Sara Updates

Introduction + My Tasks

Hi, I'm Sara - I just finished UG Y2 at the University of Manchester, and will be working with Isobel for the next few weeks to work on a CNN to determine whether a reconstructed particle's connection pathways in LAr truly belong to the same particle. This is so that we can improve shower identifications, and catch any false reconstructions.

DUNE is an experiment in Fermilab (Illinois, USA) with its far detector 1,300km away in the Sanford Underground Research Facility in South Dakota. DUNE will investigate the neutrino oscillations, CP violation in Neutrinos, and probe the neutrinos produced in supernovae. A key factor is the Liquid Argon Time Projection Chamber (LArTPC), which is the medium used to detect particles and view their paths. When a neutrino interacts with the LAr, a shower of secondary particles is produced showing complex pathways of EM showers. These contain high energy electrons/photons which further produce particles, making it difficult to distinguish between hadronic (proton/neutron) showers. When we require identification and quantification of the charged particles, we must reconstruct the path of the particle. Due to the complex nature of this, we often have to distinguish between genuine and false reconstructions due to the noise, imperfections and ambiguities of the data.

To improve this reconstruction, I will be using a Convolutional Neural Network (CNN) to improve the identification of particle showers. We will be determining whether the reconstructed particle's connection pathway truly belong to the same particle - i.e., ensuring the segments of the detected trajectory (through the LAr) actually belongs to the particle of interest. CNNs are good for this task due to its ability to learn complex patterns and characteristics of the particle trajectories and showers. We focus on distinguishing the EM showers from other non-shower events, and catch false positives and misclassifications to aid neutrino analysis.

12 June 2025

Logistically, I started with

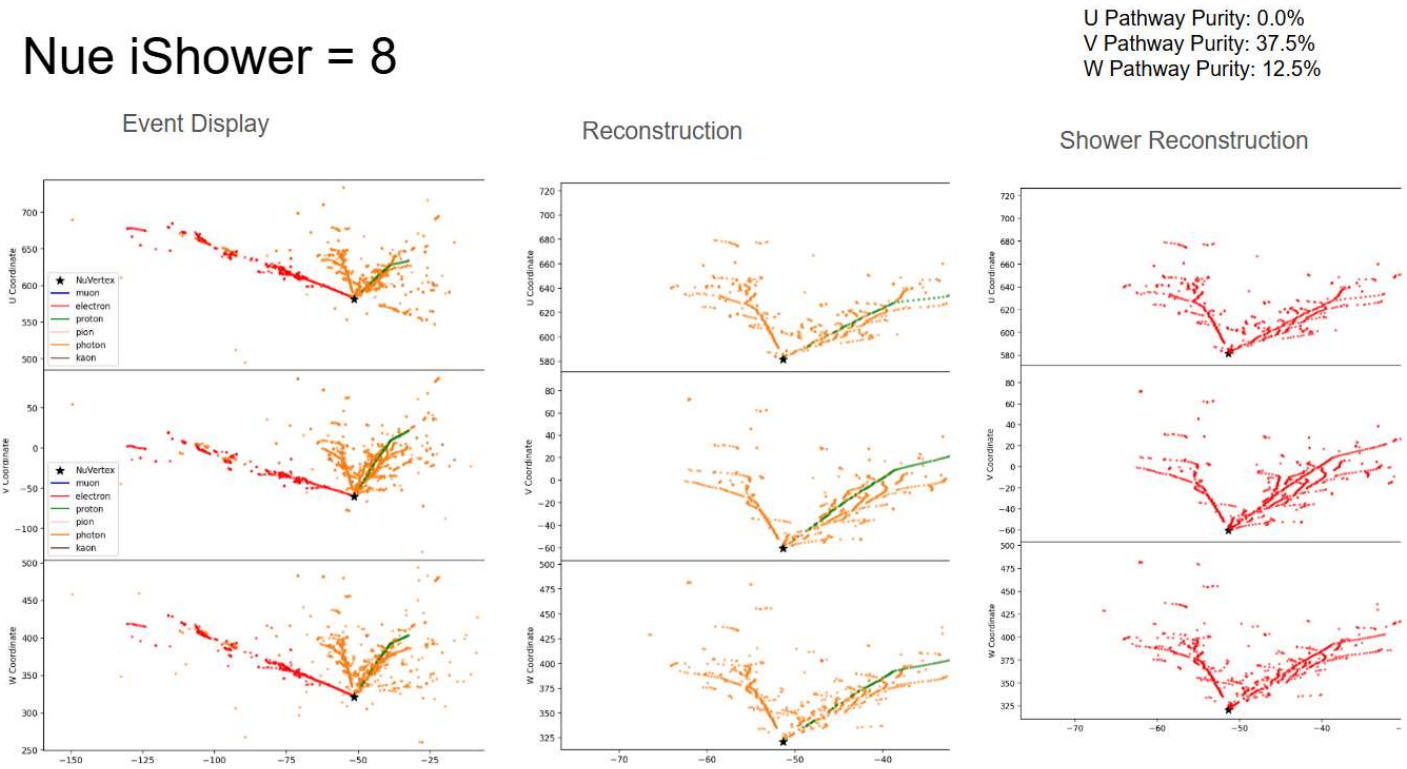
- Reading through project briefing / Isobel's PhD
- Health and Safety Induction
- Used Powershell (for the first time!) to obtain the ROOT files which contain interactions

Then, I began hand scanning through neutrino interactions to gain an intuition for what the issue is that we are trying to solve.

