Deep Learning for the Matrix Element Method

Mark Neubauer

University of Illinois at Urbana-Champaign

International Conference on High Energy Physics (ICHEP)

July 9, 2022 in Bologna, Italy

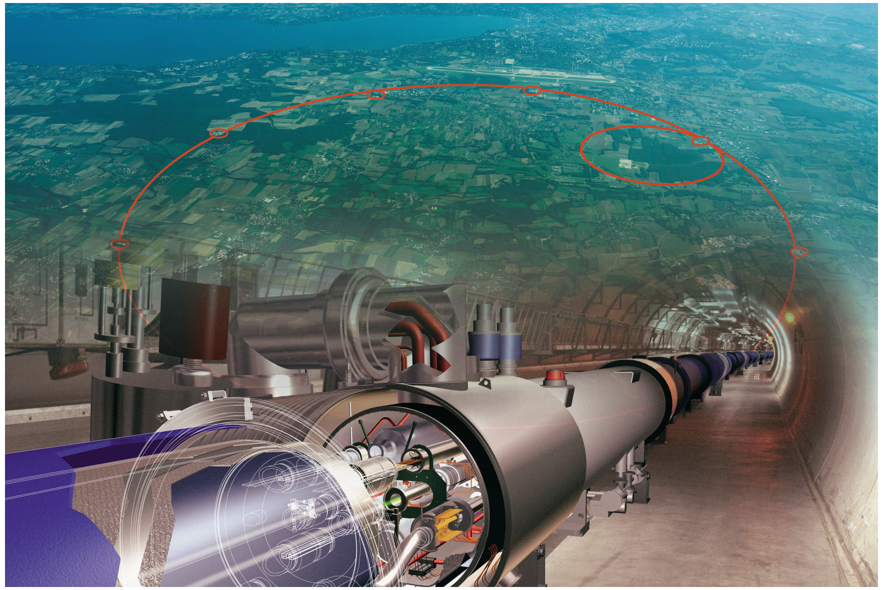
UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

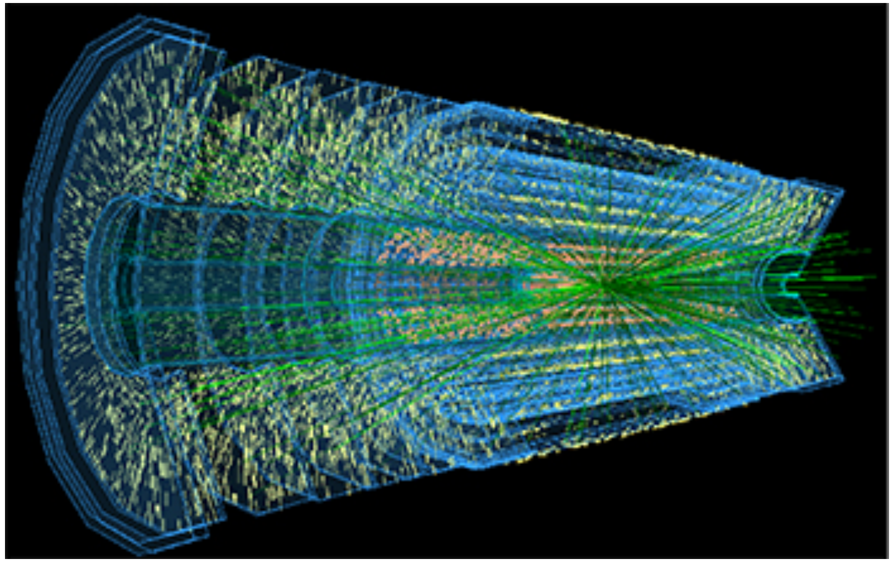
iris hep

SCAILFIN

Introduction

- The LHC's future is one of a dramatic increase in luminosity rather than energy
    - ➡ Large amount of collision data with complex events expected in future LHC running
- We want to make full use of this data by incorporating and correlating as much of the available information within each event as possible
    - ❖ Methods that employ machine learning are widely used in this context
    - ❖ Alternative: ***Matrix Element Method***





Matrix Element Method (MEM)

Ab initio calculation of an approximate probability density function $\mathscr{P}_\xi(\boldsymbol{x}|\boldsymbol{\alpha})$ for an event with observed final-state particle momenta $\boldsymbol{x}$ to be due to a process $\boldsymbol{\xi}$ with theory parameters $\boldsymbol{\alpha}$

$$\mathscr{P}_\xi(\mathbf{x}\,|\,\boldsymbol{\alpha}) = \frac{1}{\sigma_\xi^{\text{fiducial}}(\boldsymbol{\alpha})} \int d\Phi(\mathbf{y}_{\text{final}})\, dx_1\, dx_2\, \frac{f(x_1)f(x_2)}{2sx_1x_2}\, |\underbrace{\mathscr{M}_\xi(\mathbf{y}\,|\,\boldsymbol{\alpha})}_{}|^2\, \delta^4(\mathbf{y}_{\text{initial}} - \mathbf{y}_{\text{final}})\, W(\mathbf{x},\mathbf{y})$$

Dynamics from QFT→ Correlations from physics

$\mathscr{P}_\xi(\boldsymbol{x}|\boldsymbol{\alpha})$ can be used in a number of ways to search for new phenomena at particle colliders

[Sample Likelihood]
(e.g. $\boldsymbol{\alpha}$ measurements via max. likelihood)

$$\mathscr{L}(\boldsymbol{\alpha}) = \prod_i \sum_k f_k \mathscr{P}_{\xi_k}(\mathbf{x}_i\,|\,\boldsymbol{\alpha})$$

[Neyman-Pearson Discriminant]
(e.g. process search, hypothesis test)

$$p(\mathbf{x}\,|\,S) = \frac{\sum_i \beta_{S_i} \mathscr{P}_{S_i}(\mathbf{x}\,|\,\boldsymbol{\alpha}_{S_i})}{\sum_i \beta_{S_i} \mathscr{P}(\mathbf{x}\,|\,\boldsymbol{\alpha}_{S_i}) + \sum_j \beta_{B_j} \mathscr{P}(\mathbf{x}\,|\,\boldsymbol{\alpha}_{B_j})}$$

