Optimization in Machine Learning for Neutrino Classification

Nicole Naporano

August 7, 2020
Summer Undergraduate Research Symposium
UC Physics Department

What Are Neutrinos?—fundamental particles (leptons) with neutral charge

that have puzzlingly small masses and interact only with the Weak Force

- Three flavors (mass states)
- Paired with charged partner

  ν_e with electrons

  ν_μ with muons

  ν_τ with tau particles

Where do they come from?

- Collisions following Big Bang
- Proton-proton fusion in stars
- Cosmic rays
- Beta decay

What makes them so weird?

- Change flavors
  - Superposition of mass states $\nu_{1,2,3}$
  - Probability oscillation obeys Schrödinger Equation
- Leptons => don't interact with Strong Force
- Neutral charge => don't interact with EM Force
- Mysterious and sneaky
  - "Normal" hierarchy of masses assumed but not required

Oscillation frequency—function of mass, distance traveled, and energy: $\sin^2(\varphi)(\Delta m)^2 \frac{L}{E}$

- In detectors, $\cos(\varphi)$ = zenith angle from entry point
- **Mixing angles** give amplitude of periodic flavor oscillation
- **Mixing matrices**: rows = flavors, columns = mass eigenstates

Lepton Mixing (PMNS) Matrix—unitary matrix with mixing angles and complex phases

- Components $U_{\alpha i}$ with α = flavor, i = eigenstate

$$\begin{bmatrix} \nu_e \\ \nu_\mu \\ \nu_\tau \end{bmatrix} = \begin{bmatrix} U_{e1} & U_{e2} & U_{e3} \\ U_{\mu 1} & U_{\mu 2} & U_{\mu 3} \\ U_{\tau 1} & U_{\tau 2} & U_{\tau 3} \end{bmatrix} \begin{bmatrix} \nu_1 \\ \nu_2 \\ \nu_3 \end{bmatrix}$$

  - Parameterized by mixing angles:

    θ_{13} = probability (small) that ν_μ turns into ν_e

    θ_{12} = probability that ν_e turns into ν_μ or ν_τ

    θ_{23} = probability (close to 45°, nearly max.) that ν_μ turns into ν_τ

Charge-Pairity (CP) Violation—contradiction of conservation laws, charge conjugation, and pairity

- Complex phase angles invert spacial dimension
  - Change in charge sign
- Occurences in quarks too small to detect alone
- Explanations offered by Standard Model have yet to be proven

DUNE: Deep Underground Neutrino Detector — detectors at Fermilab and in South Dakota; anticipated 2026

- Graphite and Liquid Argon
  - Charged Lepton emitted or neutral current
    - Number of neutrinos and antineutrinos
  - Localize atmospheric oscillations
- Prototype detector ProtoDUNE at CERN
  - Testing AI programs
- Cherenkov radiation detected by light reflectors
  - Treat alternating orientations as pixelated images
- Seeking answers
  - Matter vs. antimatter
  - Sourced from core-collapse supernovae—black hole formation
  - Matter stability and grand unification

Problems:

- ν_τ hard to detect, looks like other particles
- ν_τ hits take up a lot of image space
  - 500 pixels x 500 pixels for just 4.5 mm
- Need extreme detail, lots of data
- 500 x 500 dataset doesn't entirely fit ν_τ
  - Smarter algorithms
- Data processed as images at various angles
  - Need flat selection efficiency over all angles

There are no standard methods to do this!

UNIVERSITY OF Cincinnati

Sparse Network— how the Aurisano Machine Learning Group is curating an algorithm

- Sparse tensors: only contain pixels with hits in them
  - Reduced amount of data and computational cost
- Train the network to have rotation invariance
- What could make a machine correctly identify ν_τ in the atmosphere?