We need to hunt down for cases we might've missed due the reconstruction.

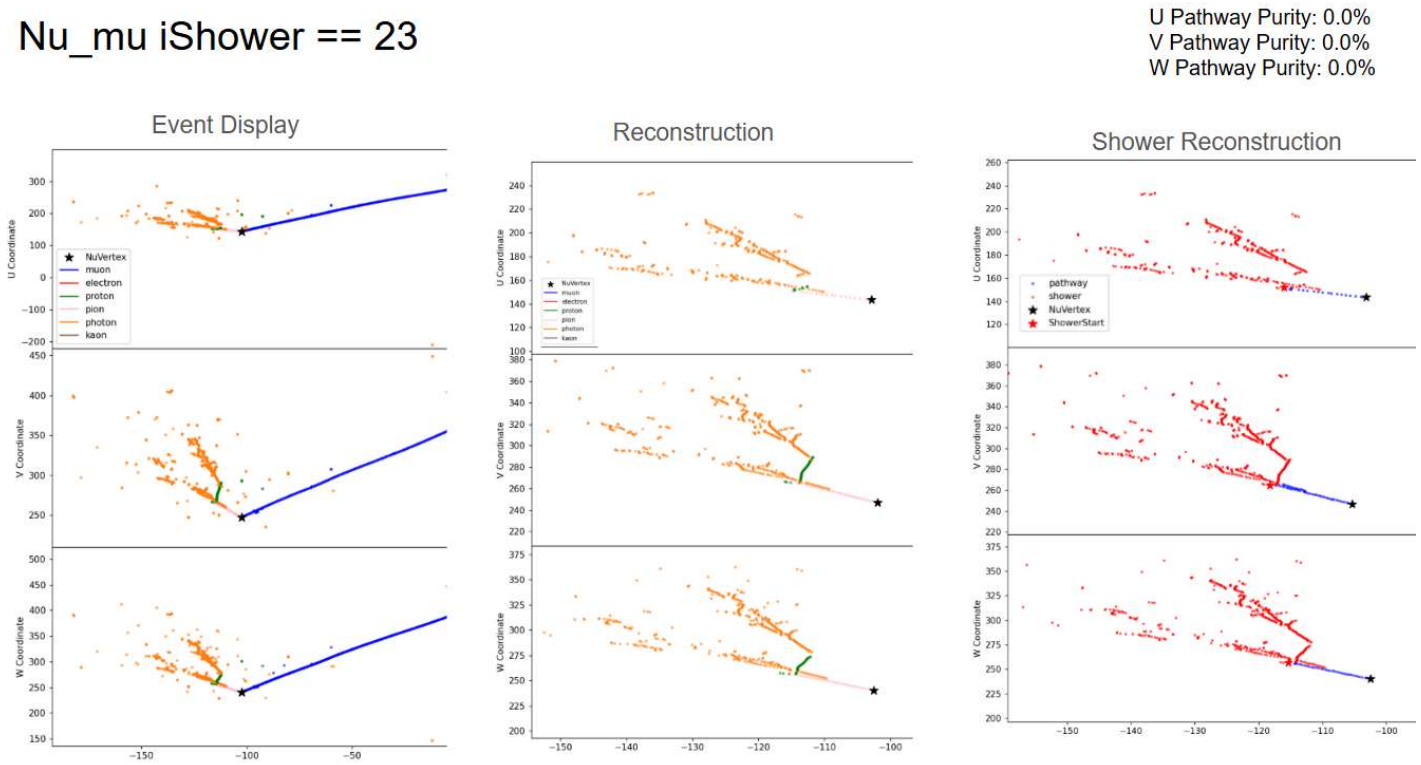
Pathway	Purity	Action	Purpose in ML
Not reconstructed to be a part of the shower	Low (path doesn't belong to the shower)	Do nothing	What non matching pathways look like
Reconstructed to be a part of the shower	Low	Remove	Bad merge example, learn to reject
Not reconstructed to be a part of the shower	High (path belongs to the shower)	Add	What a missed true connection looks like
Reconstructed to be a part of the shower	High	Do nothing	Correct match

False False Case Example - Nue iShower = 8



Photon shower reconstruction was unsuccessful because the photon showers got merged.

