

### Generative AI

Models trained to *generate* data similar to that used to train





**Deep learning**  
 Using artificial neural networks to capture connections and strengths (weights) between inputs and outputs

Large language models

Data science

Artificial Intelligence

Machine Learning

Statistics

Bioinformatics

**Generative AI**  
 Models trained to generate data similar to that used to train

We care about these things primarily  
as tools and techniques that  
enable new and novel SCIENCE

...but there are complications

Deep learning

Using artificial neural networks to calibrate connections and strengths (weights) between inputs and outputs

Large language models

Generative AI

Used to generate data used to train

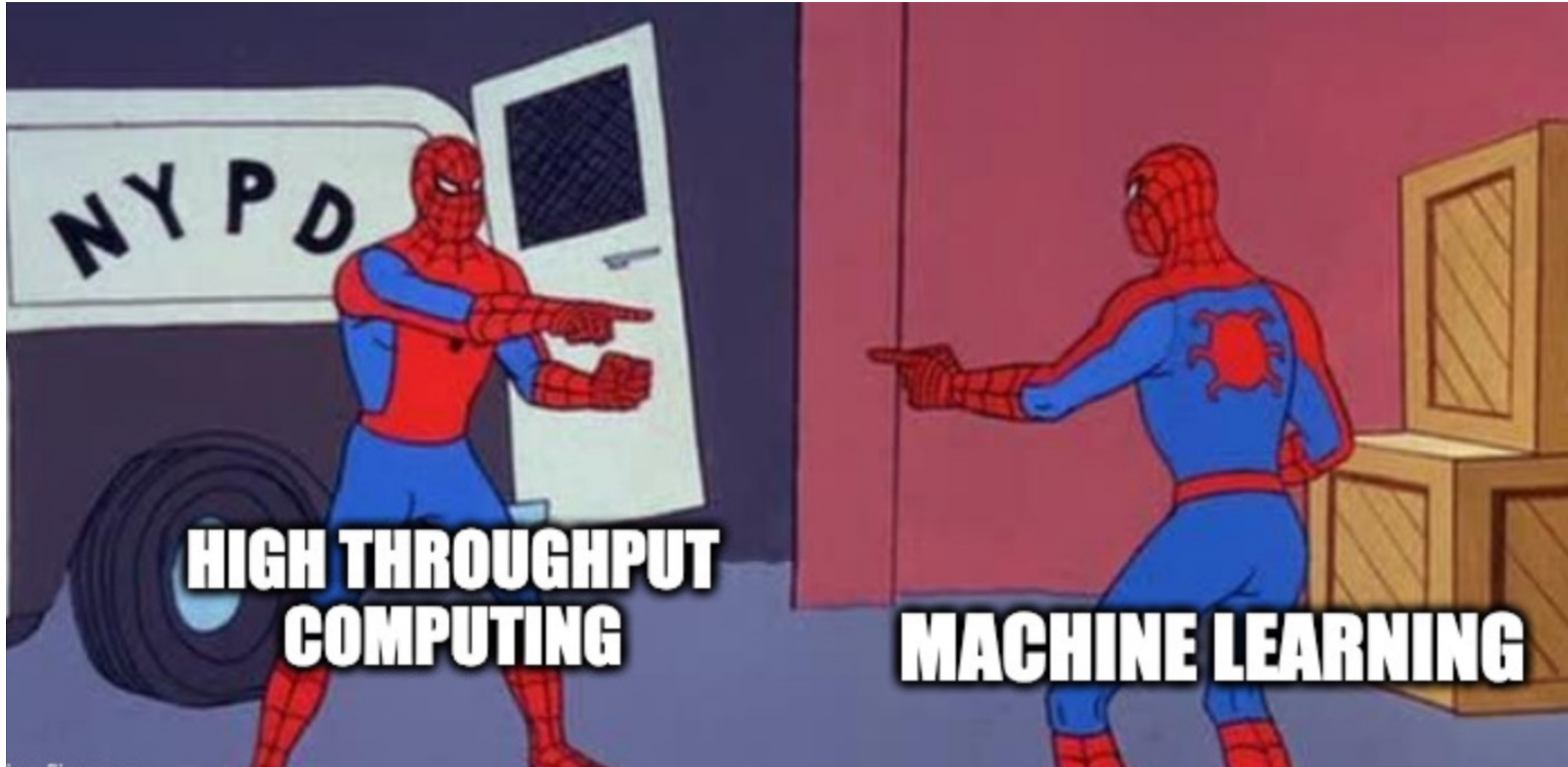
# AI and ML – inherent challenges

- Data and computing needs can be immense
- GPUs bring their own layer of complexity
  - Dropouts, user education, administration, cost, availability
- Ecosystem moves *fast* and everybody wants to be first
  - Copy-paste recipes propagate faster than truly educational ones
  - Software stacks with similar, but not quite the same, interface (recipe challenges – more to come)
- Training can be a long process
  - Checkpointing required



