# Physics Application of AI Jet Tagging Report

Neil Dotan

May 2024

## 1 Aim

The aim of this project is to make a deep learning based classifier for particle jets tagging, sensed by that Atlas detector at CERN. In this project, we will present 2 different classifiers:

1. classifier to separate QCD jets from either W/Z or top jets in a binary classification task
2. classifier to identify all three jet types in a multi-classifier

## 2 Introduction

A jet is a narrow cone of hadrons and other particles produced by the hadronization of quarks and gluons in a particle physics or heavy ion experiment.

The Atlas detector is a large cylinder around the LHC that can sense hadrons that hit it.

## 3 dataset description

In the project, we use two different datasets one that contain the individual constituencies, and one that contain some high level values, which can be calculated using from the constituents for each jet.

### 3.1 constituents dataset

our constituent dataset contain up to 20 constituents for each jet, each constituent contain a vector of size 4 that has the following properties:

- $p_T$ - the perpendicular momentum of the jet $[0, \infty]$
- $\eta$- boosted angle vertical angle $[0, \infty]$
- $\phi$ - circular angle $[-\pi, \pi]$
- $m$ - the rest mass $[0, \infty]$

Jets that have less constituents then 20 are masked with 4 vectors contain only zeros.

The dataset contains 3 different types of jets:

- QCD jets - 500k samples
- WZ jets - $\approx$ 746k samples
- TOP jets - 300k samples

notice that the amount of samples from each type are unbalance, this might make a bias problem for the more common classes.

as mentioned, jet is localized at specific area of the detector. additionally, the location which a jet is hit the detector is random as far as we know. considering these facts, we centered the $\eta$ and $\phi$ values as a pre-processing using the formula: $x_i = x_i - \bar{x}$

a useful way to present data is by an image (jets map) of the jets in the $\phi, \eta$ plane colorized by their energy (calculated using: $\epsilon_i = \frac{p_i * \cosh(\eta)}{\sum_{i=1}^{n} p_i * \cosh(\eta)}$

the main different between this 3 interactions is the how many jets we measure in the detector for a specific interaction. Theoretically, the QCD interaction yield 1 jets, the WZ interaction yield 2 and the TOP yield 3 as we can see in1:

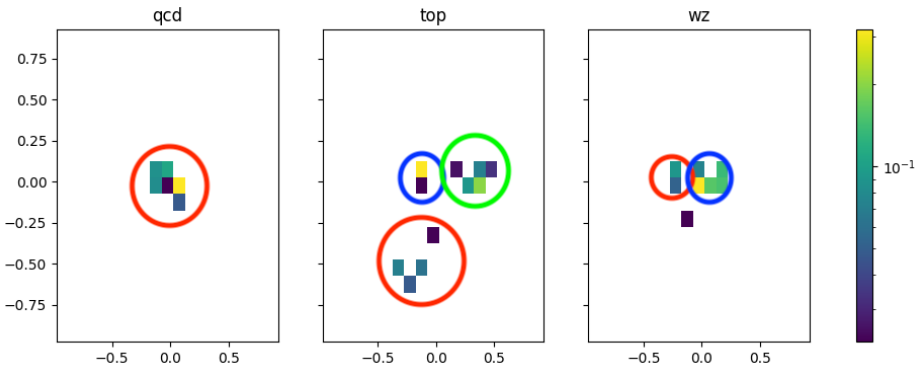


Figure 1: jets map of QCD, WZ and TOP interecation coloried by thier energies in a log scale. the x and y axis are normalized $\eta$ and $\phi$ axis. each jets is cirulated with different color

in these example the images is a pretty easy the be separated. the different jets can be identified easily. in the WZ jets we can see another region that has not been identified as a jet. this is because we know it should have only 2 regions and the other region is small and contain low energy measurement, so it probably a noise. but in other jets we can see a harder separation as possible to see at:

