Deep Learning Techniques for Track Reconstruction in $\overline{\text{P}}$ANDA

Adeel Akram

Uppsala University
on behalf of the $\overline{\text{P}}$ANDA Collaboration
adeel.akram@physics.uu.se

$\overline{\text{P}}$ANDA Collaboration Meeting
4-8 June 2018
Stockholm, Sweden

Outline

- Conventional tracking approach
- Motivation for new methods
- Intro to Machine Learning
- Deep Learning
- Programing Frameworks
- Summary

Track Reconstruction

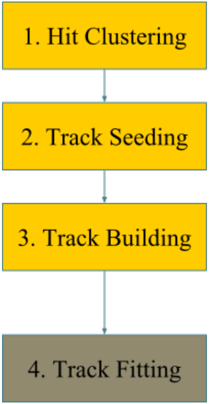
Common approach in Nuclear and Particle Physics (NPP):

Track finding

- Pattern recognition and/or classification
- Find tracklets
- Find ghost tracklets

Track fitting

- Input: tracklets
- Output: track kinematics



Track Reconstruction

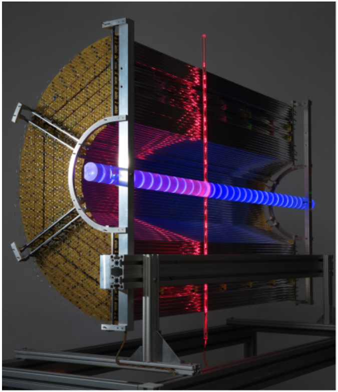
Conventional tracking methods suffer one or more of the following issues. These models

- Rely on linear dynamic models
- Are serial in nature
- Scale badly with track multiplicity
- Consume huge computing resources

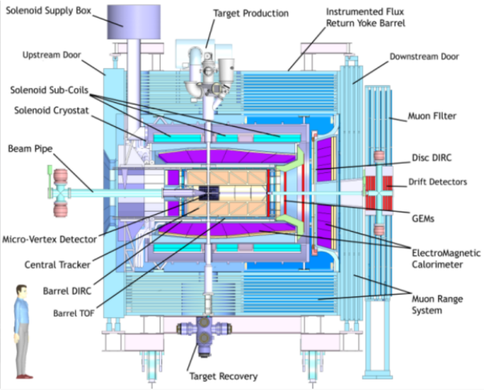
An alternate approach is using Machine Learning (ML) methods.

- Pattern recognition (*Track finding*)
- Extraction of track parameters (*Track fitting*)

PANDA Experiment



Straw Tube Tracker (STT)



Target Spectrometer

Earlier work in Deep Learing for $\overline{\text{P}}$ANDA

Two student projects have been conducted at Uppsala University in 2017 using Deep Neural Networks. Two approaches has been used

- Standard DNNs with Supervised Learning
  - With 95% prediction accuracy
- Convolutional DNNs with Supervised Learning
  - With 80% prediction accuracy

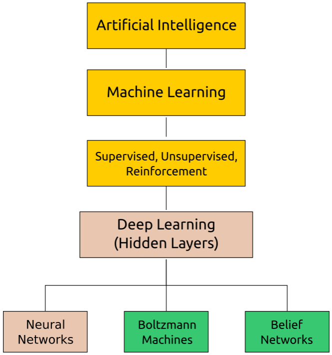
Machine Learning

Ability of machines to learn complex representations of data without explicitly programmed for this purpose.

Learning Schemes:

- Supervised Learning
  - Classification, Regression
- Unsupervised Learning
  - Clustering, Density Measurements
- Reinforcement Learning
  - Robotics etc

Deep Learning: Coming next...



Deep Learning

Standard Learning: Deep learning is an approach to introduce multiple hidden layers in the existing model. For example, we have

- Deep Neural Networks
  - Standard, Convolutional, Recurrent
- Deep Boltzmann Machines
- Deep Belief Networks

Adaptive Learning: Competitive hypotheses are introduced in a way that final outcome depends on current observation.