# AI and ML – inherent challenges

- Data
  - Economics
    - Dropouts, user education, administration, cost, availability
  - GPU
    - Training can be a long process
      - Checkpointing required
- These are not new challenges! User education, scheduling, data movement, workflow orchestration... “There is nothing new under the sun”



# Throughput machine learning in CHTC

- Throughput machine learning applies our tried and tested technologies to ML applications
  - Data movement... but at a much bigger scale (and potentially across nodes)
  - Checkpointing... but at an epoch boundary (and potentially across nodes)
  - DAGMan... but with a few extra bells and whistles (and potentially across nodes)
- Resources (GPUs) are a bit more “valuable” and policy is king
  - Prioritization, scheduling, pre-emption
- Three use cases to highlight familiarity and new twists..

# Throughput machine learning – Use case 1

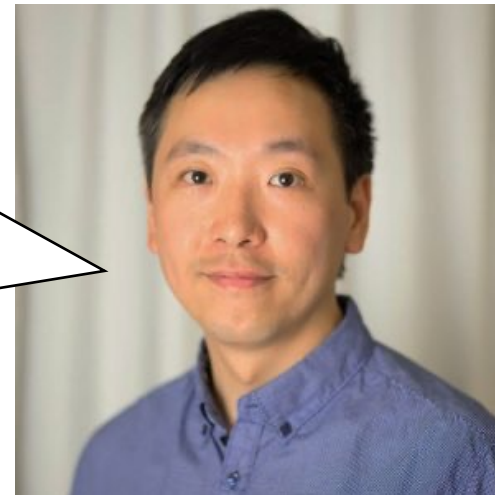
I have 18 million scientific articles, and I want to search for, extract, and synthesize information across them!  
This is a high-throughput inference problem!



I have 18 million scientific articles, and I want to search for, extract, and synthesize information across them! This is a high-throughput inference problem!