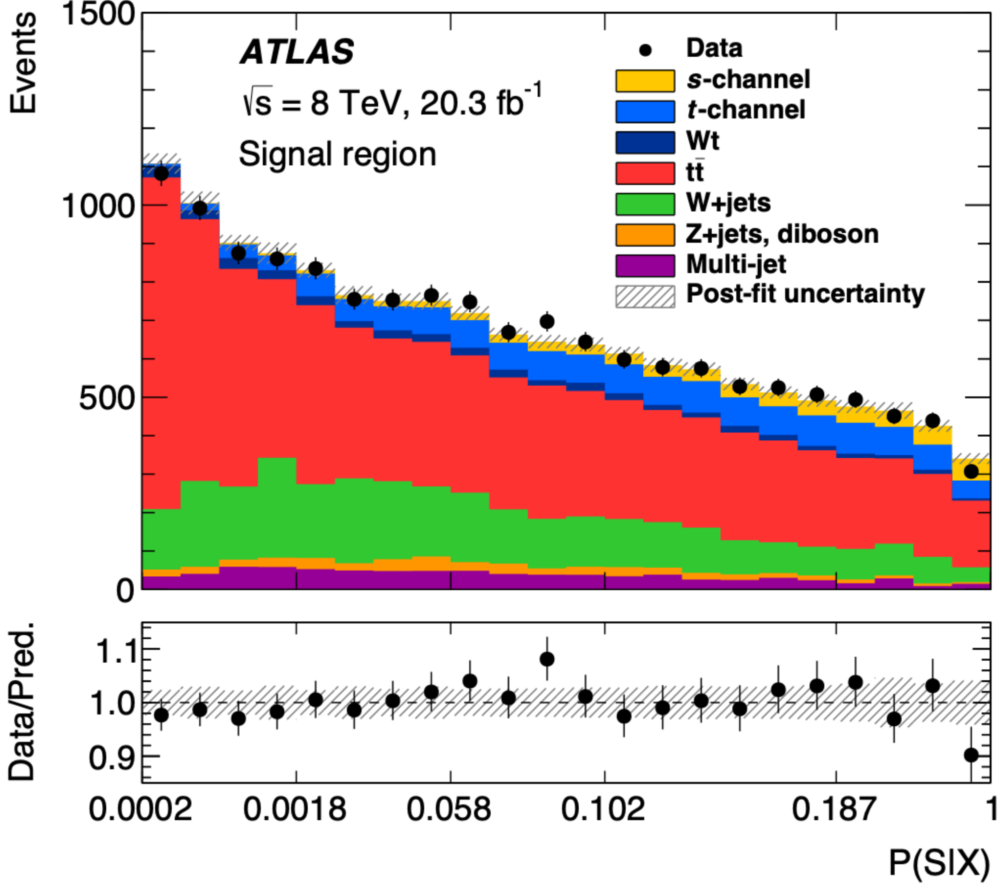
For the purpose of this talk: $\mathscr{P}_\xi(\boldsymbol{x}|\boldsymbol{\alpha})$ is a function that can be computed numerically and provides physics-driven information useful for measurements, hypothesis tests and searches

Matrix Element Method: Pros and Cons

- The ME Method has been used over the years for public physics results from collider experiments

- The ME Method has several advantages over ML-based methods
  - ❖ Does not require training
  - ❖ Incorporates all available kinematic information, including correlations
  - ❖ Has a clear physical meaning of transition probabilities in QFT

https://arxiv.org/abs/1511.05980



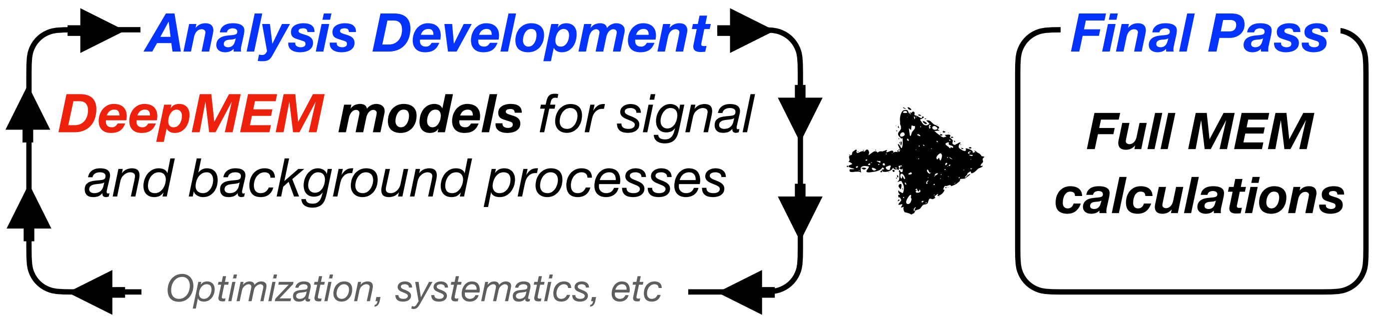
- The main limitation of the ME method: ***computationally intensive***
  - ❖ E.g. Calculating $\mathscr{P}_\xi(\boldsymbol{x}|\boldsymbol{\alpha})$ for $pp \to t\bar{t}H \to W^+bW^-\bar{b}b\bar{b} \to \ell\nu + 6$ jets involves high-dimensional integration and can take minutes per event