- Hopfield Neural Networks
- Elastic Nets
- Gussian-sum filters

Deep Neural Networks (DNNs)

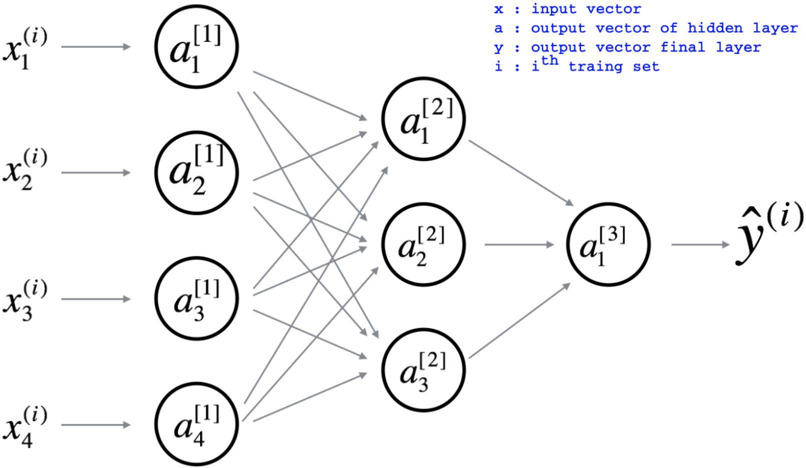


Figure: A three layered standard DNN

Mathematics of DNNs

Forward propagation:

$$z^{[l](i)} = W^{[l]}.x^{(i)} + b^{[l]}$$

$$a^{[l](i)} = ReLU(z^{[l](i)})$$

$$\hat{y}^{(i)} = \hat{a}^{[l](i)}$$

Where,
z: linear equation
b: bias or intercepts
W: weight matrix
$ReLU$: **activation function**
a: output of l^{th} layer
$\hat{y}$: estimate of final layer

The prediction:

$$y^{(i)}_{prediction} = \begin{cases} 1, & \text{if } \hat{y}^{(i)} > 0.5 \\ 0, & \text{otherwise} \end{cases}$$

Loss Function:

$$L(\hat{y}^{(i)}, y^{(i)}) = -1/m \sum_{i}^{m} [\hat{y}^{(i)} \log(y_i) + (1 - \hat{y}^{(i)}) \log(1 - y_i)]$$

Mathematics of DNNs

Error backpropagation

Using method of gradient descent:

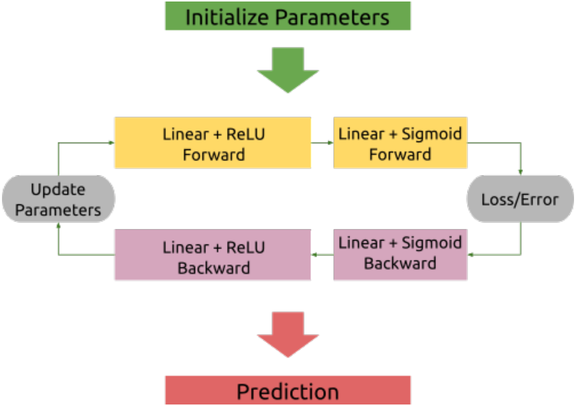
$$dW^{[l]} = \frac{\partial L}{\partial W^{[l]}}$$
$$db^{[l]} = \frac{\partial L}{\partial b^{[l]}}$$
$$da^{[l-1]} = \frac{\partial L}{\partial a^{[l-1]}}$$
$$dz^{[l]} = da^{[l]}.g^{'}(z^{[l]})$$

Parameter Update

Parameter update with the learning rate α:

$$W^{[l]} = W^{[l]} - \alpha dW^{[l]}$$
$$b^{[l]} = b^{[l]} - \alpha db^{[l]}$$

Flow Diagram



Programming Framworks

C++ Environment:

- Standard C++/ROOT
- TMVA (Toolkit for Multivariate Analysis)
- etc.

Python Environment:

- Python 3.0
- Numpy for Vectorization
- TensorFlow/scikit-learn
- etc.

TrackML Competition 2018

Summary

- Earlier work has shown "promising" results
- Different learning schemes and DNN topologies should be investigated further
- DNNs can be expanded both for online and offline data processing

What Next

- Improvements on the earlier work
- Conversion from Matlab to TensorFlow
- Reproduce results using TensorFlow

Backup

Improving DNNs

Hyperparameters tuning

- Weights, bias, learning rate etc.

Regularization

- L2 Regularization
- Dropout Regularization

Optimization

- Adam Optimizer (used in PANDA, Stud. Project)
- Genetic Algorithm

Normalization

Additional Resources