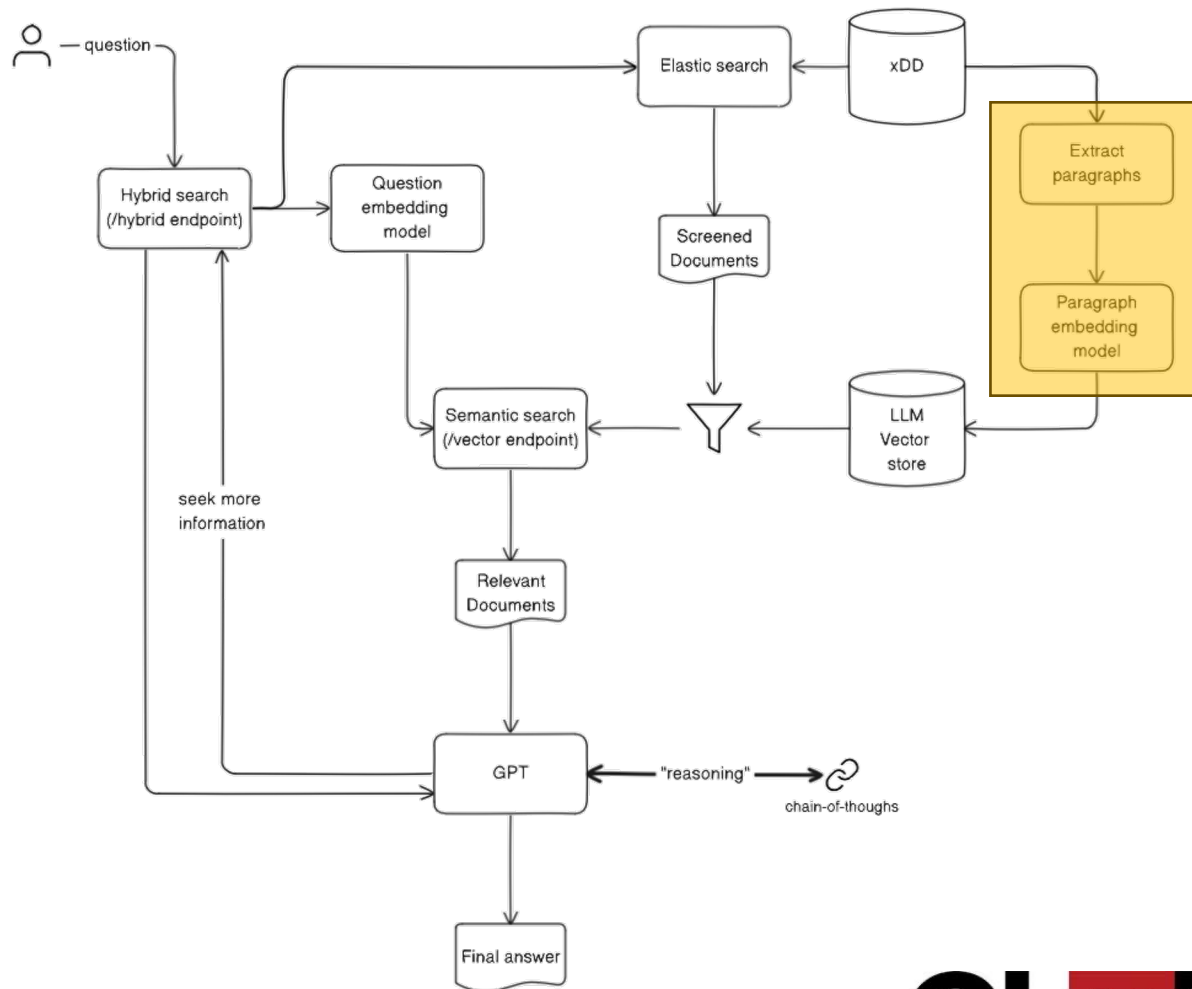
I can help with the LLM+RAG side of things!



Jason Lo, DSI collaborator

# Throughput inference

Illustration of v0.3.0 (/react endpoint)



- Using existing models for inference in traditional high-throughput approaches (with or without GPUs)
- Examples:
  - Embedding
    - Vector or scalar representations of input data
  - Predictions
  - LLM security studies

## Topic

Choose a topic

- COVID-19
- Dolomites
- Climate Change
- CriticalMAAS
- Geoarchive

## Ask preset questions

Ask preset questions

Given CFCs and SF6 are not natu... ▾

Run

## Advanced settings

You can customize the QA system, all of these settings are available in the [API route](#) as well.

model

- gpt-3.5-turbo-16k
- gpt-4
- gpt-4-1106-preview

retriever top-k

5 - +

elastic search screening phase top-k

100 - +

verbose



Given CFCs and SF6 are not naturally occurring, what ocean body (and what part) should contain the highest concentration?



Thought: To answer this question, I need to consider the sources and behavior of CFCs (chlorofluorocarbons) and SF6 (sulfur hexafluoride) in the environment. Since these substances are not naturally occurring, their presence in the ocean is likely due to human activities. The highest concentrations would probably be found in parts of the ocean that are closest to major industrial areas where these compounds are emitted, and where there is less water circulation to dilute them. However, I need more specific information to determine which ocean body and what part would contain the highest concentration.



Action: \_search\_retriever



Action Input: Which ocean body and part has the highest concentration of CFCs and SF6?