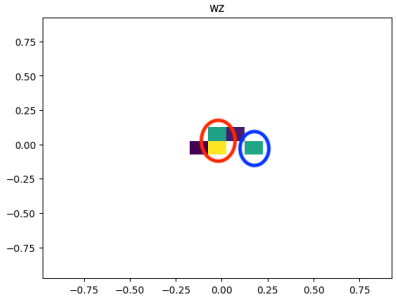


Figure 2: WZ jets map that more challenging to identify its interaction. for first locking, it is possible to misidentify it as a QCD jet, because it seems to have measured particle only in one area

## 3.2 high level dataset

The high level dataset contains information about the average values of the constituencies, and several classical algorithms which yield the probability of a jet to be a specific type.

for analyzing the data, we will check the histogram of the features as presented in the plot 3

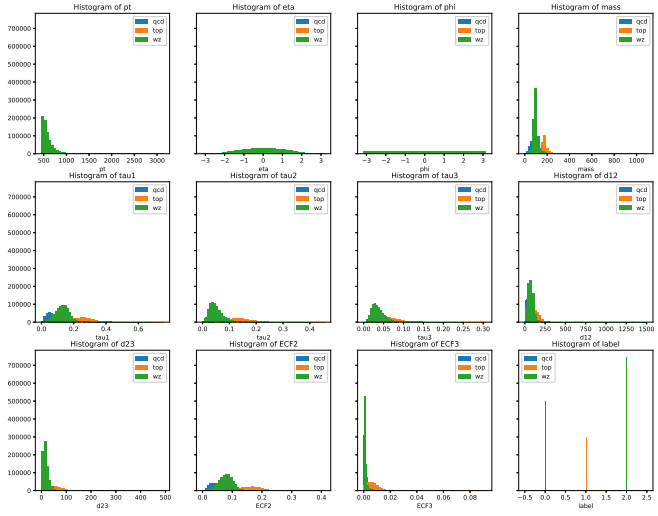


Figure 3: histogram of all the features of the high level dataset. the features: pt, eta, phi, mass are the average values of the individual constituents features. The other features extracted using classical algorithms on the raw data. we can see the unbalance between the different jets type. and that QCD and TOP can be probably separated pretty easily but separating from WZ probably be harder

# 4 method

first, we will bring a general overview about the architecture. As we can see from the "jets map", the jet's type is strongly depends on the correlation between the individual constituents, transformer are suitable to this task. The guiding principle in this work is the keep the network size pretty small for fast training, therefore, we are not expending our embedding space to be larger then the 4 vectors of the constituents. after the transformer layer we will merge the values from the "high level dataset". As seen in 3, some of the properties has pretty good separability, so it make sense to inject these into the model as well.

Detail sketch of the architecture is presented at 4 We will present here the model for multi-class classification. For the binary classification, The main structure is the same, the only different is the last layer, which change to a one dimension output with sigmoid activation function.

Our network has two different input layers, one for the constituents (marked with c) and one the level data (marked with h). Both c and h pass through a learnable normalization layer, this layer contain small amount of parameters, so its suit for our guiding principle of small networks. After the normalization, both c and h pass through dropout layer for regularization. After that, c is pass through a transformer encoder, and a global average pooling for flatten it and make it possible to concatenate it with h. After this step, the combined c/h data pass through a hidden MLP layer with relu activation function. The last part of the network is an MLP with three outputs and Softmax activation - standard way to handle three labels classification problem.