Classes of particles: muons, pions, kaons, michel electrons, particle shower, diffuse scattering, and highly ionized particles (HIP)

Hyperparameters: learning rate (LR), weight decay (WD), gamma, step size, network depth

Optimization: loss function, activation function, automated optimizers

Optimization Algorithms: AdamW vs. Ranger

UNIVERSITY OF Cincinnati

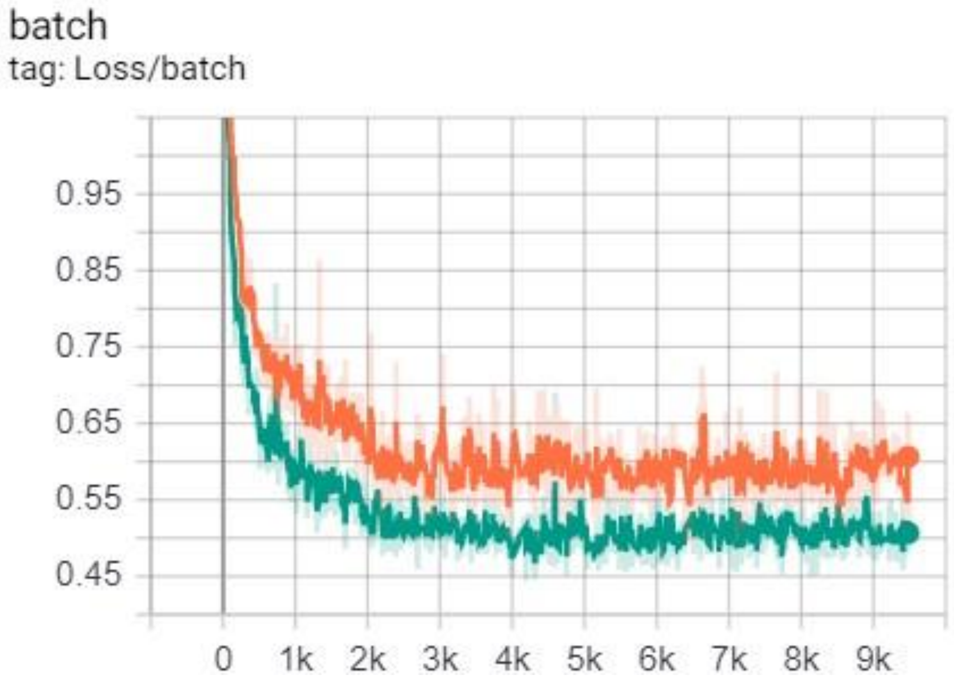
AdamW—adaptive moment estimation

- Adaptive Gradient Algorithm + Root Mean Square Propogation
- Memory usage minimized
- Efficient when properly tuned
- History of successful super convergence

Ranger—RAdam (Rectified Adam) + LookAhead

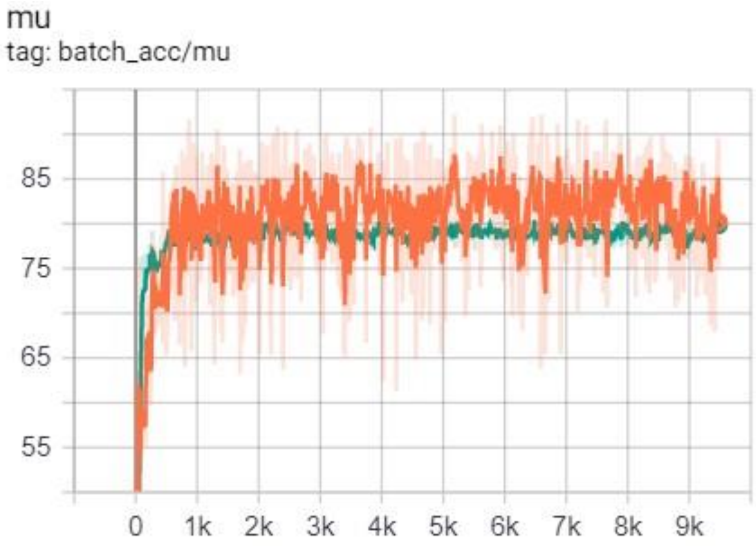
- Incorporates gradient centralization
  - Restricted loss function
- Quickly converges efficiency of multiple tasks
- Computationally efficient
- Compatible with Mish activation function