Improved estimates of ventilation rate changes and CO<sub>2</sub> uptake in the Pacific Ocean using chlorofluorocarbons and sulfur hexafluoride [Sonnerup, R. E., Bullister, J. L., & Warner, M. J. (2008)]

As the useful dating horizons of all of these tracers propagate deeper into the water column during the coming decades, CFC levels of waters ventilated during the “linear window” of the atmospheric CFC history (1970 – 1988) will provide simpler, more accurate representations of ventilation timescales (and associated anthropogenic CO<sub>2</sub> uptake and oxygen utilization rates) into deeper waters than during the 1990s, and the continuing penetration of the rapidly increasing, and now readily measurable [Bullister and Wisegarver, 2008; Tanhua et al., 2008], SF<sub>6</sub> tracer should provide improved water mass dating tools in upper ocean waters, especially when exploited in a multitracer modeling approach to quantify mixing’s impacts on the tracer ages.

Source: <https://xdd.wisc.edu/api/v2/articles/?docid=56a2f8e0cf58f19442529c99>



A global estimate of the full oceanic 13 C Suess effect since the preindustrial [Eide, M., Olsen, A., Ninnemann, U. S., & Eldevik, T. (2017)]

Ask a question.



# Throughput machine learning – Use case 2

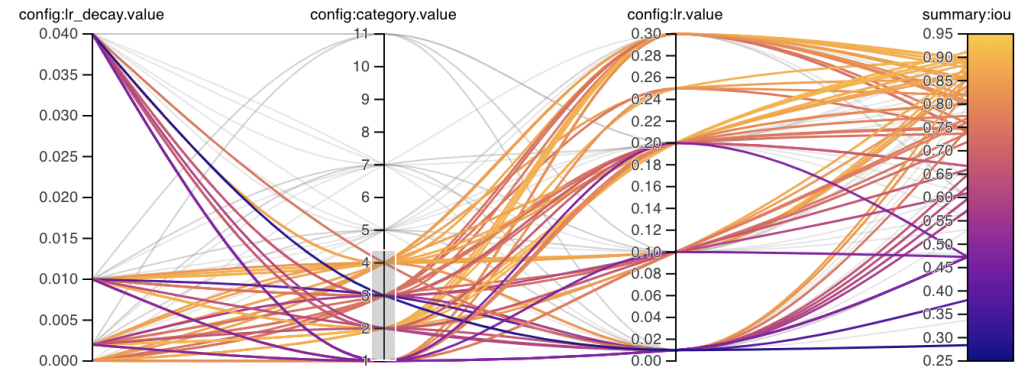
I want to train many models, empirically measure their predictive power, and use those models to drive scientific exploration.



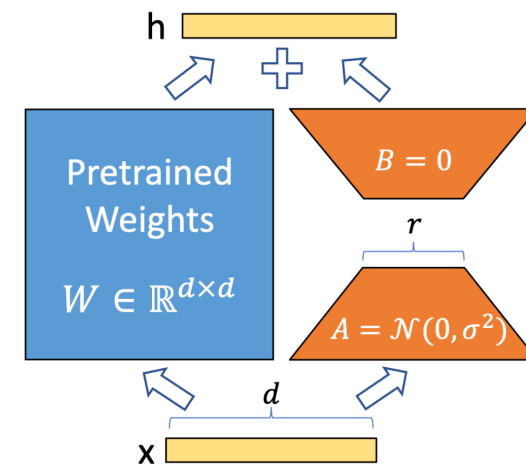


# Throughput Training

- Training (potentially many) relatively small models to drive additional inference or science work
- Examples:
  - Hyperparameter optimization
  - Ensemble learning
  - Fine-tuning
  - Ablation studies