MEM in the Machine Learning (ML) Era

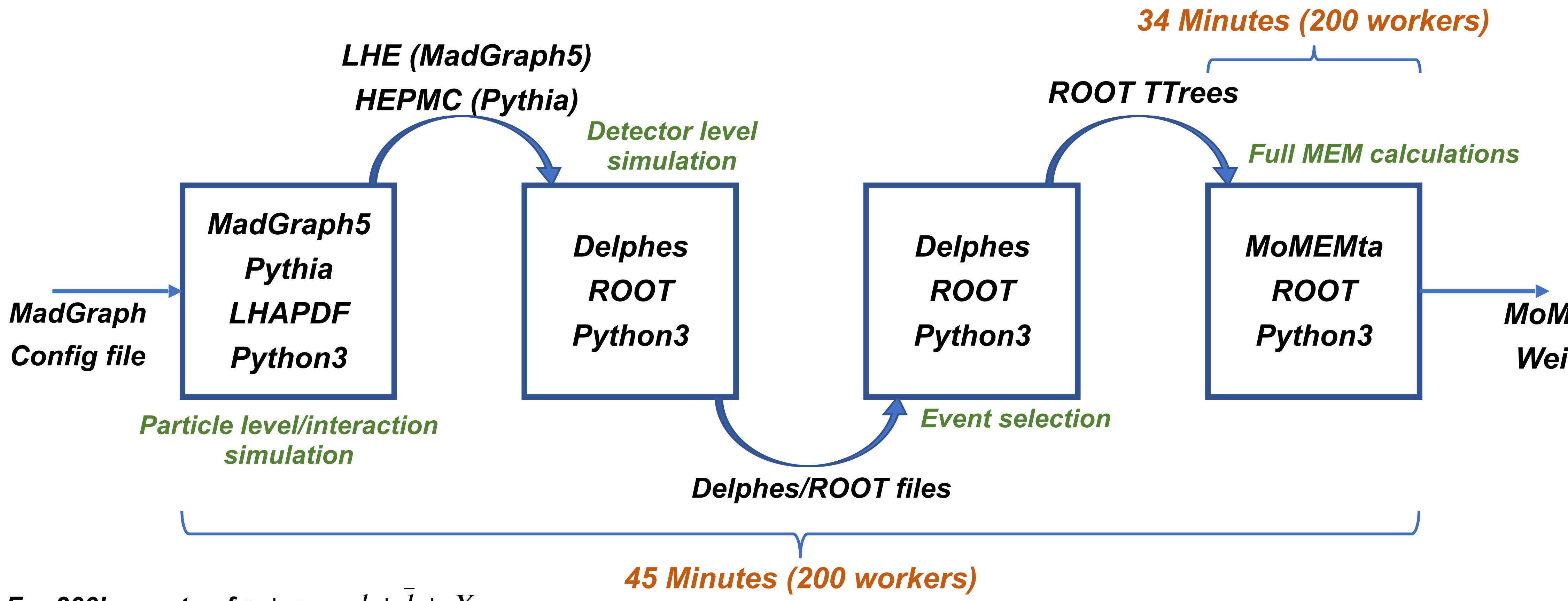
- It was proposed in 2017 by MN, *et al.* in [1] (cf. [2], [3], [4]) to use ML methods to approximate MEM calculations so they are *sustainable*

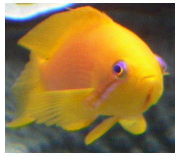
MEM Model Development

Possible Usage in Analysis



Current ME Method Calculation Pipeline





For 300k events of $\mathrm{p} + \mathrm{p} \rightarrow l + \bar{l} + X$

	Parallel Time	Serial Time
Entire Pipeline	45 Minutes	150 Hours
MoMEMta	34 Minutes	113 Hours

