Summer ML Triggers

Week 4 Report

Russell Zhang
dg22882@bristol.ac.uk

Supervisor Dr Sudarshan Paramesvaran

Particle Physics Laboratory

Tables

Part 1

Implementation

Content:

- 1^{st} step: Our Raw dataset is in ROOT format. Firstly, I transformed ROOT file into awkward arrays which is a kind of non-regular arrays package. And then I use None value to fix awkward arrays as regular arrays. Next, I normalised data to make them refer to same distribution. Finally can transformed these awkward arrays into numpy arrays which are readable for Machine Learning.
- 2^{nd} step: Based on author's code, I made some adjustment to our own dataset and successfully run ParticleNetLite++ on linux server using GPUs. Author's neural networks architecture aims to multi-categorical jet tagging problem. So its parameters and network size are very huge. But our task is binary classification problem and we have computational cost. So I simplify the architecture and reduce the parameters of NN.

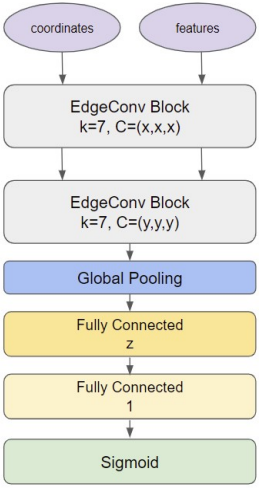
Part 2

Experiments

Content:

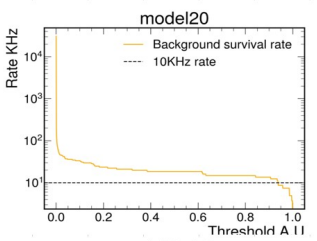
- With Jeronimo's help, we set up multi-testing standards (metrics) plot for training models.
- Now, we have training curves, confusion matrix, efficient rate curve and background rates plot.
- Train models with different parameters refer to different neural networks architecture and plot metrics of each experiment.

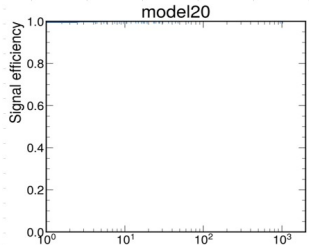
General Setup

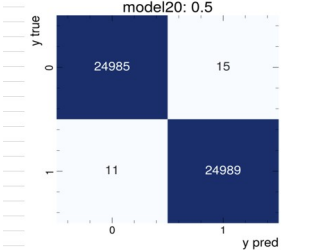


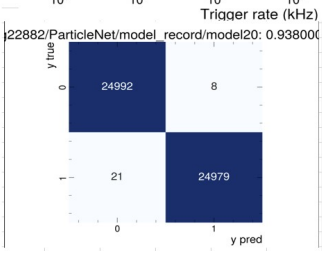
- On the left handside, the flow chart is the main architecture of our ParticleNetLite++. We can change parameters x and y to make defferent structures of 3-layer multi-layer perceptions and perform experiemnts.
- Fix k-nearest-neighbor parameter to 7 and batch size to 64, this is the best suitable size after multipule testing.
- The learning speed is so fast even using a very small learning rate. So we perform learning rate polynomialDecay Strategy when training models.
- x: neurons of 1^{st} EdgeConv Block 3-MLP.
- y: neurons of 2^{st} EdgeConv Block 3-MLP.
- z: neurons of dense layer

x=16, y=16, z=16



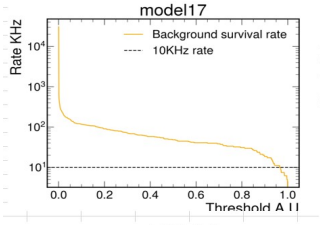


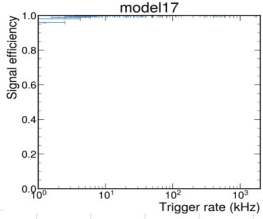


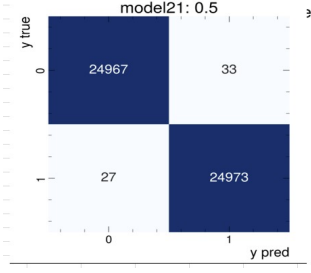


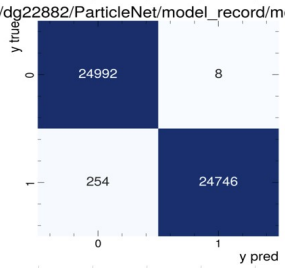
- Minimum threshold: 0.9380000000000001
- Accuracy: 0.9994
- Efficiency: 0.9992
- Test loss: 0.2953
- Volatility: 18.5140

