How easily can neural networks learn relativity?

ACAT 2017

Kartik Chitturi
Dr. Peter Onyisi

August 24th, 2017
Introduction
• Relativistic invariants are key variables in HEP problems and are believed to be learnt implicitly by deep learning approaches.

• We investigate the minimum network complexity needed to accurately extract such invariants. Doing so will help us understand how complex a neural net needs to be to obtain certain functions.
We used Keras (with a Tensorflow backend) on a NVIDIA Tesla K80.
How complex?
Essentially a functional fit with many parameters

Problem:
Networks with > 1 layer are very difficult to train.

Consequence:
Networks are not good at learning non-linear functions.
(like invariant masses!)

In short:
Can’t just throw 4-vectors at NN.

Neural Networks
Essentially a functional fit with many parameters

Taken from Daniel Whiteson, *Deep Learning in Particle Physics*,
ACAT, August 23rd, 2017
Neural Network Architectures

We take in 4-vectors of events and study the following problems using NNs.

- $p_T$ of $Z \rightarrow \mu\mu$
- $p_T$ of $t\bar{t} \rightarrow \mu\mu$
- Invariant mass of $t\bar{t} \rightarrow \mu\mu$

We adjusted these hyperparameters and studied how this affected accuracy:

- # Nodes
- # Layer
- Activation Function
Finding $p_T$ of $Z \rightarrow \mu\mu$ and $t\bar{t} \rightarrow \mu\mu$
We see that with a single layer, we train well.
1 layer with 10 nodes is enough for almost perfect accuracy.
Looking at the weights generated for the hidden layer, we find:

- $E$ and $p_z$ are given weights of 0
- The weights for $p_x$ and $p_y$ are symmetric for both muons

Thus neural network is learning a **non-linear** function of

$$(p_{x1} + p_{x2}) \text{ and } (p_{y1} + p_{y2})$$
Adding an additional layer provided no increase in accuracy. However accuracy actually dipped with both softplus and ReLU activations for when we had a single node in the 2nd layer. This same problem occurs for $p_T$ of $t\bar{t}\rightarrow \mu\mu$. 

Heatmap using ReLU 

Log of Cost 

# nodes in layer 1  

0 1 2 3 4 5 6 7 8 9 10 

# nodes in layer 2  

1 2 3 4 5 6 7 8 9 10 

$Z \rightarrow \mu\mu$
ReLU and Softplus suffer from having a zero gradient for negative inputs. During training, neurons ‘die’ when they get put in a state where the output 0 for all inputs.
LeakyReLU was specifically made to fix this.
Thus using LeakyReLU made this problem go away.
Dropout may have also worked.
We see that the same accuracy occurs with $t\bar{t}$ confirming sample independence for finding $p_T$ of dimuon production.
Finding Invariant Mass of $t\bar{t} \rightarrow \mu\mu$
As seen before, a single layer with 10 nodes can very accurately predict $p_T$. 

![Graph of $p_T$ for $Z \rightarrow \mu\mu$, 10 Nodes](image1)

$r = 0.999957107541$

![Graph of $p_T$ for $t\bar{t}$, 10 Nodes](image2)

$r = 0.99991019276$
Invariant Mass of $t\bar{t} \to \mu\mu$

Using the same neural net architecture (1 layer with 10 nodes, LeakyReLU) we get a lower accuracy ($r = 0.97$). We get a better accuracy with 20 nodes in a single layer.
Invariant Mass of $t\bar{t} \rightarrow \mu\mu$

We finally achieve similar accuracy as $p_T$ when given a large number of nodes in 2 layers.
We see little change happening across neural nets when they have a low number of nodes.
Summary

Used neural nets for regression of transverse momentum and invariant mass in two-body systems.

LeakyReLU was used in all neural nets (other activation functions were experimented with).

A single layer with 9/10 nodes has almost perfect accuracy for finding $p_T$ (in a certain range) of dimuon production.

Sample-independence for finding $p_T$ was shown through $t\bar{t}$.

Still trying to understand what features of the mass problem is causing the issues.

We plan to look into other invariants such as decay angles.