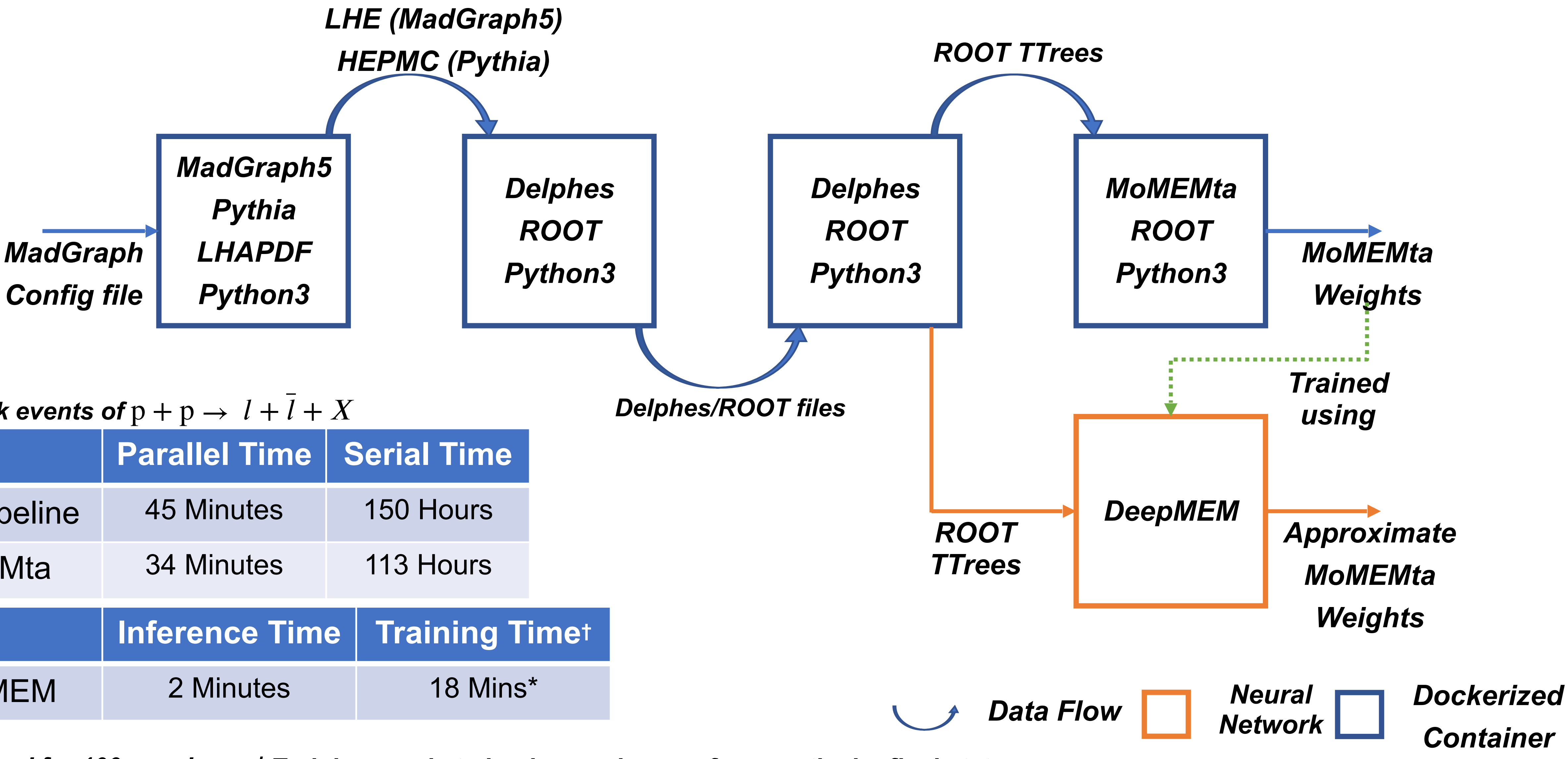
Dockerized Container

Data Flow

DeepMEM Objectives

- Address the challenges of MEM while retaining the benefits:
  - Retain the transparency and accuracy of MEM calculations
  - Reduce the time required by MEM calculations
- **Deep Neural Networks:** arbitrary function approximators that scale well with data
- Replace the calculations performed by MoMEMta with a deep neural network trained using MoMEMta outputs
- Final calculations used in an analysis would be performed using the full pipeline for accuracy – Using DeepMEM expedites calculations during research and development

MEM Pipeline using DNN Approximations



For 300k events of $p + p \to l + \bar{l} + X$

	Parallel Time	Serial Time
Full Pipeline	45 Minutes	150 Hours
MoMEMta	34 Minutes	113 Hours

	Inference Time	Training Time†
DeepMEM	2 Minutes	18 Mins*

** Trained for 100 epochs* *† Training needs to be done only once for a particular final state*

Data and Selection Description

We consider the simple Drell-Yan Process with lepton pair final state:

$$\mathrm{p} + \mathrm{p} \rightarrow l + \bar{l} + X$$

- Parsing the ROOT TTrees produced after event selection, we use the 4-momentum of the final state particles and MET

- Mass is a very good discriminant, and we keep the neural network blind to the mass by excluding it (following the approach of [4])

- Inputs:
  - Pt, Eta, Phi components of leptons and jet(s)
  - Magnitude and Phi of the MET
  - 14 input parameters

- Outputs:
  - Log transformed MoMEMta weight values

- Final Dataset contains ~300K events

Multiprocessing DataLoader

- PyTorch In-built dataloader is built for image/computer vision data – loads individual samples based on user mappings
    - This is inefficient for contiguous, tabular data

- No out-of-the-box solution that can address the issues
    - Tabular data is loaded faster in chunks
    - The dataset might be too large to fit in memory at once

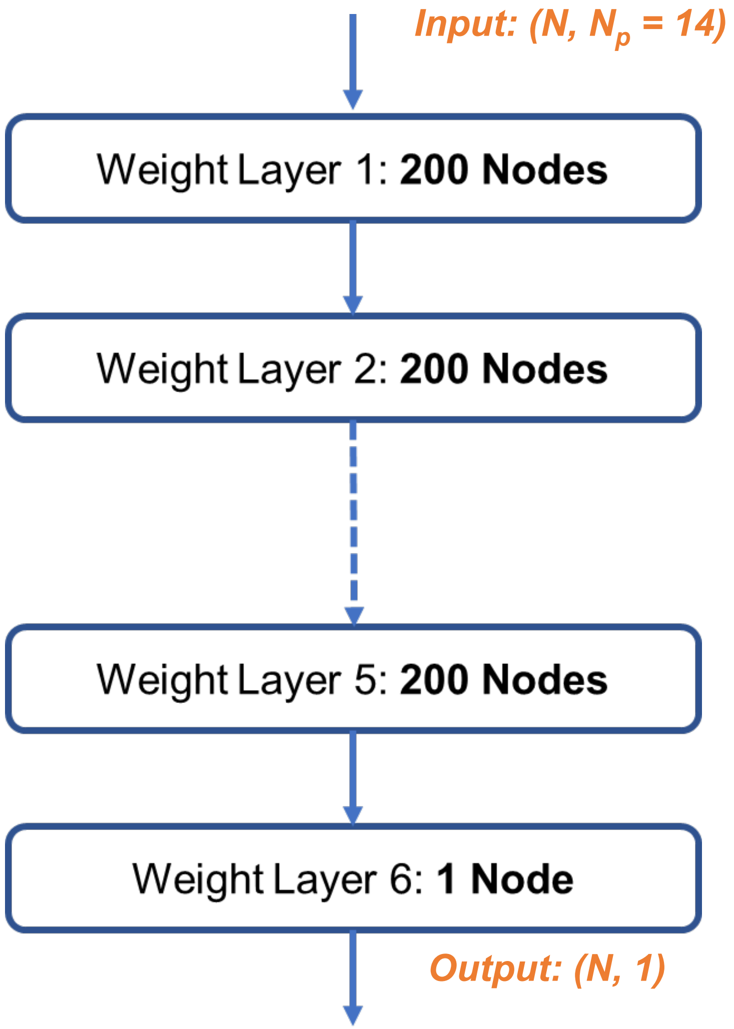
- Data Managing and Loading Module:
    1. Parse ROOT TTrees based on user-input
    2. Use Python Multiprocessing library constructs to store a "cache" of data
    3. Spawn processes using PyTorch to load data from cache
    4. Load the next chunk of data and replace the "cache"

- We get significantly faster data loading in comparison to the in-built dataloader

