Deep Neural Network to Extract the Dimuon Properties

January 21, 2023

Introduction

- We use the reconstructed single track information of the Drell-Yan events to train the neural network.
- Input tensor features: charge, position at station 1 drift chambers, momentum at station 1, position at station 3, momentum at station 3.
- Target tensor features: dimuon vertex position, dimuon vertex momentum and dimuon mass.
- Data set was split to train: validate: test = 60: 20: 20.
- Our main goal is to train the neural network to extract the dimuon vertex information.

Neural Network Architecture

- Feed-forward deep neural network witch contains 2 blocks. Classification block will try to identify the origin of the tracks and regression block will try to extract the dimuon features.
- Classification block;
  - Contain 2 hidden linear layers.
  - In the forward pass all the layers are activated by the ReLu activation function.
  - In the back propagation loss is calculated by CrossEntropyLoss.
- Regression block;
  - Contains 3 hidden linear layers.
  - In the forward pass all the layers are activated by the ReLu activation function.
  - In the back propagation loss is calculated by MSELoss.

- Classification block is trained with vertex z position (hot id) and regression block is dimuon features.
- We use the Adam optimizer in the back propagation step.
- Learning rate = 0.00008 and L2 penalty = 0.0001.
- Total loss is calculated;

  total loss = loss clas. + alpha * loss reg.

  alpha = 0.009 is a non trainable hyper parameter.
- We use the batch training to train the neural network with batch side = 64 for 200 epochs.
- Total trainable parameters = 10902 and training data size = 1519596. Rule of thumb training data size = 10* total trainable parameters.

```
dimuNet(
  (tagger): Tagger(
    (fc1): Linear(in_features=26, out_features=124, bias=True)
    (fc2): Linear(in_features=124, out_features=124, bias=True)
    (fc3): Linear(in_features=124, out_features=124, bias=True)
    (fc4): Linear(in_features=124, out_features=5, bias=True)
  )
  (regressor): Regressor(
    (fc1): Linear(in_features=31, out_features=248, bias=True)
    (fc2): Linear(in_features=248, out_features=248, bias=True)
    (fc3): Linear(in_features=248, out_features=248, bias=True)
    (fc4): Linear(in_features=248, out_features=248, bias=True)
    (fc5): Linear(in_features=248, out_features=7, bias=True)
  )
)
```

Loss Curves

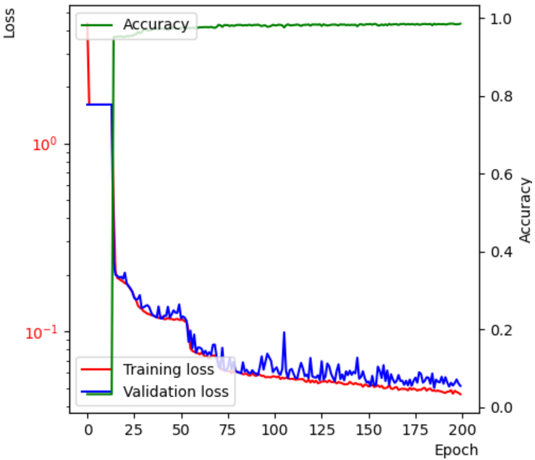

Figure 1: Classification loss for each epoch.

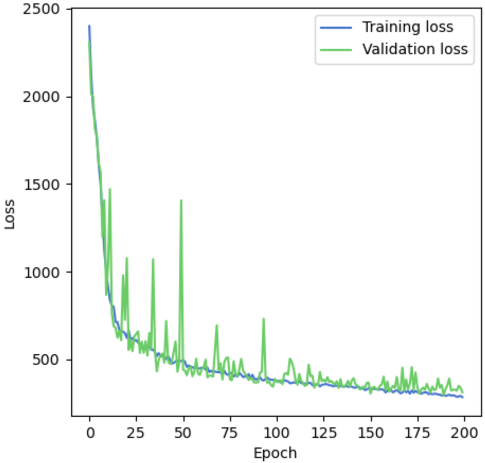

Figure 2: Regression loss for each epoch.

Classification

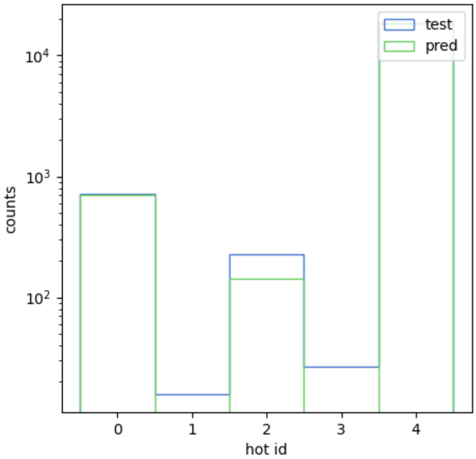

Figure 3: Prediction of the classification.

- Tracks are coming from colimeter(id = 0), target (id = 2) and beam dump (id = 4) are predicted well. But tracks are coming from air (id = 1 and 3) region has a bad prediction.