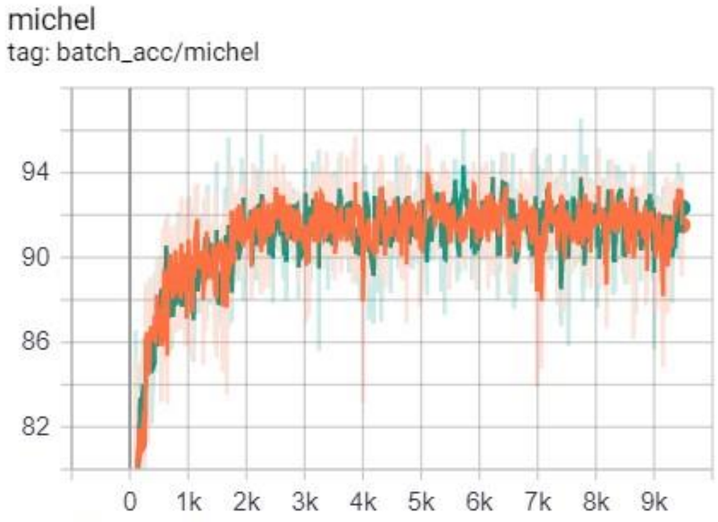
Default Values:
Weight Decay = 0.01, **Gamma** = 0.1

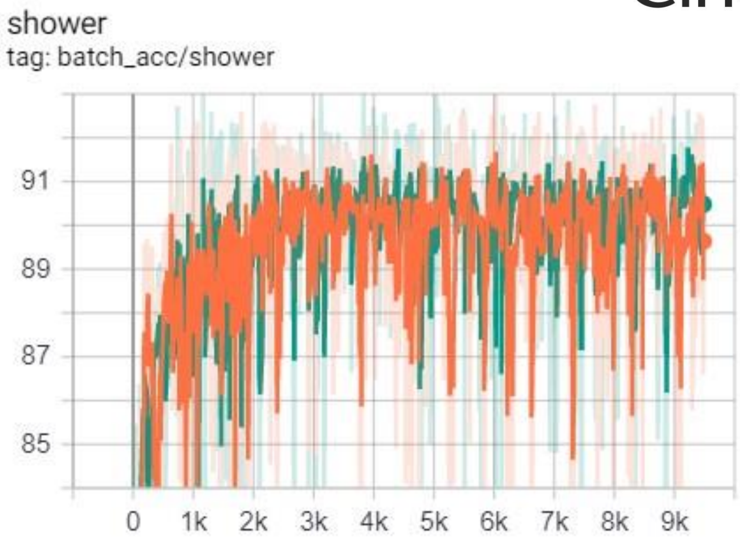


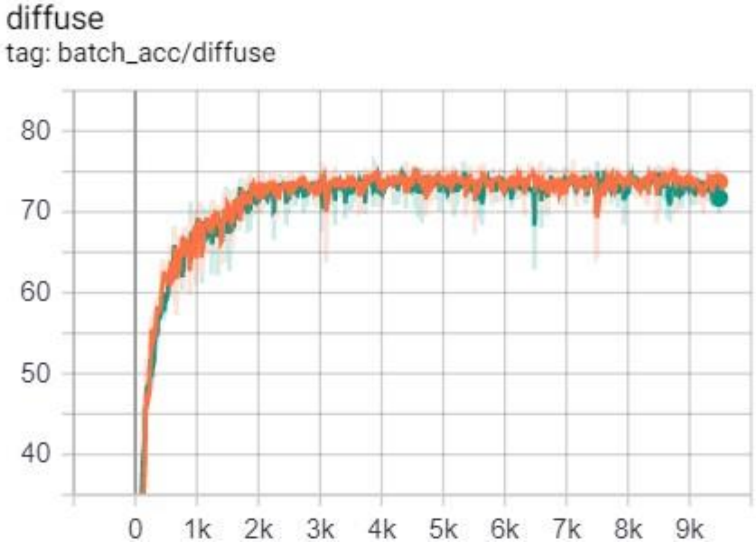
AdamW: End Accuracy = 79.48
Ranger: End Accuracy = 79.13

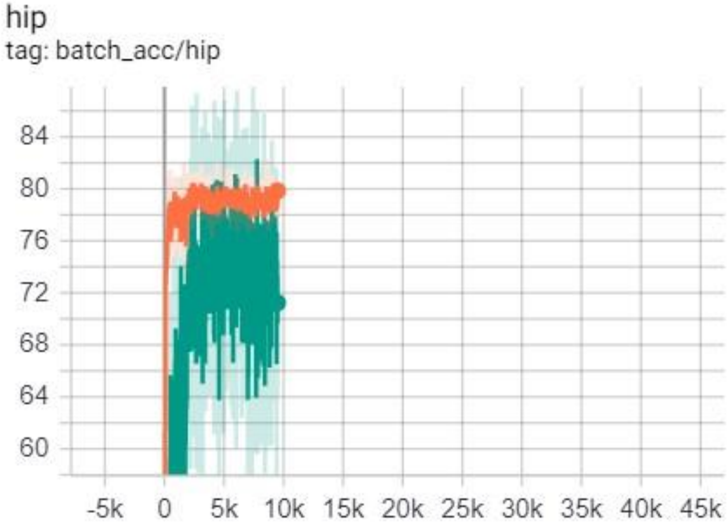
AdamW: End Loss = 0.4824
Ranger: End Loss = 0.6179