Load times are for 100 epochs of the MoMEMta test dataset

	Load Time
In-Built	506 s
Our Implementation	55 s

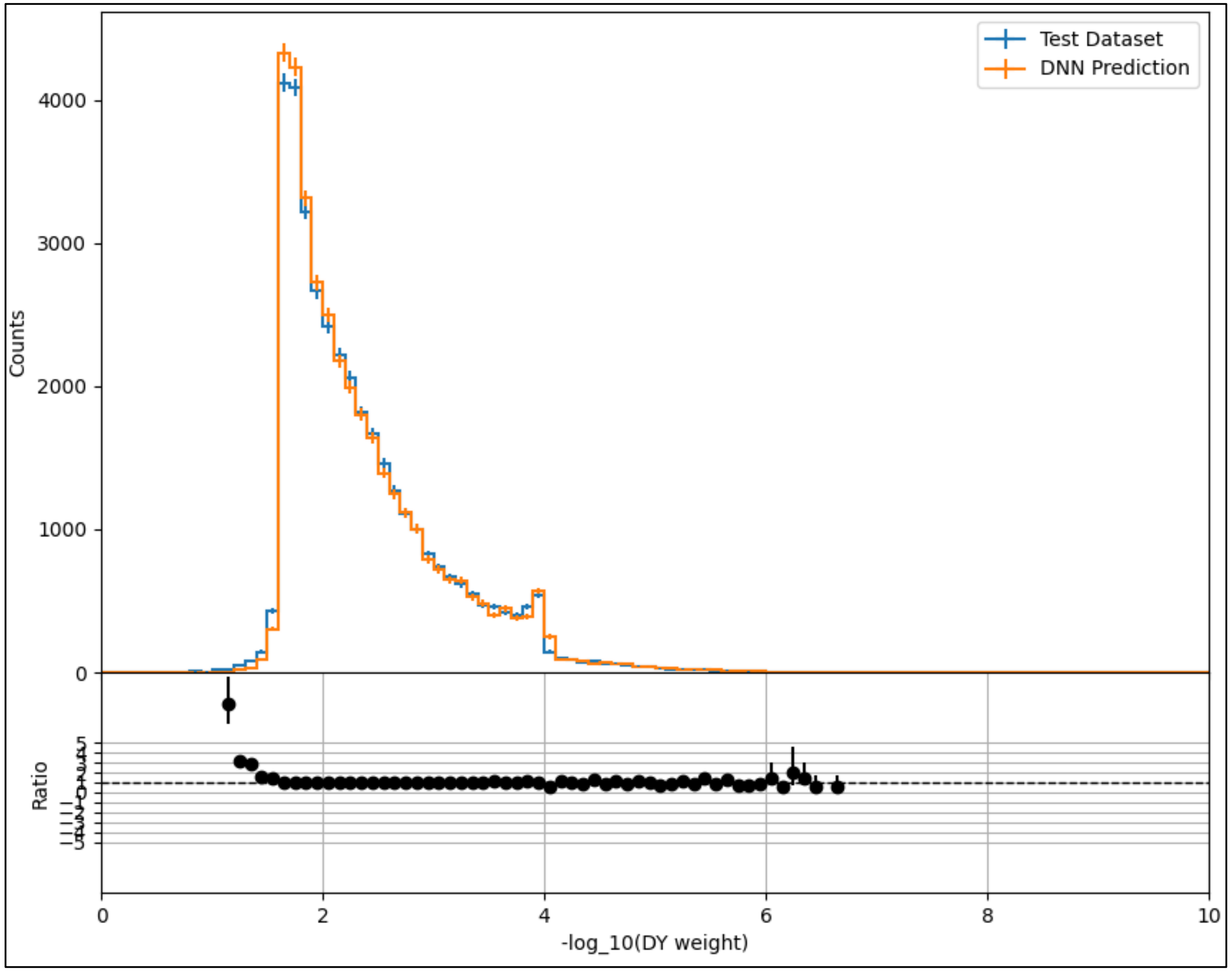
Network Architecture



DNN A: *5 Deep and fully-connected Layers*

- We use a Fully-Connected Deep Neural Network with 5 deep layers of 200 Nodes each
- We split the data 8:1:1 for training, validation and testing purposes
- The output is the approximate transformed MoMEMta weights for N = ~270k training and validation events
- The network is trained for 100 epochs

Results using DNN



Results from DNN A: *5 Deep and fully-connected Layers*

- Testing on **unseen data** gives us a good visual fit between DeepMEM predictions and the test data
- Ratio = $\dfrac{\#\, of\ predicted\ events\ in\ bin}{\#\, of\ actual\ events\ in\ bin}$
- Mean Absolute % Error = **1.6%**

where $\mathrm{MAPE} = \dfrac{100}{n} \sum_{t=1}^{n} \left| \dfrac{A_t - F_t}{A_t} \right|$

- However, we can see that the network cannot generalize well on bins that do not contain a lot of events

Residual Networks

Residual Networks are neural network architectures that **incorporate skip connections** in the network architecture

Ease training for deep networks by **providing shortcuts for backpropagation**, while gaining accuracy from the depth of the network (see ref [5])

ResNets have empirically shown to have better results for aggressively deep networks (ILSVRC 2015) [5]