x=8, y=16, z=16



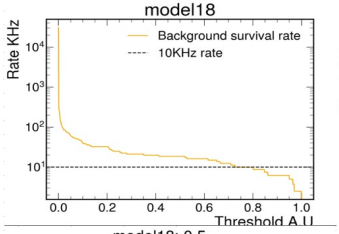


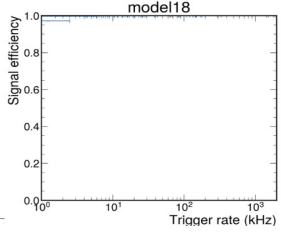


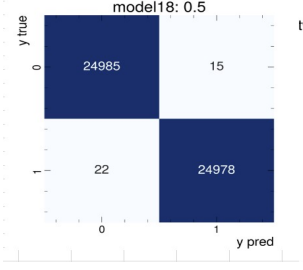


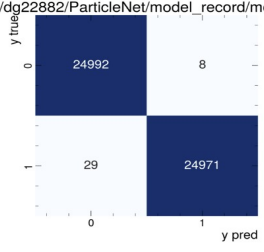
- Minimum threshold: 0.965
- Accuracy: 0.9989
- Efficiency: 0.9982
- Test loss: 0.3202
- Volatility: 17.5000

x=8, y=8, z=16



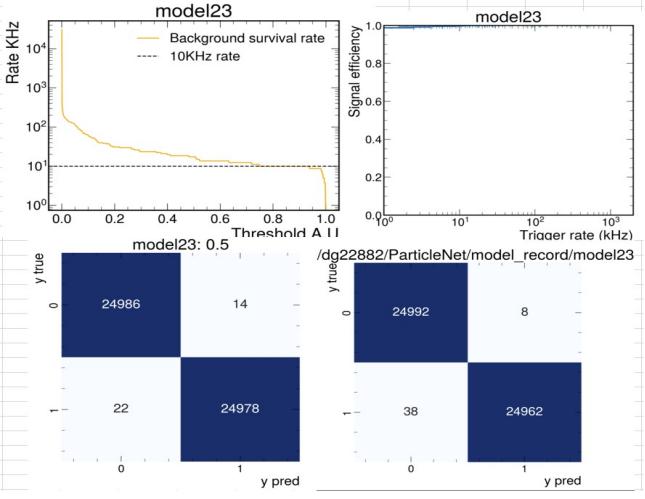






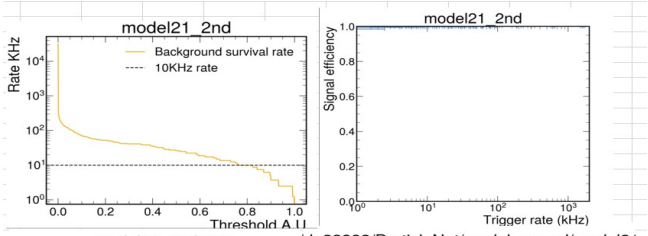
- Minimum threshold: 0.727
- Accuracy: 0.9993
- Efficiency: 0.9988
- Test loss: 0.1385
- Volatility: 8.1796

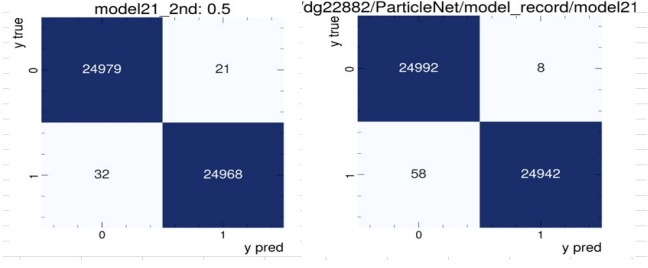
x=4, y=8, z=16



- Minimum threshold: 0.749
- Accuracy: 0.9991
- Efficiency: 0.9985
- Test loss: 0.1531
- Volatility: 1.0461