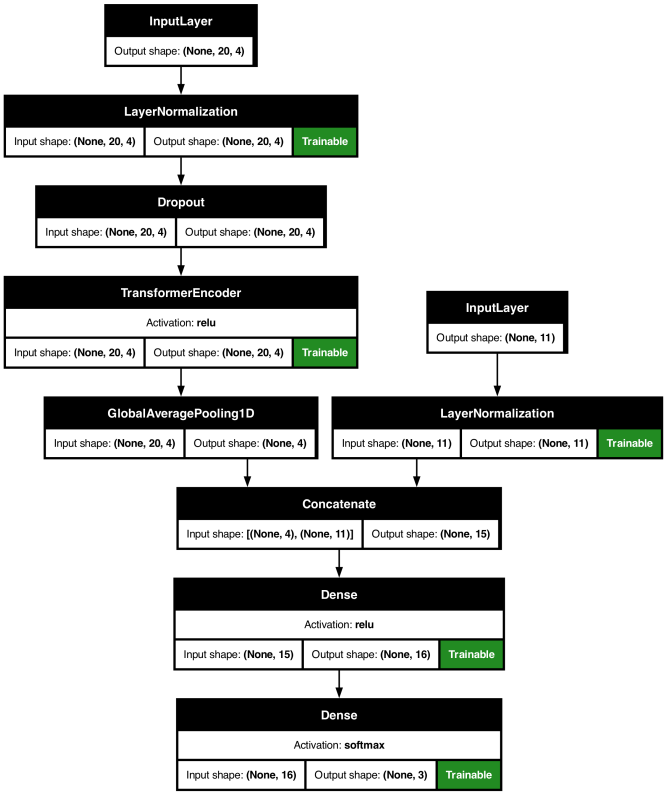


Figure 4: The model for multi-class classification. The are two input layers - one for the constituents and one for the high level data. Each of the input layers pass to a normalization and dropout layers. the constituents layer passed to a transformer encoder and after it to a global average to flatten the layer. After it, we concatenate the data from the different output and pass it to a dense network. The total number of trainable parameters is 581

For the loss function, we have picket one of the popular function for each task. Binary Cross-entropy for the binary classification task and sparse categorical Cross-entropy for the multi-class classification. For the optimizer, we have used AdamW also a standard choice. All the hyper parameters can be seen in table 1

| | |
|---|---|
| weight decay | 0.0001 |
| dropout rate | 0.1 |
| learning rate | 0.001 |
| norm epsilon | 10e-5 |
| number of attention layers | 1 |
| number of attention heads | 1 |
| transformer MLP layer dimension | 16 |
| MLP layer after data combination dimension | 16 |

Table 1: the hyper parameters that we have used for training the model

# 5 Results

## 5.1 Multi-class Classification

The results for the multi-class Classification are present in **??**. as we can see, the network stop to learn after about 25 epochs. This is probably related to the small number of network parameters (581) comparing to the size of the training data ($\approx 1.5M$). We can also see fluctuations in both of the training data and the validation data. This might indicate that our learning rate is to high.

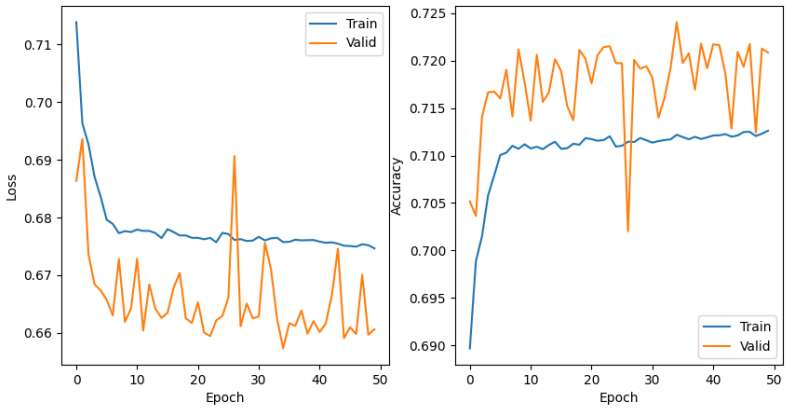


Figure 5: The results for the multi-class classification. The graphs shows the loss (left) and the accuracy (right) of the model over 50 epochs. The blue graph represent the training data and the orange the validation data.

the confusion matrix can is presented in figure 6. as we can see, The performance are pretty decent and even the accuracy is less then 80% even for the best separable case (TOP from QCD or WZ)