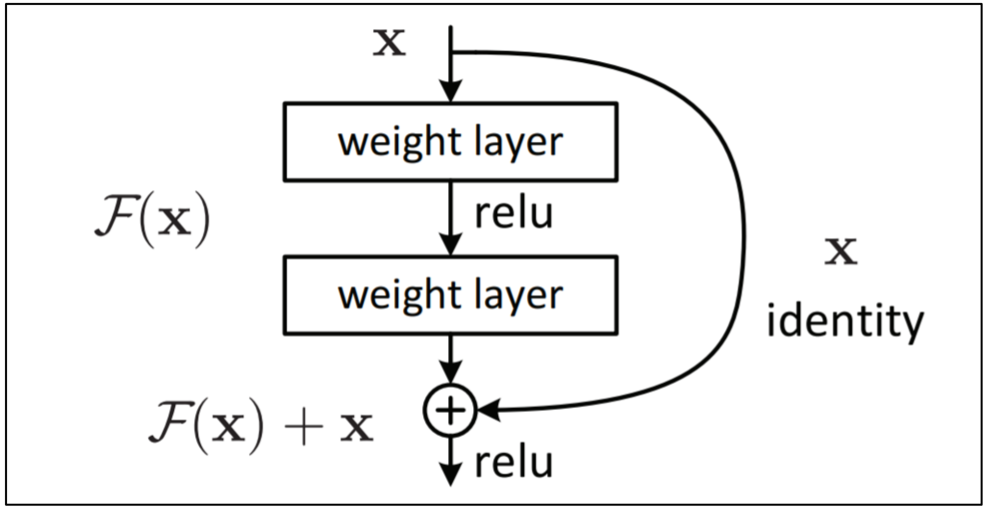
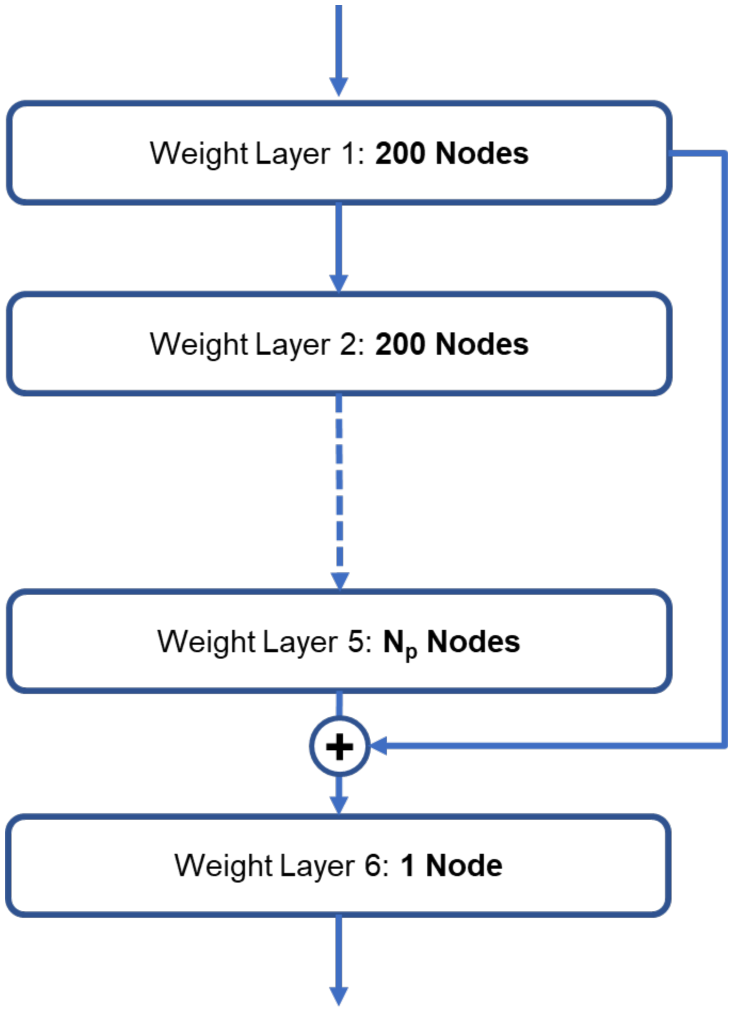


Image credit & Ref: [5] K. He, X. Zhang, S. Ren, J.Sun, Deep Residual Learning for Image Recognition

Why do ResNets work?

- They address the gradient vanishing phenomenon
- Smaller loss values can successfully transmit through a deep network and update the earlier layers

Residual Network Architecture



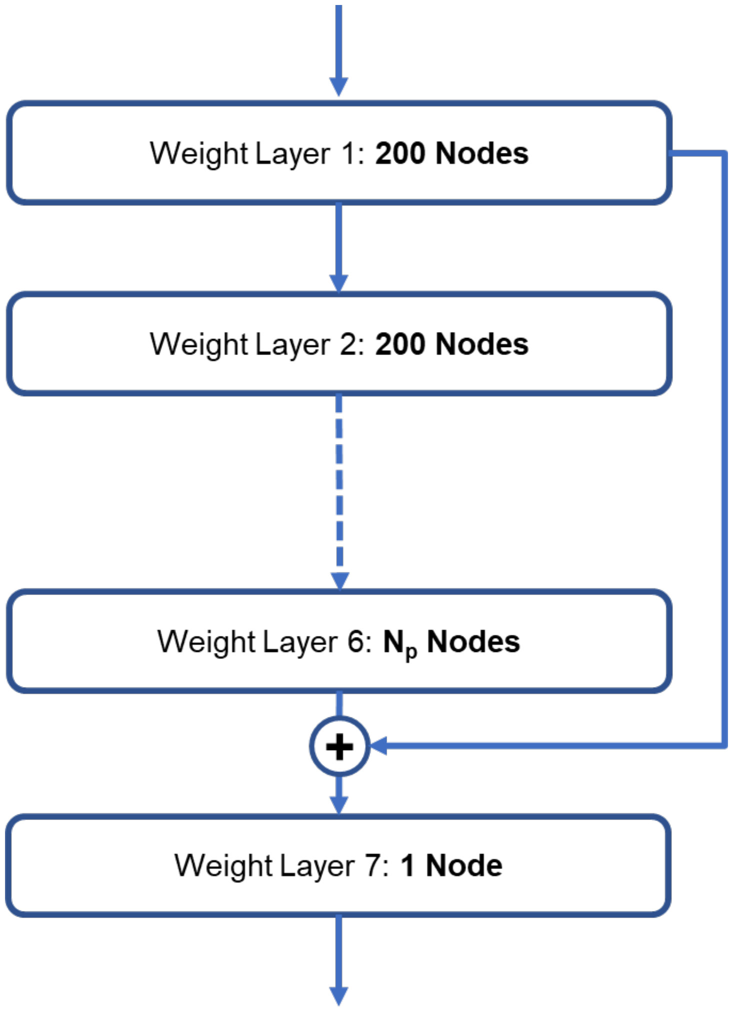
ResNet A: *5 Deep Layers followed by a skip connection*