False Case Example - Numu iShower = 23

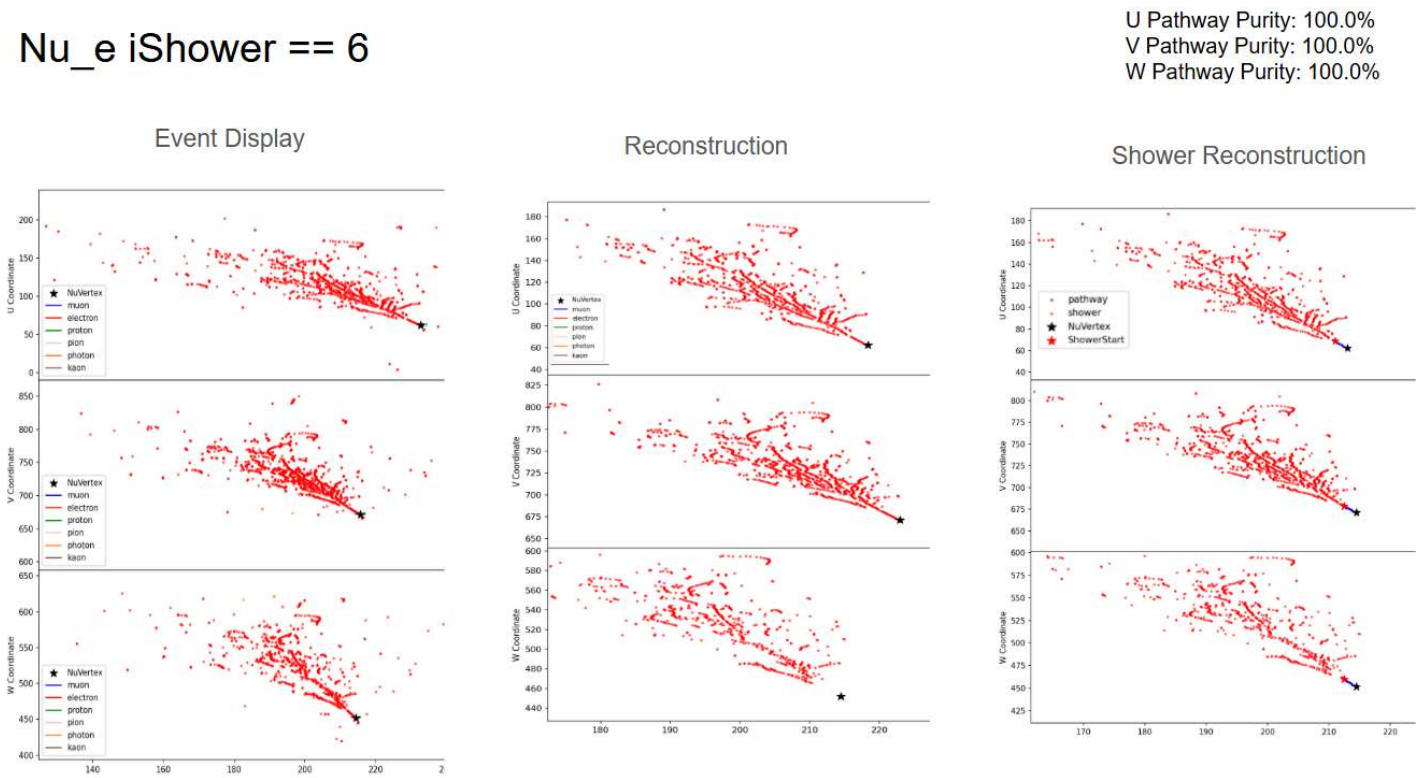


This is a case of the topology looking correct, but the pathway does not belong in the shower.

- Correct reconstruction of the photons
- Connection pathway lining up to the start of the shower
- Truth label says none of the pathway aligns with the shower

So this is a false case.

True Case Example - Nue iShower = 6



At every step the images look similar, indicating successful reconstruction.

What's next?

We can train an algorithm by making it observe the shower reconstruction, and catch examples that it would currently miss.

19 June 2025

This Week's Work:

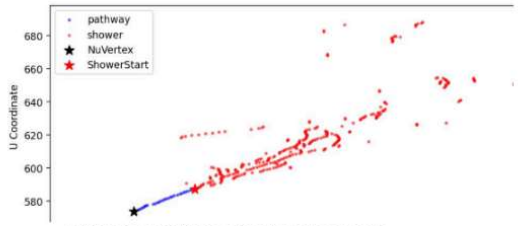
At the end of last week, we began making images to train and test our neural network. The goal was to use the shower reconstruction view to generate images captures the pathway and the start, which will be later used to train a CNN. This was a complicated task because we had to pick a 3D distance and project that into each of the 2D views (U,V,W coordinates), while ensuring that the size of the image was consistent.

The first step was to build a programme that takes the shower reconstruction view, centre it to the start of the shower, then bin the grids with hits and express it in a matrix. This contained the information for whether the hit was a pathway or a shower, and expressed it in blue/red. To

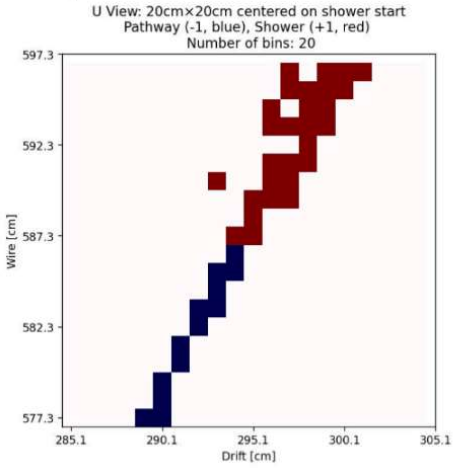
begin with, the region that we captured was arbitrary but consistent across the three views. The result of this is shown below.