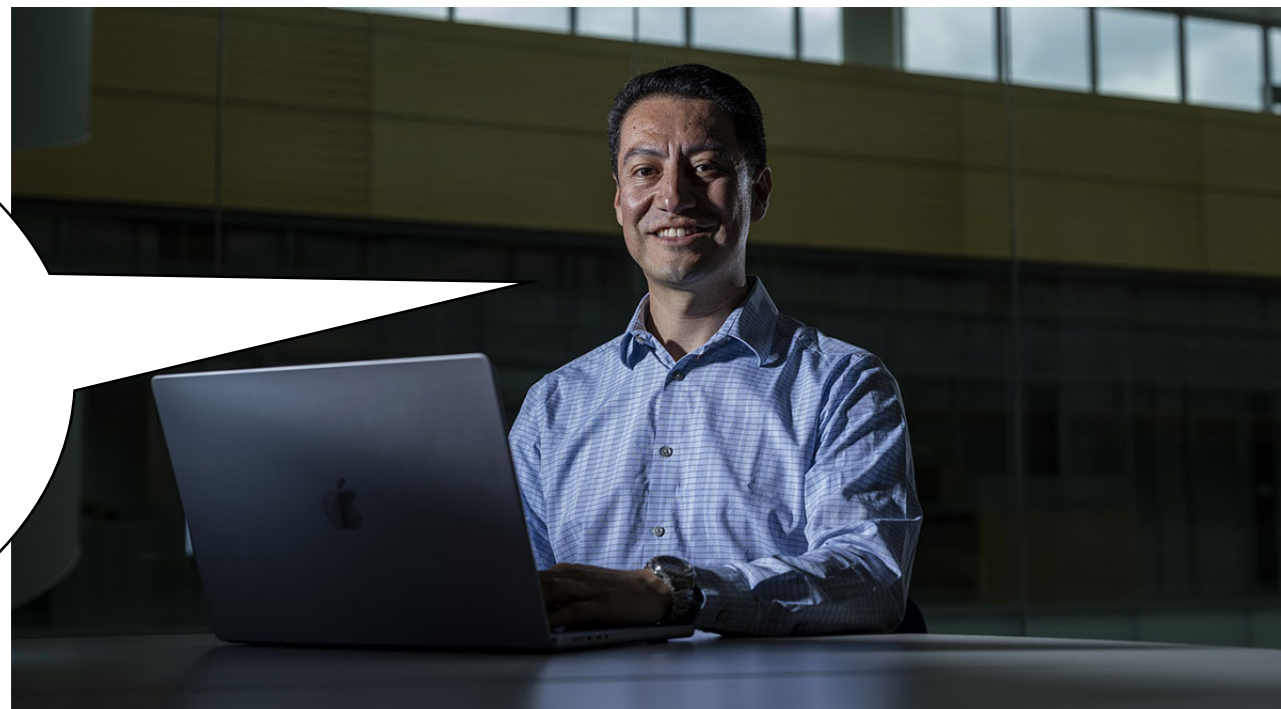
From <https://docs.ultralytics.com/guides/hyperparameter-tuning/>



LoRA: Low-Rank Adaptation of Large Language Models (<https://arxiv.org/pdf/2106.09685>)

# Throughput machine learning – Use case 3

I want to create a foundation model for bioimaging and want to scale training across multiple nodes!



# Multi-node training in CHTC

- Aka what can we do with existing tools?
- Challenges:
  - Heterogeneous resources (solved with requirements)
  - Asynchronous resource acquisition (pytorch elastic)
  - Nodes involved need a rendezvous server address (solved with DAG)
  - Performance
    - TBD... but won't compare to the big dogs in the LLM fight.

# Multi-node training in CHTC (POC)

## train.dag

```
PROVISIONER RANK0 gpu.sub
JOB WORKER gpuworker.sub
SCRIPT PRE WORKER injectRank0Name.sh
```

## gpu.sub

```
...
executable = train.sh
arguments = rank0 9640
transfer_input_files = train.py, train.sh,
/usr/lib64/python3.9/site-
packages/htcondor/htchirp/htchirp.py
...
```

## train.sh

```
...
if [ "$1" = "rank0" ]
then
    # rank0 port
    h=$(hostname)
    p=$2
    echo "$h $p" > contact_file
    python3 htchirp.py put contact_file rank0_contact
    rm contact_file
else
    h=$1
    p=$2
fi
torchrn --nnodes 1:2 --nproc_per_node 1 --rdzv_backend
c10d --rdzv-id 1 --rdzv-endpoint "$h:$p"
...
```

## gpuworker\_sub.template

```
...
executable = train.sh
arguments = SERVER_IP_ADDRESS_GOES_HERE
SERVER_PORT_GOES_HERE
requirements = (GPUS_MaxSupportedVersion > 12000) &&
(GPUs_GlobalMemoryMb >= 32000) && machine !=
"SERVER_IP_ADDRESS_GOES_HERE"
...
```