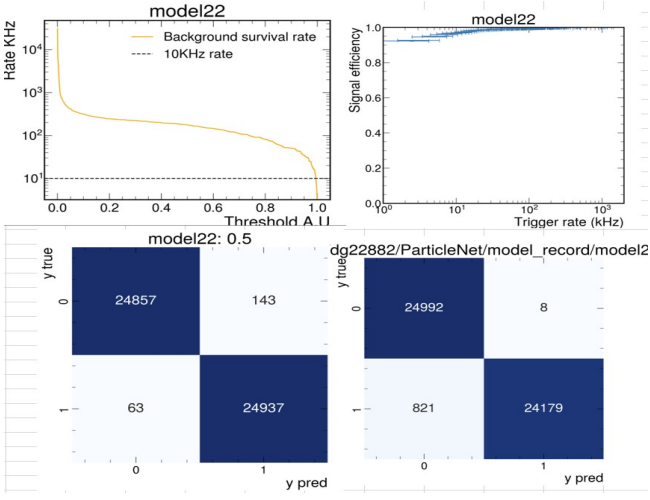
x=4, y=4, z=8





- Minimum threshold: 0.761
- Accuracy: 0.9987
- Efficiency: 0.9977
- Test loss: 0.1691
- Volatility: 6.3678

x=2, y=2, z=2

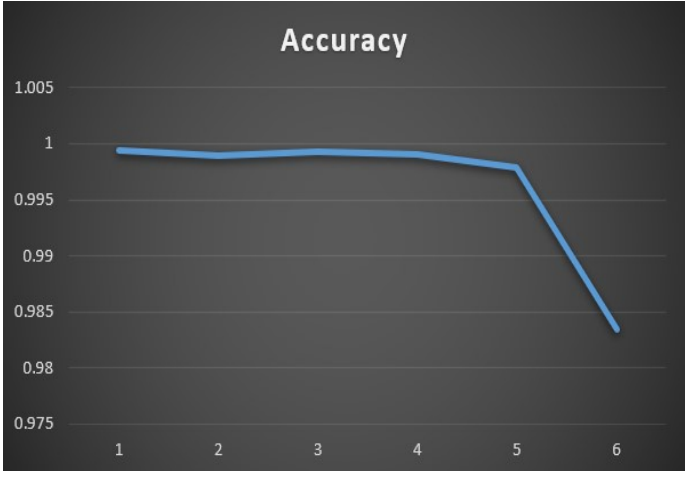


- Minimum threshold: 0.994
- Accuracy: 0.9834
- Efficiency: 0.9672
- Test loss: 0.3864
- Volatility: 0.1633

Parameter Size

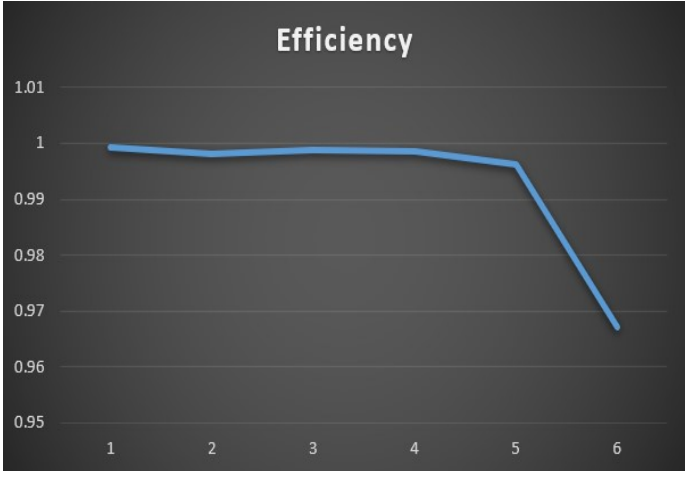
- 1: x=16, y=16, z=16
- 2: x=8, y=16, z=16
- 3: x=8, y=8, z=16
- 4: x=4, y=8, z=16
- 5: x=4, y=4, z=8
- 6: x=2, y=2, z=2