We include a skip connection into the original DNN A while retaining Depth

(This Network is less complex than DNN A)

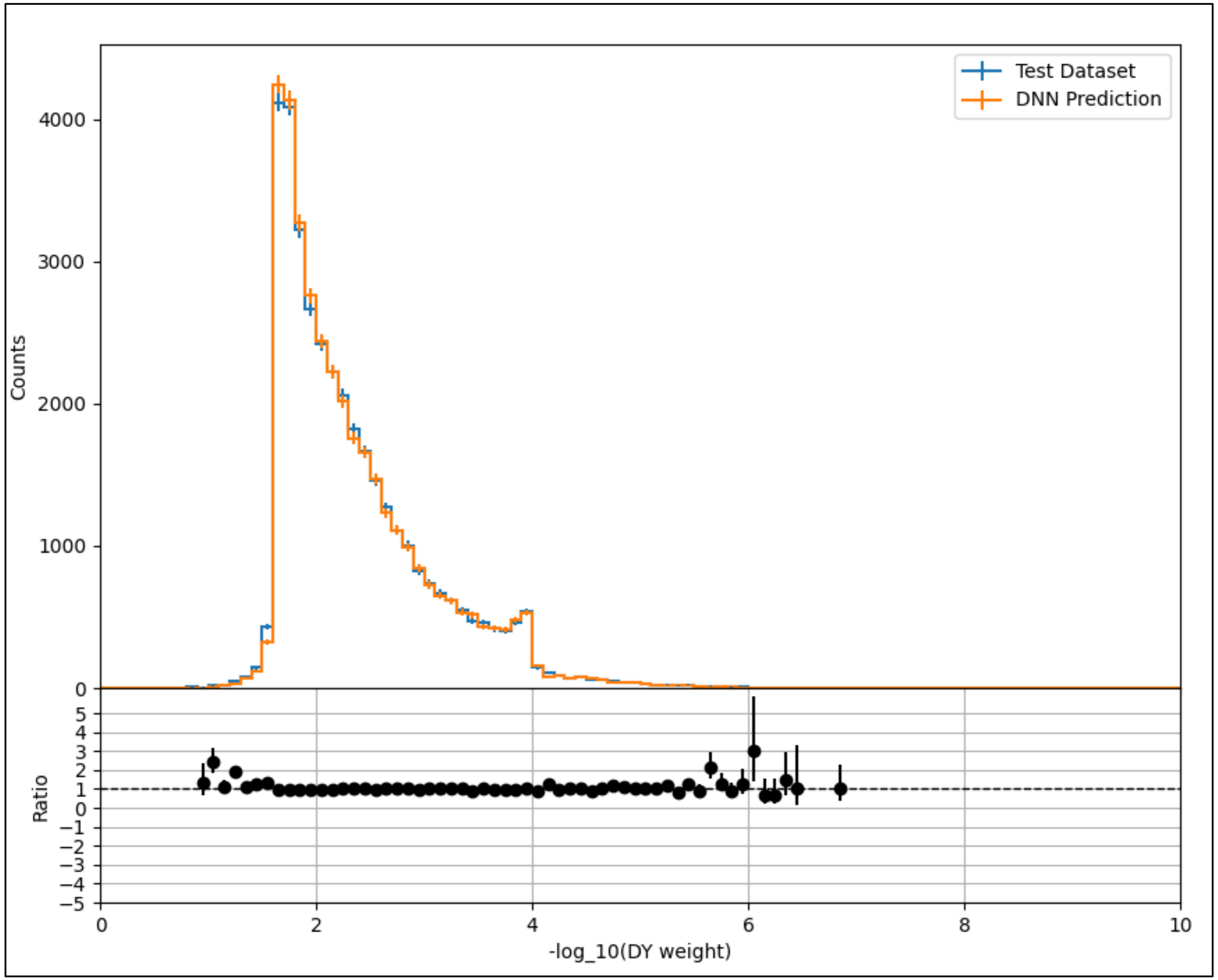
We include a skip connection into the original DNN A by adding an extra layer to the depth

(This Network is more complex and deeper than DNN A)