## injectRank0Name.sh

```
#!/bin/sh

# A prescript for the workers to inject
# the address and port of the rdzv server
# into their submit file.

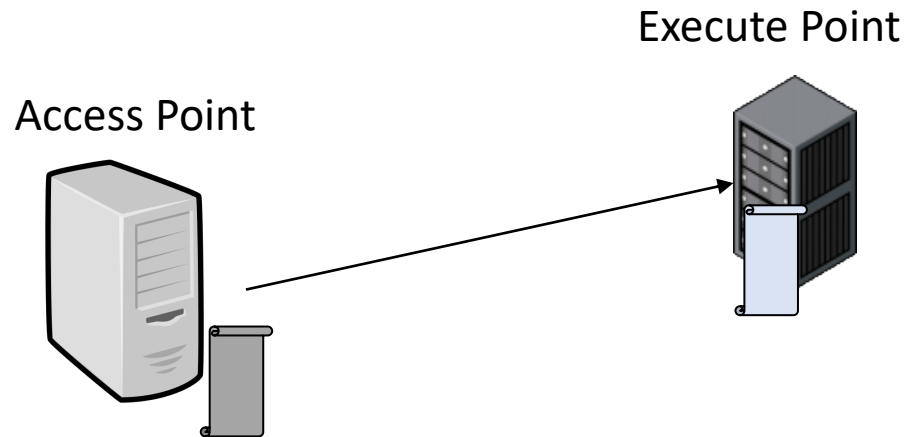
# We assume this is in a file in the
# cwd named "rank0_contact"

read name port < rank0_contact

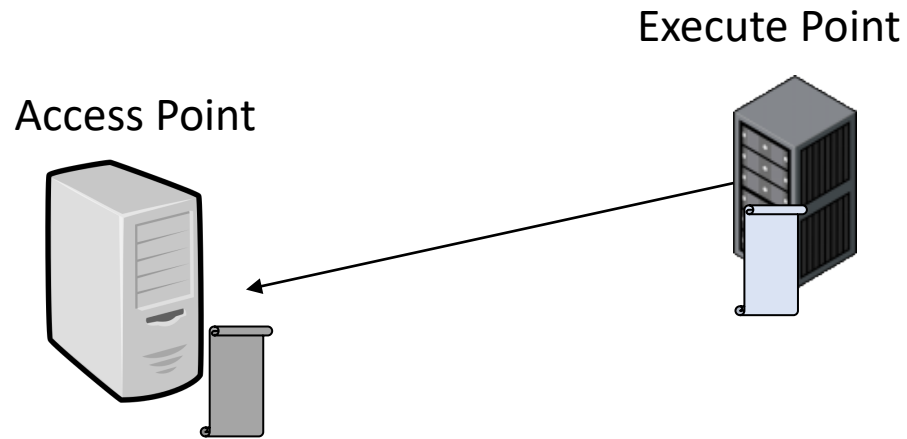
sed -e "s/SERVER_IP_ADDRESS_GOES_HERE/${name}/g" \
    -e "s/SERVER_PORT_GOES_HERE/${port}/g" < gpuworker.sub.template \
    > gpuworker.sub
```

# Multi-node training in CHTC (POC)

1. Provisioner (rank0) node starts



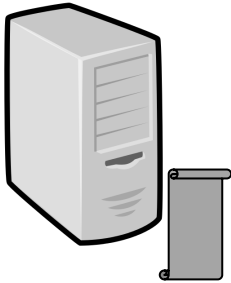
# Multi-node training in CHTC (POC)