Accuracy



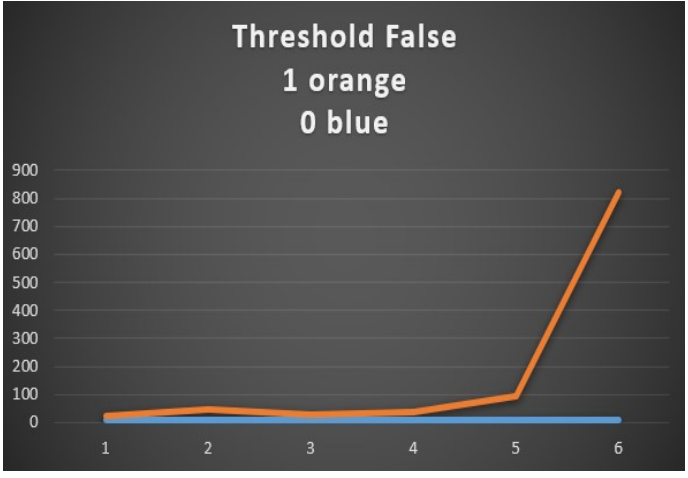
- 1: x=16, y=16, z=16
- 2: x=8, y=16, z=16
- 3: x=8, y=8, z=16
- 4: x=4, y=8, z=16
- 5: x=4, y=4, z=8
- 6: x=2, y=2, z=2

Efficiency



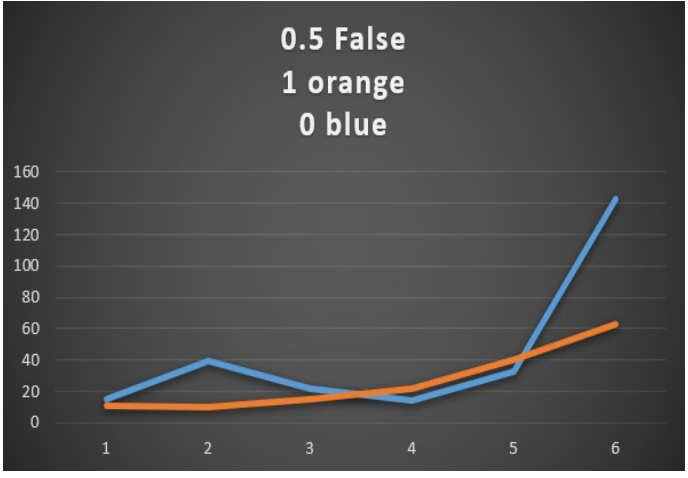
- 1: x=16, y=16, z=16
- 2: x=8, y=16, z=16
- 3: x=8, y=8, z=16
- 4: x=4, y=8, z=16
- 5: x=4, y=4, z=8
- 6: x=2, y=2, z=2

Threshold False Num



- 1: x=16, y=16, z=16
- 2: x=8, y=16, z=16
- 3: x=8, y=8, z=16
- 4: x=4, y=8, z=16
- 5: x=4, y=4, z=8
- 6: x=2, y=2, z=2

0.5 False Num



- 1: x=16, y=16, z=16
- 2: x=8, y=16, z=16
- 3: x=8, y=8, z=16
- 4: x=4, y=8, z=16
- 5: x=4, y=4, z=8
- 6: x=2, y=2, z=2

Summury

- The model still works very great in small parameter size. But performance will be harmed a little bit by reducing params (still very high).
- But I'm confused about this (whether it's a bug or Graph Neural networks is too powerful). So I need more different combination dataset to prove it.

Part 3

Next Weeks Plans

Content:

- Design fast transform algorithms for making dataset which each event only contains 100 particles based on most highest pT.
- Sort algorithms of making dataset and write more functions to make code shorter.
- Construct different dataset to verify this strange phenomenon (high accuracy with low params size).
- Perform experiments on cut-sized dataset. These experiments will be more realistic.