ResNet B: *6 Deep Layers followed by a skip connection*

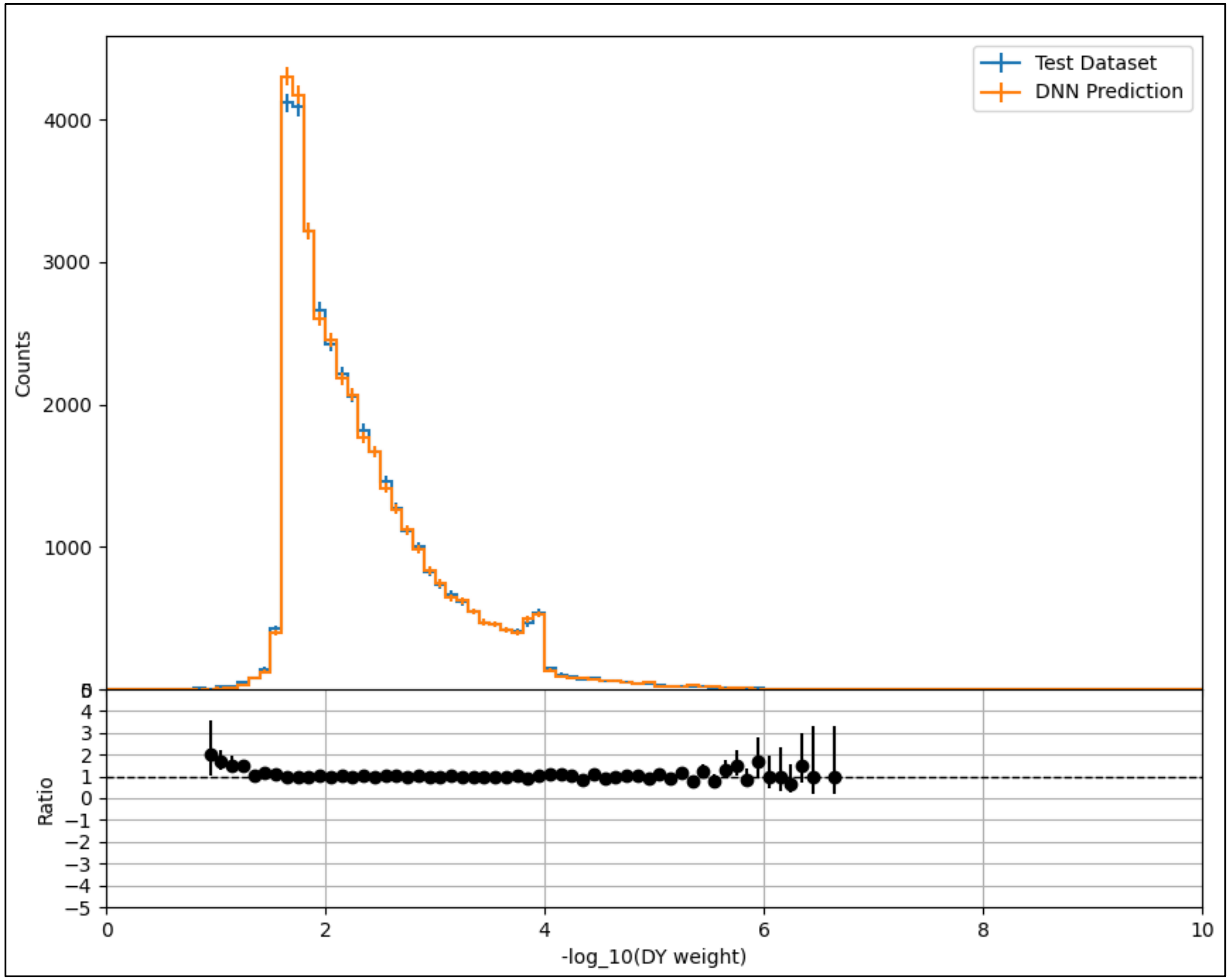
Results using Residual Network A



Results from ResNet A: *5 Deep Layers followed by a skip connection*

- We see much better generalization using this architecture
- Mean Absolute % Error = **1.4%**
- We argue that adding a skip connection improved the results since ResNet A was less complex than DNN A

Results using Residual Network B

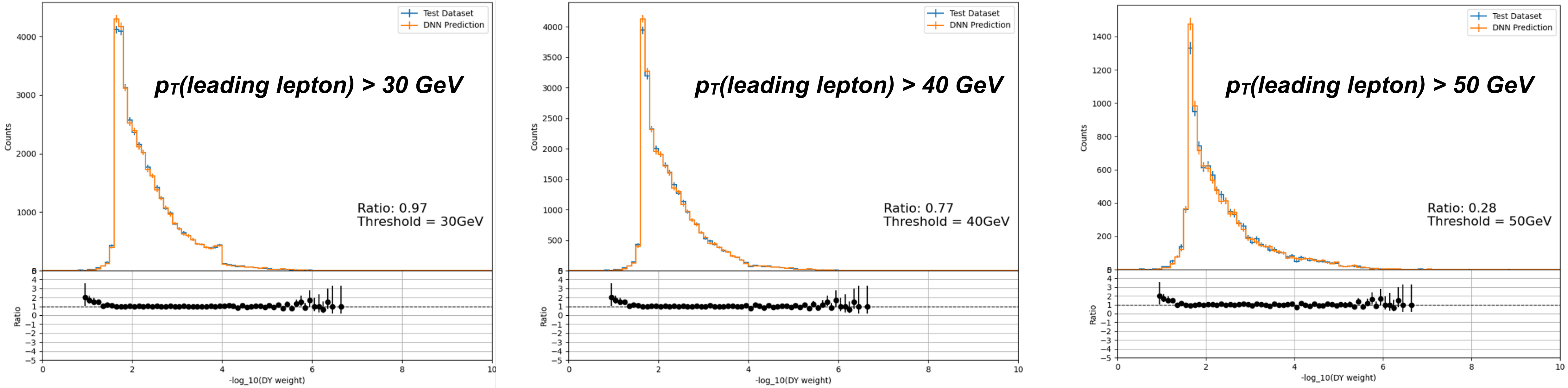


Results from ResNet B: *6 Deep Layers followed by a skip connection*

- We see even better generalization using this architecture
- Mean Absolute % Error = **1.2%**
- A more complex network with a skip connection gives us slightly better results by leveraging its depth

Generalization in Kinematic Phase Space

Check ResNet B modeling on different kinematic subsets of the data (no retraining!)



Good modeling retained ⟶ robust against leading lepton pt cut

Similar results for subsets through jet pt thresholds

Summary and Future Work

- Implemented ML methods to approximate MEM calculations and demonstrated the viability of this approach on a simple DY process
- Implemented a versatile and performant Multiprocessing Dataloader
- Implemented Residual Network architecture for better generalization
- Checked that the model is robust against kinematic selections

❖ A next step in this study is to go beyond the simple DY process to other processes with more complex decays and final state particles

❖ Explore other ML architectures which include physics constraints

❖ Generate simulated data and models adhering to FAIR principles

  ❖ See talk by Avik Roy (UIUC) in this session

DeepMEM is an **open-source python library distributed on PyPI** that can be used on similar datasets: `python -m pip install deepmem`

Acknowledgments

- This work was performed by Mihir Katare and Matthew Feickert, with guidance from Avik Roy

Mihir Katare

Matthew Feickert

Avik Roy

- This work was supported through grants from the National Science Foundation: IRIS-HEP (OAC-1836650) and SCAILFIN (OAC-1841456)