1. Provisioner (rank0) node starts
2. Provisioner node chirps back address and port

# Multi-node training in CHTC (POC)

Access Point



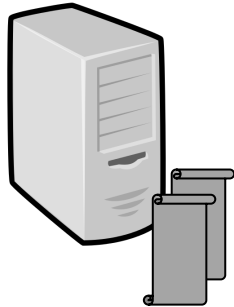
Execute Point



1. Provisioner (rank0) node starts
2. Provisioner node chirps back address and port
  - 2a. rank0 torchrun starts (🎲)

# Multi-node training in CHTC (POC)

Access Point



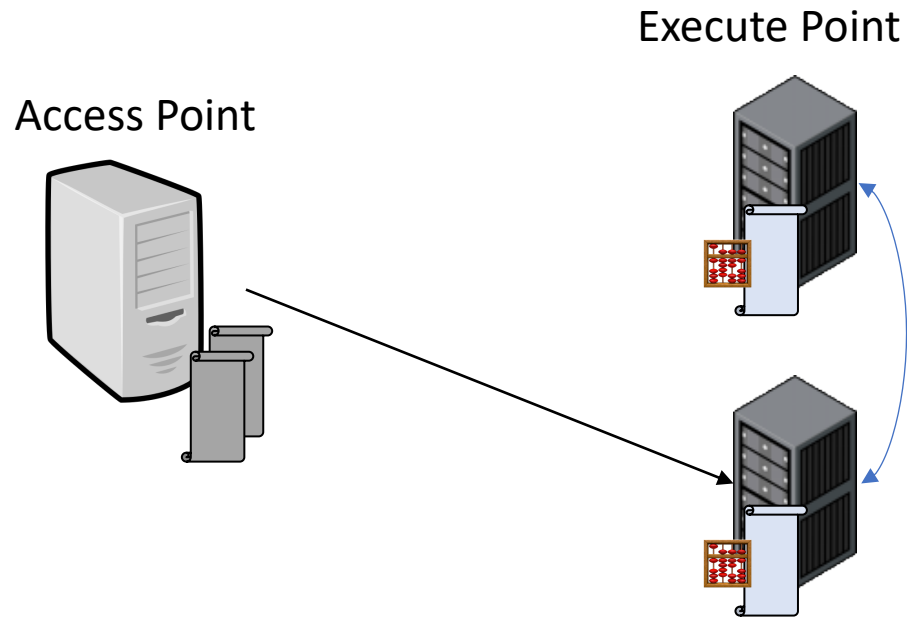
Execute Point



1. Provisioner (rank0) node starts
2. Provisioner node chirps back address and port
  - 2a. rank0 torchrun starts (🎲)
3. Pre-script for worker node runs, injecting rank0 job address into worker submit template



# Multi-node training in CHTC (POC)




1. Provisioner (rank0) node starts
2. Provisioner node chirps back address and port
  - 2a. rank0 torchrun starts (🎲)
3. Pre-script for worker node runs, injecting rank0 job address into worker submit template
4. Worker node runs and torchrun starts (🎲) with rank0 node defined as rendezvous server

# ML workflows and usage in CHTC

- A (very) unscientific survey
- Training vs inference:
  - 60/40 split training to inference
- Software stack:
  - Pytorch (~60%), parabricks, Lightning
- Domains: Bioinformatics, LLMs, empirical ML (bias detection), ...
- Notable feedback:
  - “For what it’s worth, more physical gpus of moderate size (16GB memory) is much more preferable for my purposes than an H100 with 80GB memory.”
  - “We are super thankful for the the CHTC providing GPU support, as this research would not have been possible without it!”

# Ongoing work

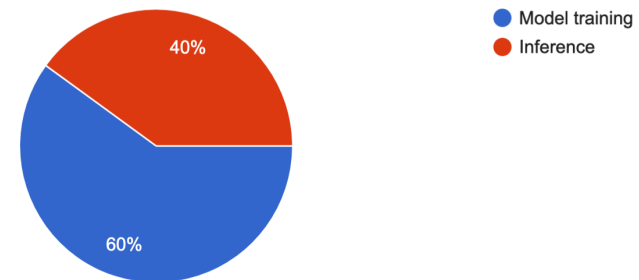
- Pelican/pytorch integration
- Tutorials, documentation, recipes
  - <https://github.com/CHTC/templates-GPUs>
  - <https://chtc.cs.wisc.edu/uw-research-computing/machine-learning-htc>
- Additional DAG workflows
  - Evaluation + early abort
- More surveys and resource understanding



**Kristina Zhao**

Mentor(s):  
Emma Turetsky and Ian Ross

**Integrating PyTorch and Pelican**



# Future plans

- Answers to:
  - How does the heterogeneity of resources and availability impact a checkpointed + restarted training process?
  - How do we effectively move datasets for training (and inference!)?
  - What, if any, gaps exist in enabling users to handle large ensemble trainings?
- ...and where-ever AI developments and this community take us.