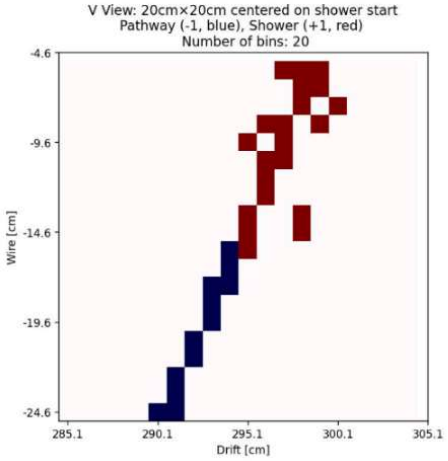
High Purity Nue iShower = 12

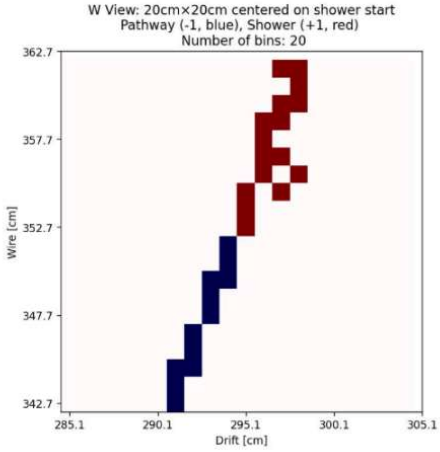
U Pathway Purity: 93.1%
V Pathway Purity: 94.59%
W Pathway Purity: 97.44%



Comments:
A good example with topological sense and high purity, all images look similar



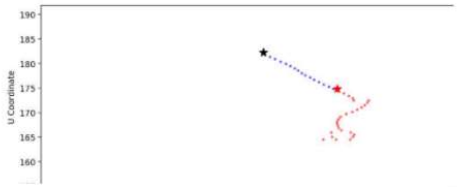


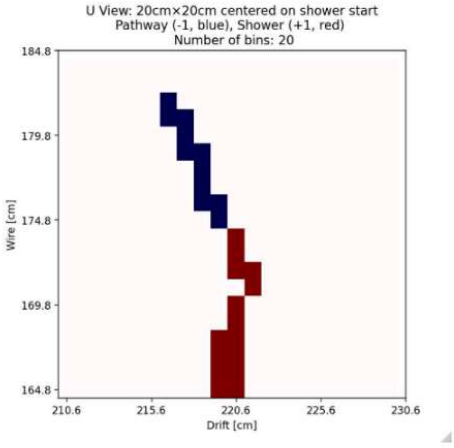


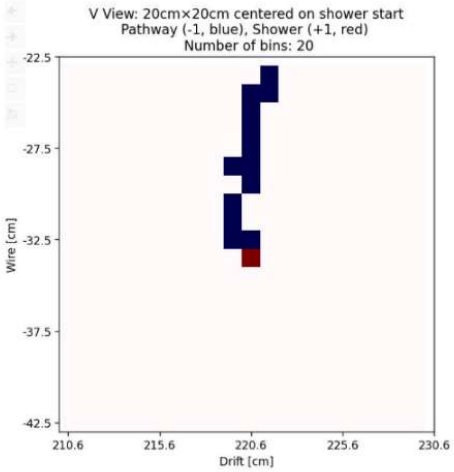
Medium Purity Nue iShower = 0

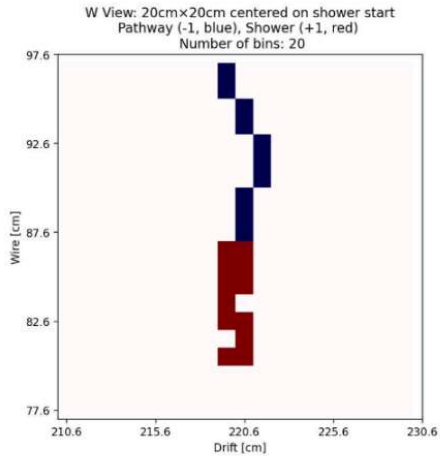
U Pathway Purity: 0.0%
V Pathway Purity: 79.41%
W Pathway Purity: 48.15%

Comments:



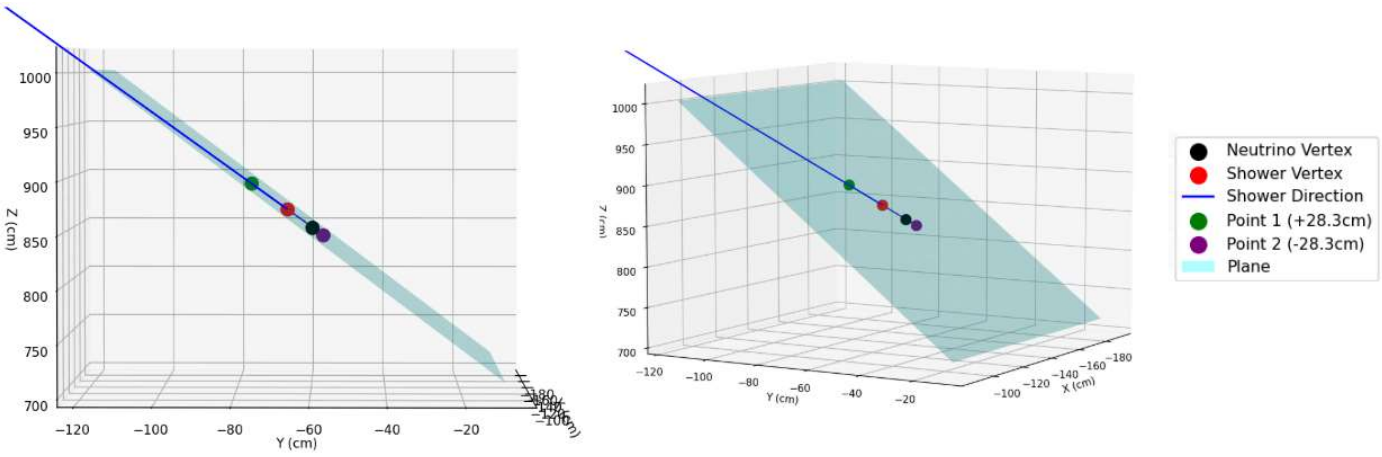




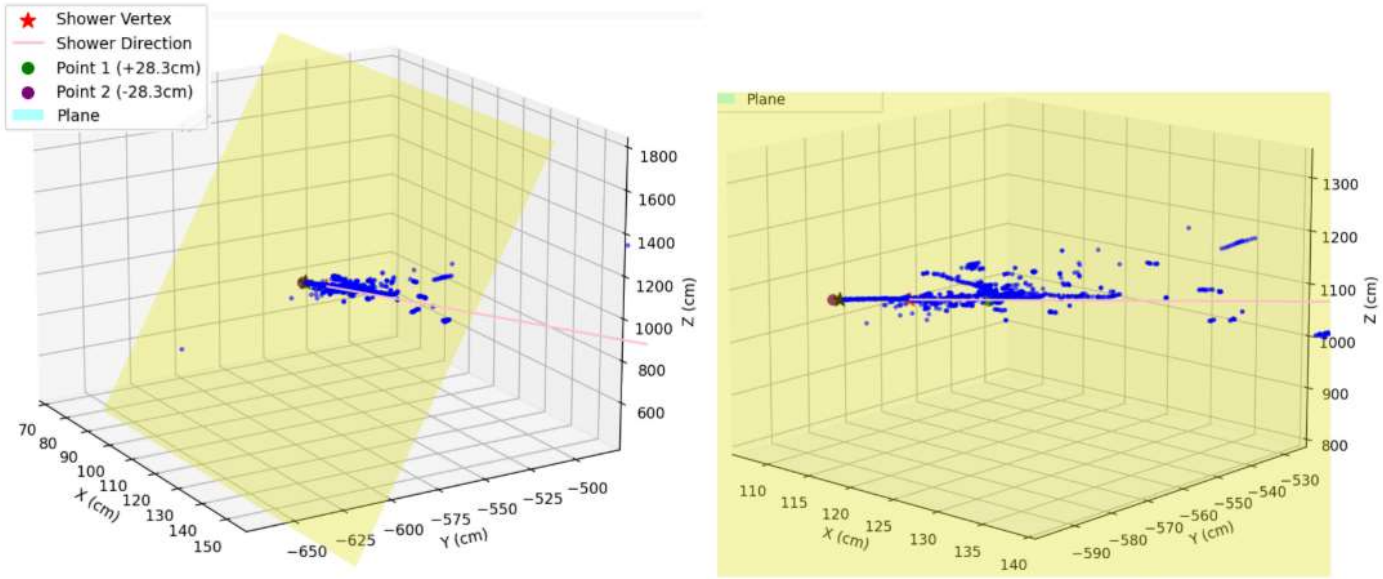


We can see that there is high agreement with the images for well reconstructed and high purity interactions, whereas lower purity results in three seemingly different images.

The next step was to obtain distances based on the 3D neutrino interaction. This was done by first forming a vector from the neutrino vertex to the shower vertex, and building a plane that is parallel to it. Then, two points were selected on that plane - one that is 28.3cm away on one side, and 28.3cm on the other (labelled point 1 and 2). These were originally going to be the corners of a square with lengths of 40cm. The visualisation of this is shown below.

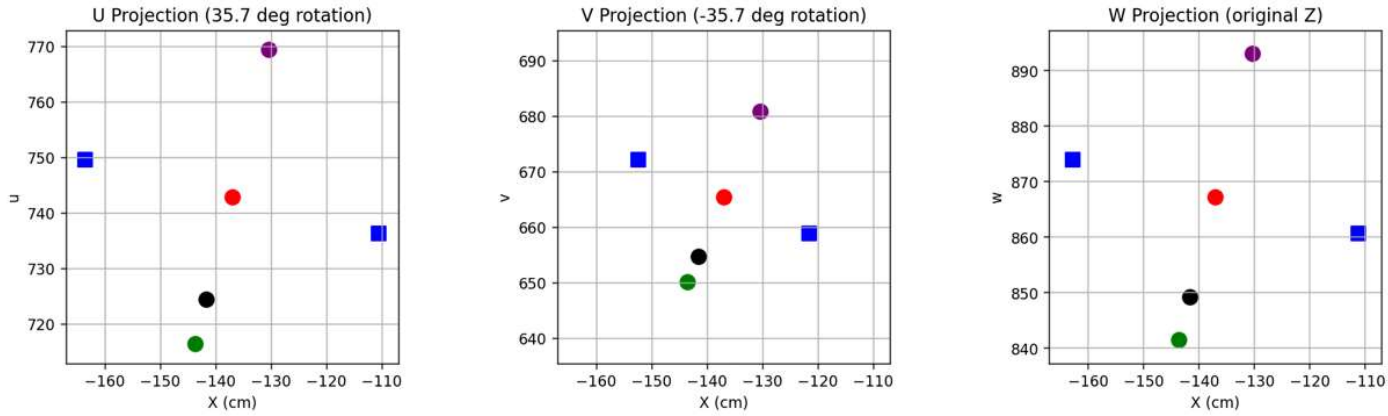


(Isobel also made a 3D visualisation of the shower, and I added the plane + vector onto it, shown below)