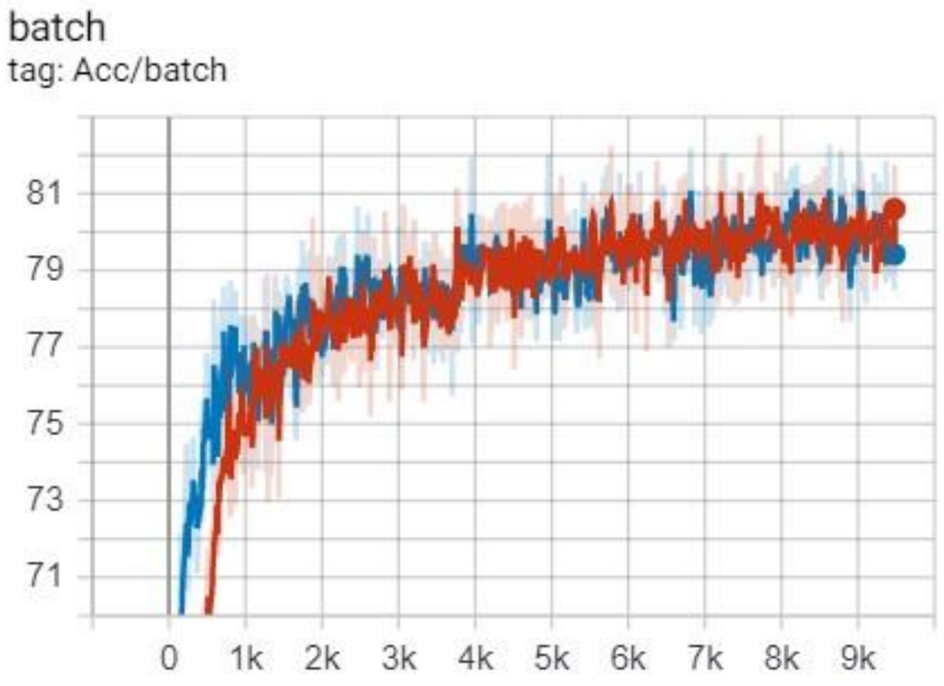


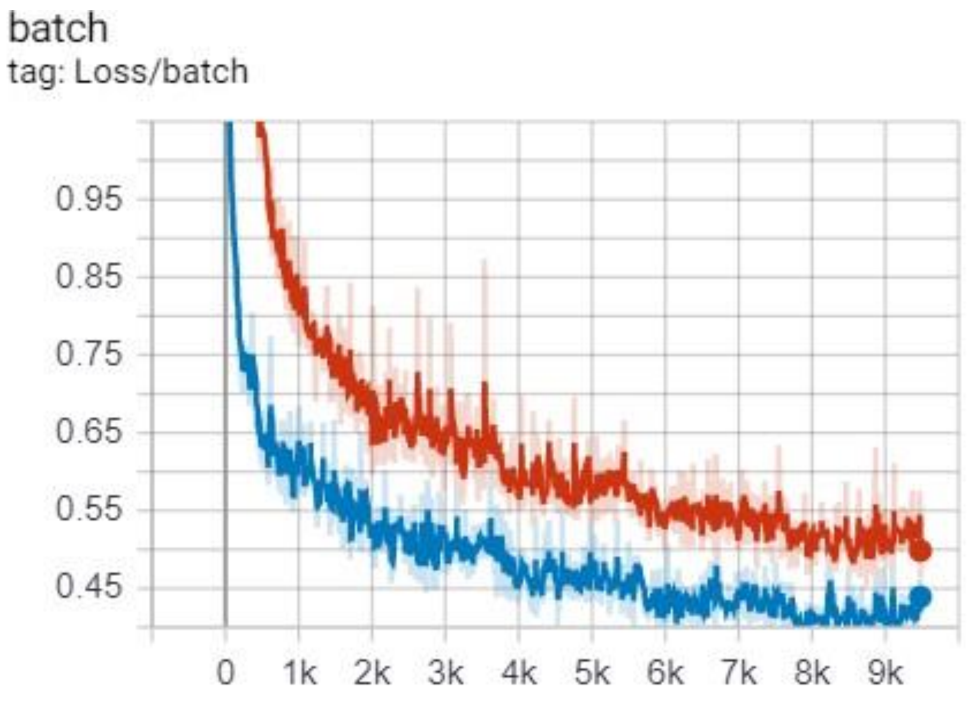


Conclusion: Ranger improved muon accuracy and HIP stability, but overall didn't outperform AdamW; no strong evidence that switching is worth it.

Chosen Values:

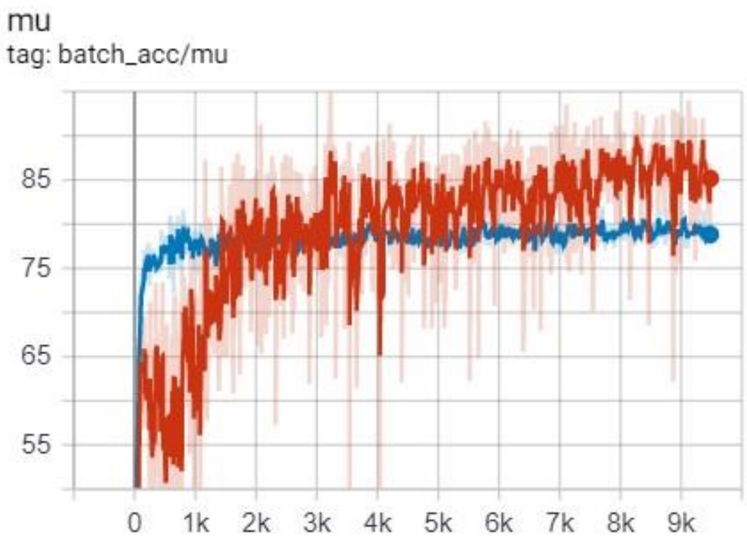
Weight Decay = 0.02, **Gamma** = 0.75

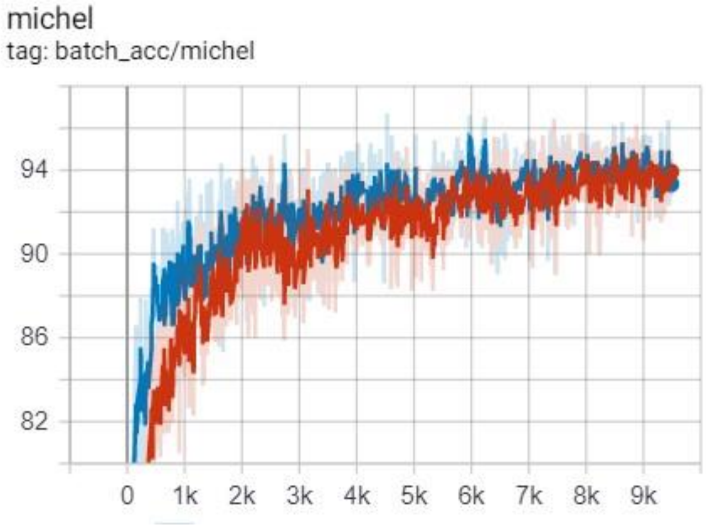


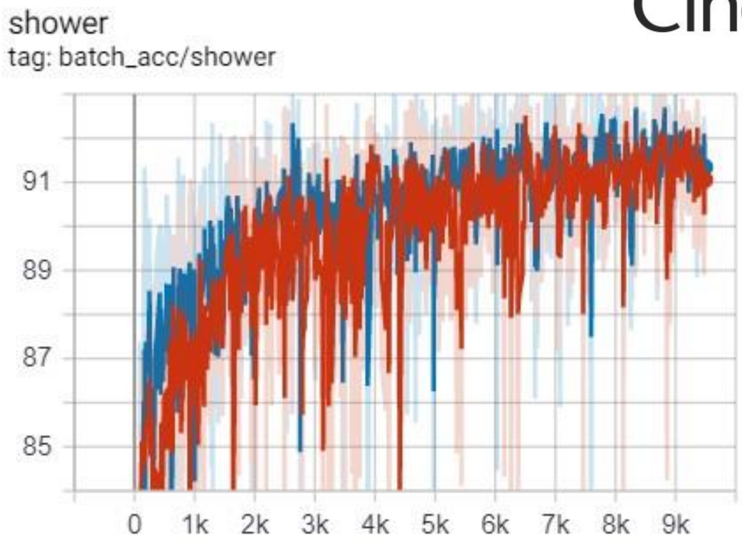


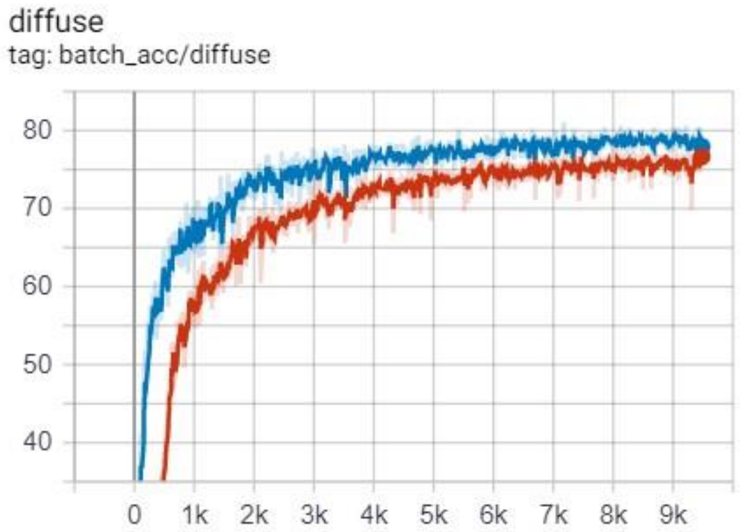
AdamW: End Accuracy = 78.48
Ranger: End Accuracy = 81.75

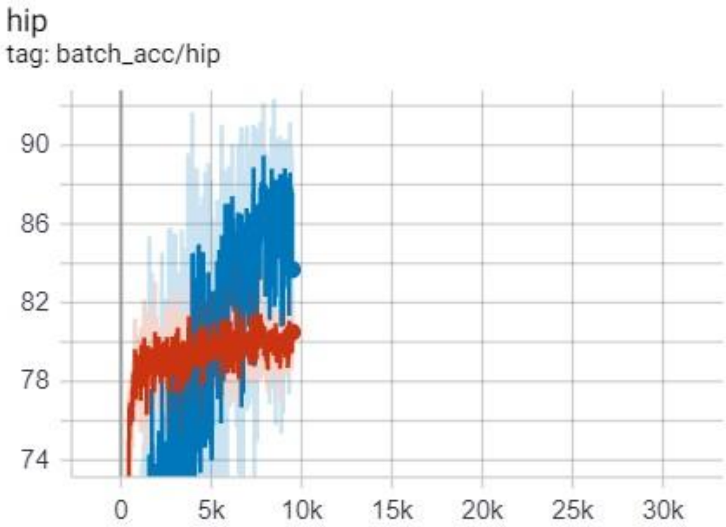
AdamW: End Loss = 0.4421
Ranger: End Loss = 0.4636











Conclusion: **Ranger** improved muon and diffuse accuracy, but overall didn't outperform **AdamW**; no strong evidence that switching is worth it.

Implementing SHERPA: "a Python Hyperparameter Optimization"

UNIVERSITY OF Cincinnati

Optimizing Parameters:

