# Running Machine Learning Pipelines on RCC's HPC Systems

#### Materials:

github.com/rcc-uchicago/ml-pipelines-workshop

J.D. Laurence-Chasen, Ph.D.
Computational Scientist
Research Computing Center

## The plan

- 1. Overview of ML pipelines
- 2. RCC resources
  - Hardware, Software
- 3. Submitting ML jobs to Midway
  - Checking GPU engagement in your scripts
  - sbatch setup

#### Learning objective

Be able to submit batch ML jobs (Tensorflow/PyTorch) to Midway GPU nodes for accelerated training.



### A typical ML pipeline

Prepare data

- Cleaning, processing, etc.
- Lots of interactivity
- Usually underestimated

Design model

- Tensorflow? PyTorch?
- CNN? RNN? LSTM? Etc...

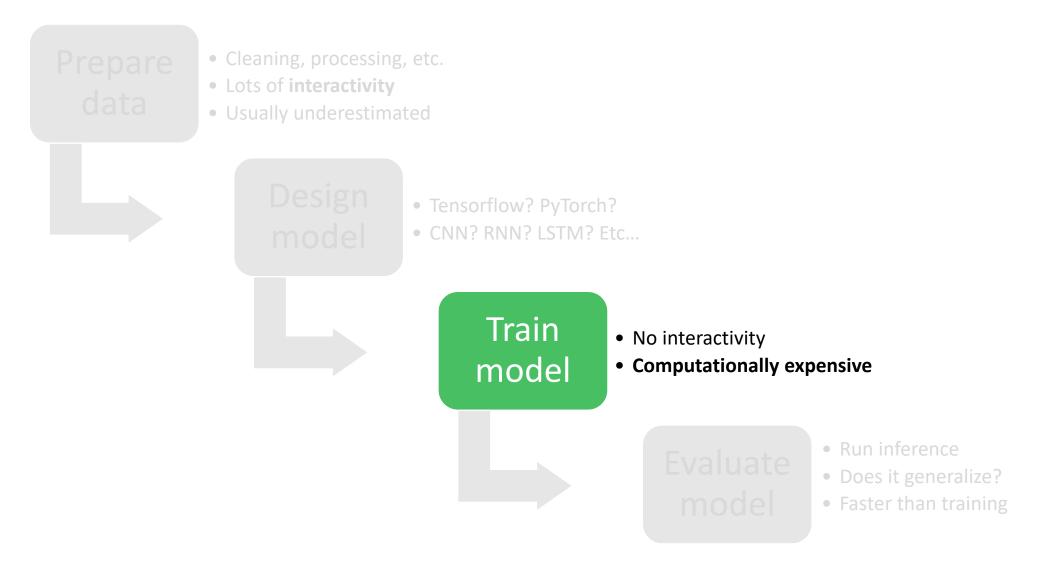
Train model

- No interactivity
- Computationally expensive

Evaluate model

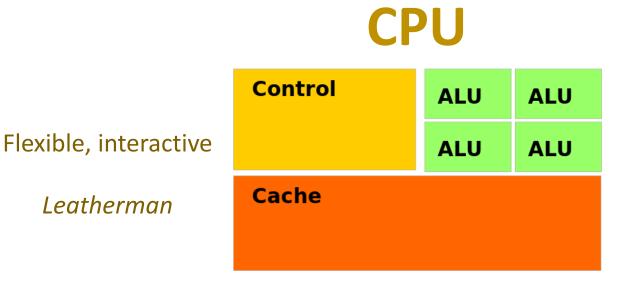
- Run inference
- Does it generalize?
- Faster than training

#### ML model training is time-consuming but *not* interactive



GPUs accelerate ML model training...why?

#### Different core counts for different functions



**Tens** of cores

GPU

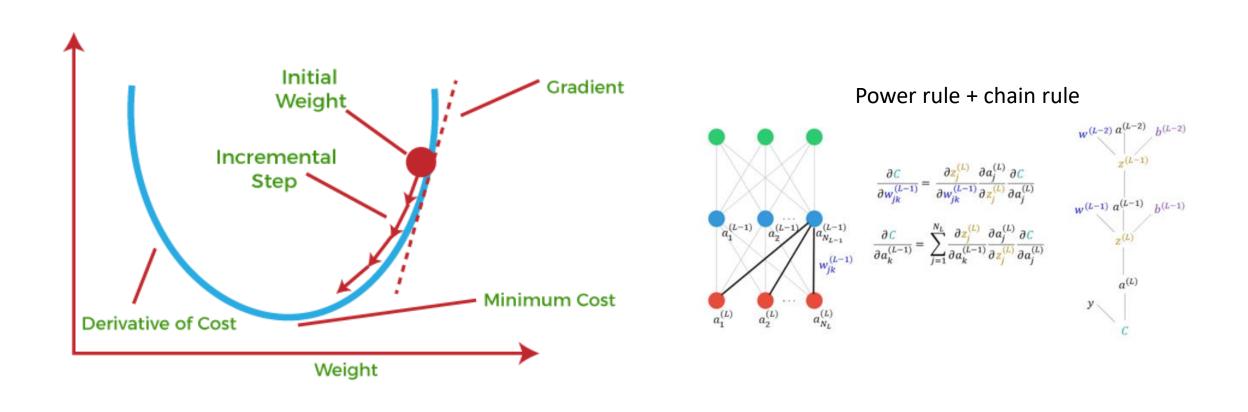
Specialized for many repeated calculations

Sharp fillet knife

**Hundreds** to **thousands** of cores

ALU = arithmetic logic unit

#### ML model training is a lot of repeated calculations



See Andrej Karpathy's YouTube Channel

## Communal Midway GPU nodes

System	Midway2		Midway3	
GPU Type	NVIDIA Telsa K80	NVIDIA Tesla V100	NVIDIA Quadro RTX 6000	NVIDIA Tesla A100
G3D Benchmark	7,025	16,235	19,554	n/a
# of Nodes	6	5	5	1
# of GPUs per node	4	4	4	4
# of cuda cores per GPU	4,992	5,120	4,608	6,912
Memory per GPU	24 GB	16 GB	24 GB	40 GB

Total # of GPUs	24	44
-----------------	----	----

#### Some useful software tools on Midway2 and Midway3

- Tensorflow
- PyTorch
- DeepLabCut
- MONAILabel
- Intel Al Toolkit

## Let's try it