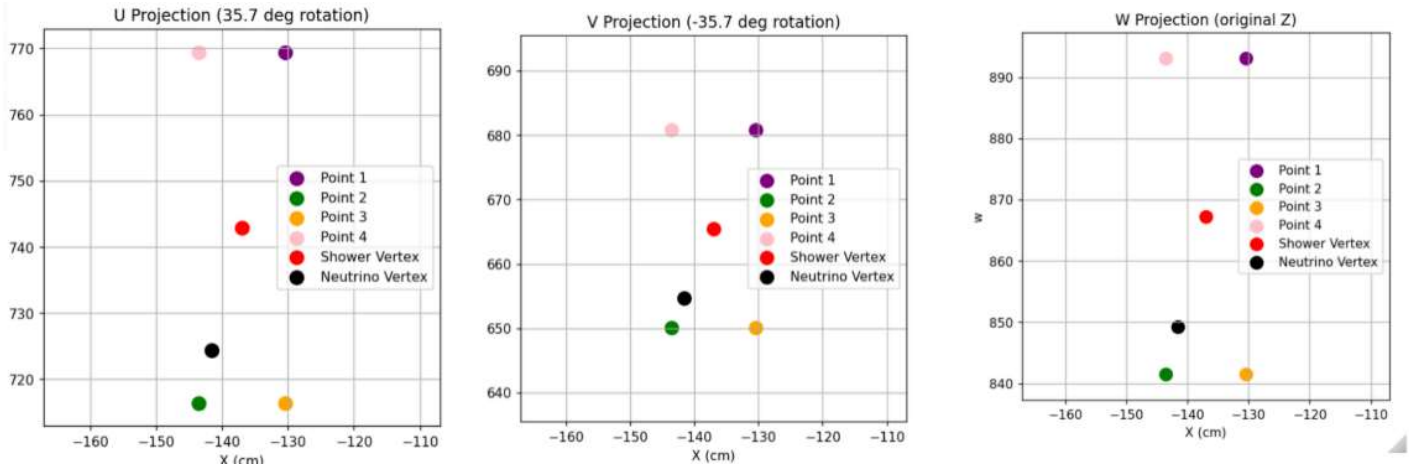


From here, the cartesian coordinates of Point 1 and 2 were projected into 2D for each of the U,V, and W views, using equations that account for the angles that they are tilted by. Initially, I generated Point 3 and 4 (in blue) in 2D to form a square, but Isobel realised that the drift coordinates (x coordinates in the plots below) were different, which would mean that the area

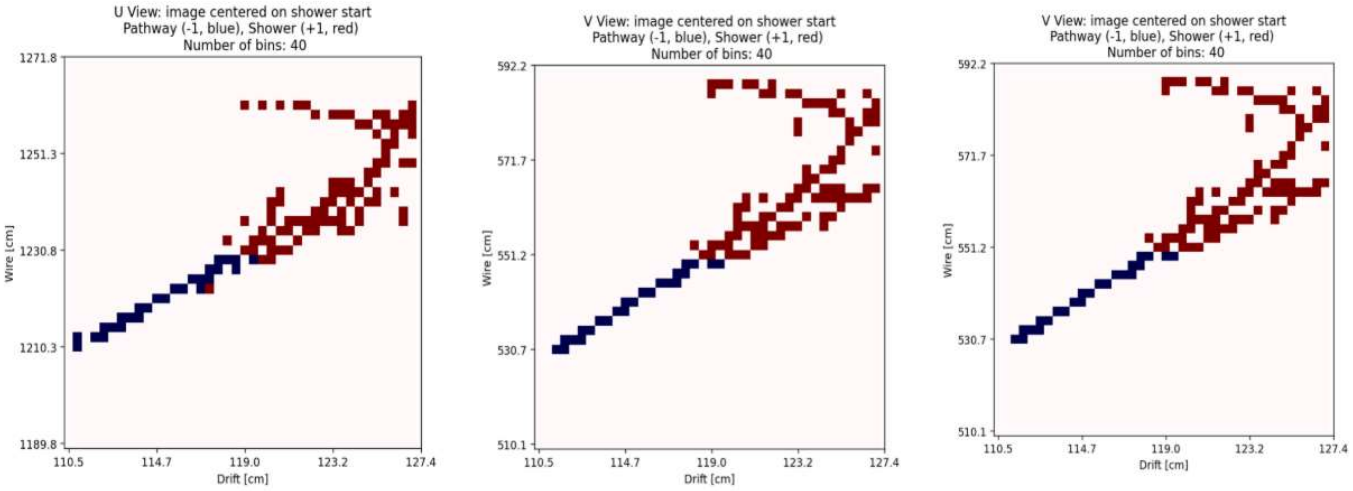
of the square, and subsequently the image size would be different. This was not suitable for training a neural network.