Predictions

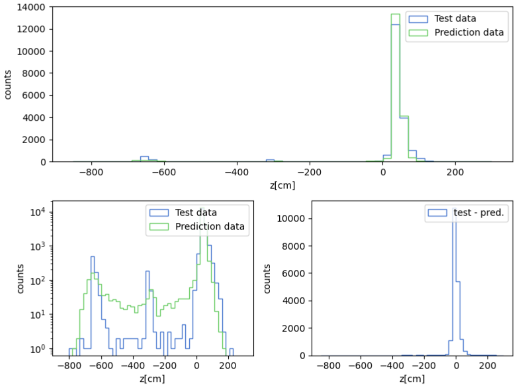

Figure 4: z vertex position.

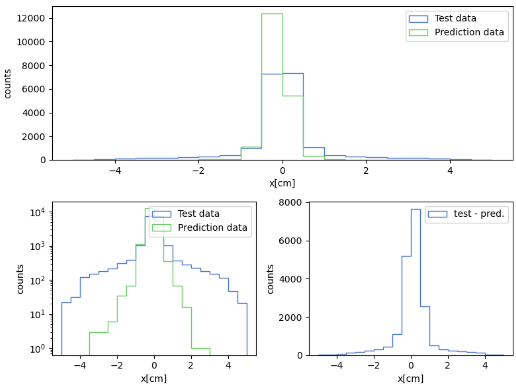

Figure 5: x vertex position.

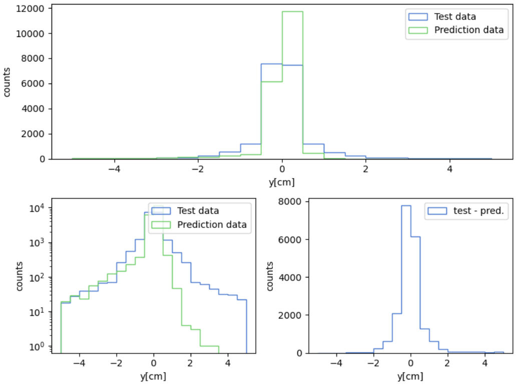

Figure 6: y vertex position.

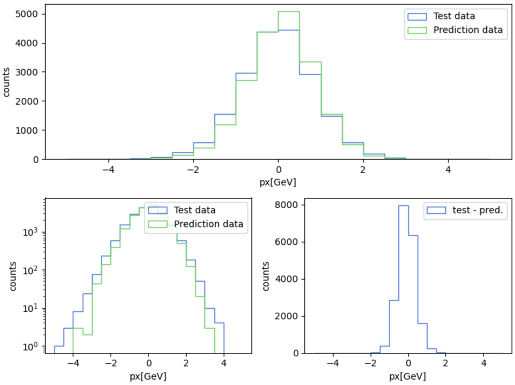

Figure 7: px at the vertex.

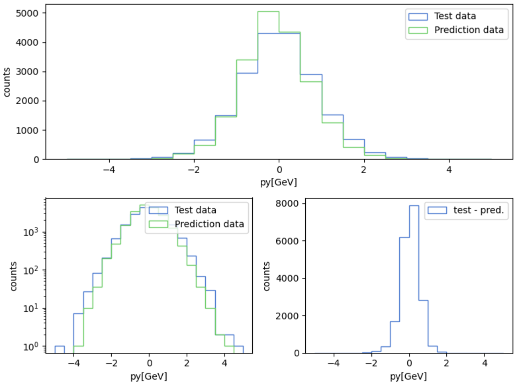

Figure 8: py vertex position.

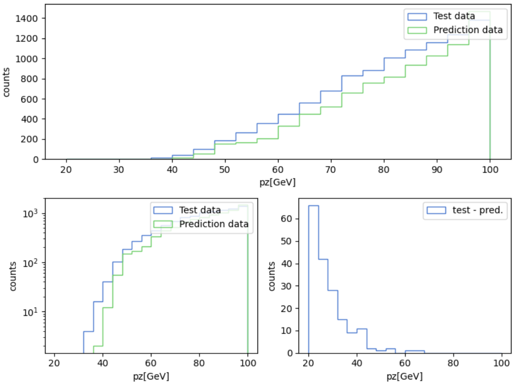

Figure 9: pz at the vertex.

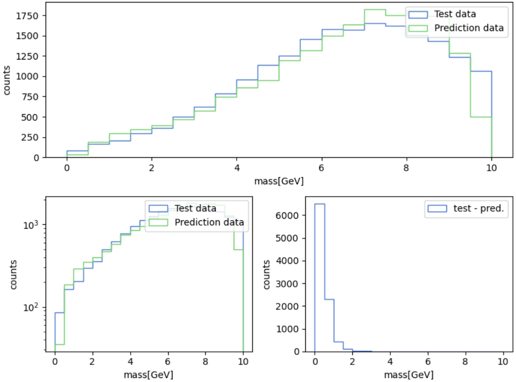

Figure 10: dimuon mass.

- Since tracks are unique we can use the constitutional neural network for the classification. But even with the input channel = 1, CNN fails the classification.
- Batch normalization and Dropout layers also reduce the accuracy of the results (some how ?)