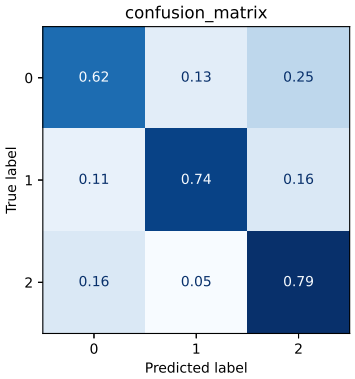


Figure 6: Confusion matrix for multi-class classification. The representation of the classes are: 0 - QCD, 1 - W/Z, 2 - TOP

## 5.2 Binary Classification

The binary classification task was to separate QCD jets from either W/Z or top jets. We expect to better results for this task, Because the network does need to identify between the W/Z and the top jets. Even though, a problem that can be introduce for this network is the high imbalance between the classes, where for one class we have (QCD) 500k training samples and for the other one (W/Z + top) we have $\approx$ 1M samples.

figures 7, 8 presents the lost and the accuracy of the binary models. the plots are the same as the plot of the multi-class classification, so we wont go into detail about them again. The test accuracy is 78%

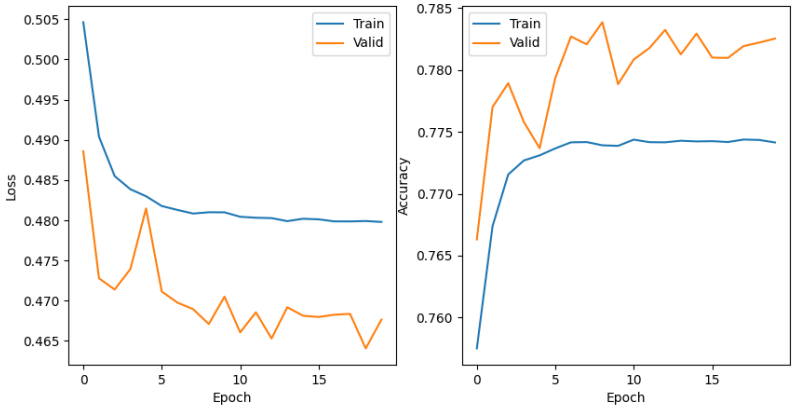


Figure 7: The results for the multi-class classification. The graphs shows the loss (left) and the accuracy (right) of the model over 20 epochs, as we saw that the networks stops to learn after that amount of epochs. The blue graph represent the training data and the orange the validation data. The test accuracy is 78%

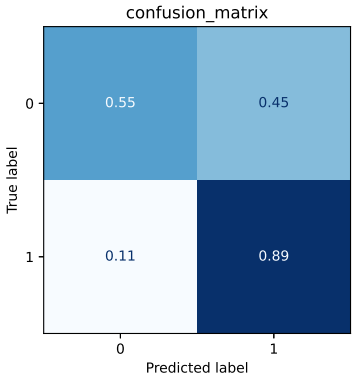


Figure 8: Confusion matrix for multi-class classification. The representation of the classes are: 0 - QCD, 1 - W/Z or TOP

we can see that as expected, the binary classification achieve higher performance respect to the accuracy. There is large amount of false The confusion matrix indicate that the model have a bais to the W/Z or TOP.

# 6 Conclusions

This project set out to develop deep learning classifiers for jet tagging in particle physics, leveraging data from the ATLAS detector at CERN. Two classifier models were constructed: a binary classifier to distinguish QCD jets from W/Z and top jets, and a multi-class classifier to identify all three types of jets.

A primary objective of this project was to achieve effective jet tagging using a small and computationally efficient network. The dataset used comprised both constituent-level and high-level features, carefully pre-processed to center the angular measurements and handle class imbalance. The models employed a small transformer architecture combined with high-level features, aiming to balance performance with computational efficiency.

The results was encouraging and show that even with a small transformer and MLP (for the high level data) we can achieve decent results.

Future work could focus on several areas to improve performance:

- increase the network size by all aspects - the transformer for the constituents and the MLP for the high level data
- applying PCA on the high level data - the high level data contains some features that are probably does not relevant for the classification (for example, $\eta$)