So, we simplified this by using the point 1 and 2 coordinates to complete a rectange, allowing the consistency of the drift coordinates while the U, V, W axis was not bound to a particular length. This difference in area was justified because the way that this region would be binned would be consistent, so the image that would be generated would be of the same area. The modification gave me the coordinates in (x, u, v, w) for the four corners of the rectange and in each view.



With those points, I was able to figure out the distances of interest for our image, and then set the length for the binning programme. This should now be consistent!



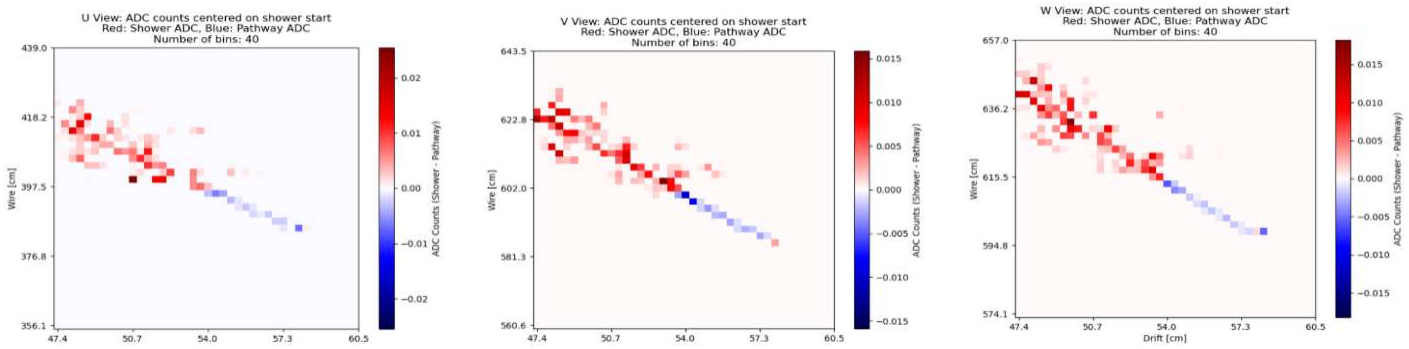
What's next?

We can further add the information about the charge of the particle (analogue digital count?) to the bin image assigning shades of colours. This would generate images of the same size across each coordinate system, which would be useful to train the CNN.

26 June 2025

This Week's Work:

On Friday, we finished creating images (which correspond to 2D arrays) to feed into the CNN. Since then, Isobel translated my python code into C++ and fixed some conceptual issues, allowing it to run with all of the shower files. Here is the result of the visualisation. The change since last week is that the colour range expresses the charge count.



On Monday, I worked through a ML tutorial and began constructing the framework for our CNN. Once the images were ready, I used the U-Coordinate images as the input and created truth labels based on the purity rating. Initially, the layers chosen were the same as that for

MNIST (handwritten letter recognition used in the tutorial), so it was not fully tailored to this purpose. I made it a point to keep the data loading, model building and training codes into a separate python file to keep the jupyter notebook clean.

```
Total params: 166,531 (650.51 KB)

Trainable params: 166,531 (650.51 KB)

Non-trainable params: 0 (0.00 B)

Epoch 1/10
214/214 ━━━━━━━━━━━━━━━━━━━━ 6s 20ms/step - accuracy: 0.5093 - loss: 0.9846 - val_accuracy: 0.4889 - val_loss: 0.8427
Epoch 2/10
214/214 ━━━━━━━━━━━━━━━━━━━━ 4s 19ms/step - accuracy: 0.4940 - loss: 0.8398 - val_accuracy: 0.5029 - val_loss: 0.8423
Epoch 3/10
214/214 ━━━━━━━━━━━━━━━━━━━━ 5s 21ms/step - accuracy: 0.5065 - loss: 0.8273 - val_accuracy: 0.4950 - val_loss: 0.8390
Epoch 4/10
214/214 ━━━━━━━━━━━━━━━━━━━━ 4s 19ms/step - accuracy: 0.5253 - loss: 0.8295 - val_accuracy: 0.4880 - val_loss: 0.8304
Epoch 5/10
214/214 ━━━━━━━━━━━━━━━━━━━━ 4s 20ms/step - accuracy: 0.5956 - loss: 0.8185 - val_accuracy: 0.6692 - val_loss: 0.7916
Epoch 6/10
214/214 ━━━━━━━━━━━━━━━━━━━━ 5s 21ms/step - accuracy: 0.6506 - loss: 0.7799 - val_accuracy: 0.6718 - val_loss: 0.7511
Epoch 7/10
214/214 ━━━━━━━━━━━━━━━━━━━━ 4s 19ms/step - accuracy: 0.6664 - loss: 0.7498 - val_accuracy: 0.6748 - val_loss: 0.7394
Epoch 8/10
214/214 ━━━━━━━━━━━━━━━━━━━━ 4s 19ms/step - accuracy: 0.6680 - loss: 0.7405 - val_accuracy: 0.6710 - val_loss: 0.7386
Epoch 9/10
214/214 ━━━━━━━━━━━━━━━━━━━━ 4s 20ms/step - accuracy: 0.6752 - loss: 0.7290 - val_accuracy: 0.6739 - val_loss: 0.7327
Epoch 10/10
214/214 ━━━━━━━━━━━━━━━━━━━━ 4s 20ms/step - accuracy: 0.6695 - loss: 0.7314 - val_accuracy: 0.6733 - val_loss: 0.7321
```