- Learning Rate

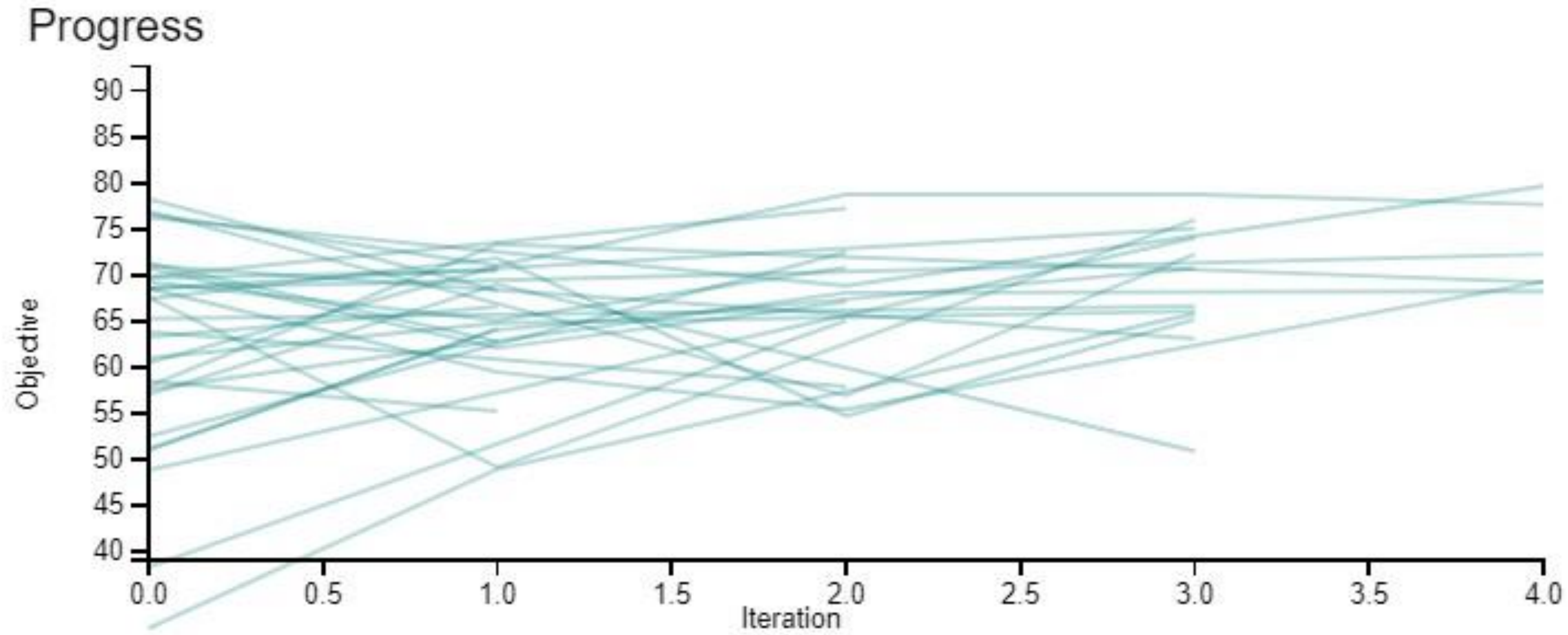
  Range = [0.0001, 0.1]

- Weight Decay

  Range = [0.01, 0.1]

- Network Depth

  Range = [2, 6]



Goals:

- Identify relationships: parameter values and objective score
- Narrow down ideal ranges for each parameter
- Test behavior with different algorithms

Parameters: LR, WD

Highest Accuracy: 70.8

- LR = 0.09907

Lowest Accuracy: 26.551

- LR = 0.08083

Algorithms—choosing and optimizing parameter values

Currently using: **GPyOpt Bayesian Optimization**

- Wrapper based on GPy
    - Gaussian modeling
- Good for many iterations

Loss Function: Currently using Categorical Cross Entropy, could consider customizing in the future.

Algorithms to consider in the future:

- **Asynchronous Successive Halving**—good for many hyperparametrs; stops early to reduce computational cost
- **Local Search**—analyzes small changes to the model; good when running fewer trials than GPyOpt

Moving Forward...

- Test the network on atmospheric datasets
- Push and identify limits of network parameters
- Verify previous results: AdamW vs Ranger

Moving Forward...

Unifying Workflow:

- Centralizing code for future use
- Implementing SHERPA
- Have been working with ProtoDUNE and NOvA data separately
- Translate from PyTorch dense tensors to MinkowskiEngine sparse convolution
    - Most activation functions are dense, so modify code to take only the sparse tensor's feature tensor, treat as dense

Preparing to test activation functions:

- LeakyReLU
    - Backpropogation
    - Downward slope for negative inputs
- SELU
    - ReLU for positive input, scaled exponential for negative input
    - Output: mean = 0, RMS = 1
- Swish
    - Bounded below, not above
    - Smooth

UNIVERSITY OF Cincinnati

Thank you!

Are there any questions?