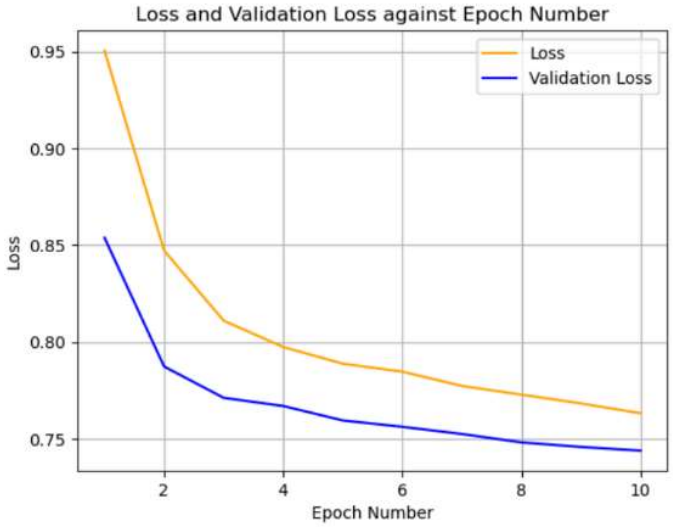
(I was very happy when this first ran since it was a milestone for this project.)

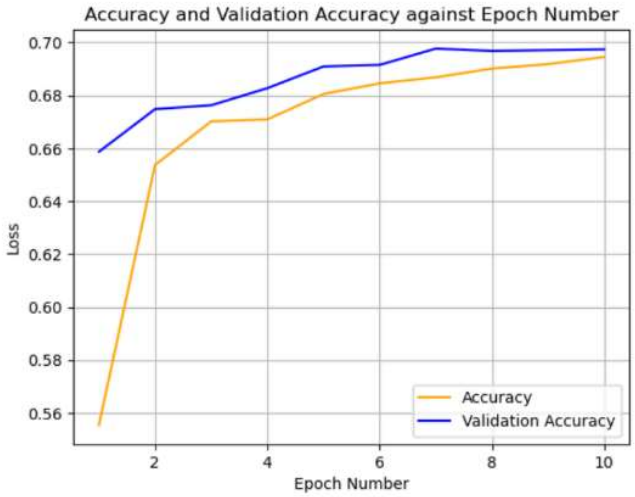
Once this ran smoothly, I began tailoring the CNN to our purposes. This was done through several customisations:

- I avoided onehot encoding (requirement to use an array e.g. [1,0,0], [0,1,0], [0,0,1] to express classes of 0, 1, 2) by using the sparse_categorical_crossentropy
- Normalised the input so that the matrix values ranged from 0 to 1
- Added a sparse layer and dropout layer

I then obtained useful metrics to understand the strengths and weaknesses of this model. This was done by considering the history, confusion matrix and the ROC curve:

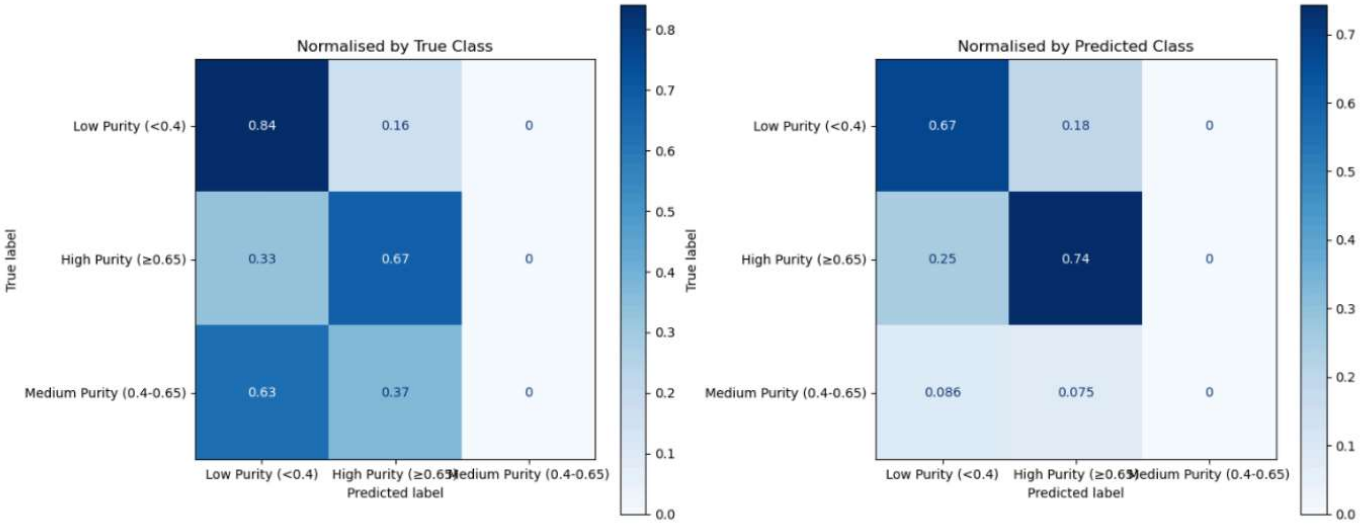
First, the history:





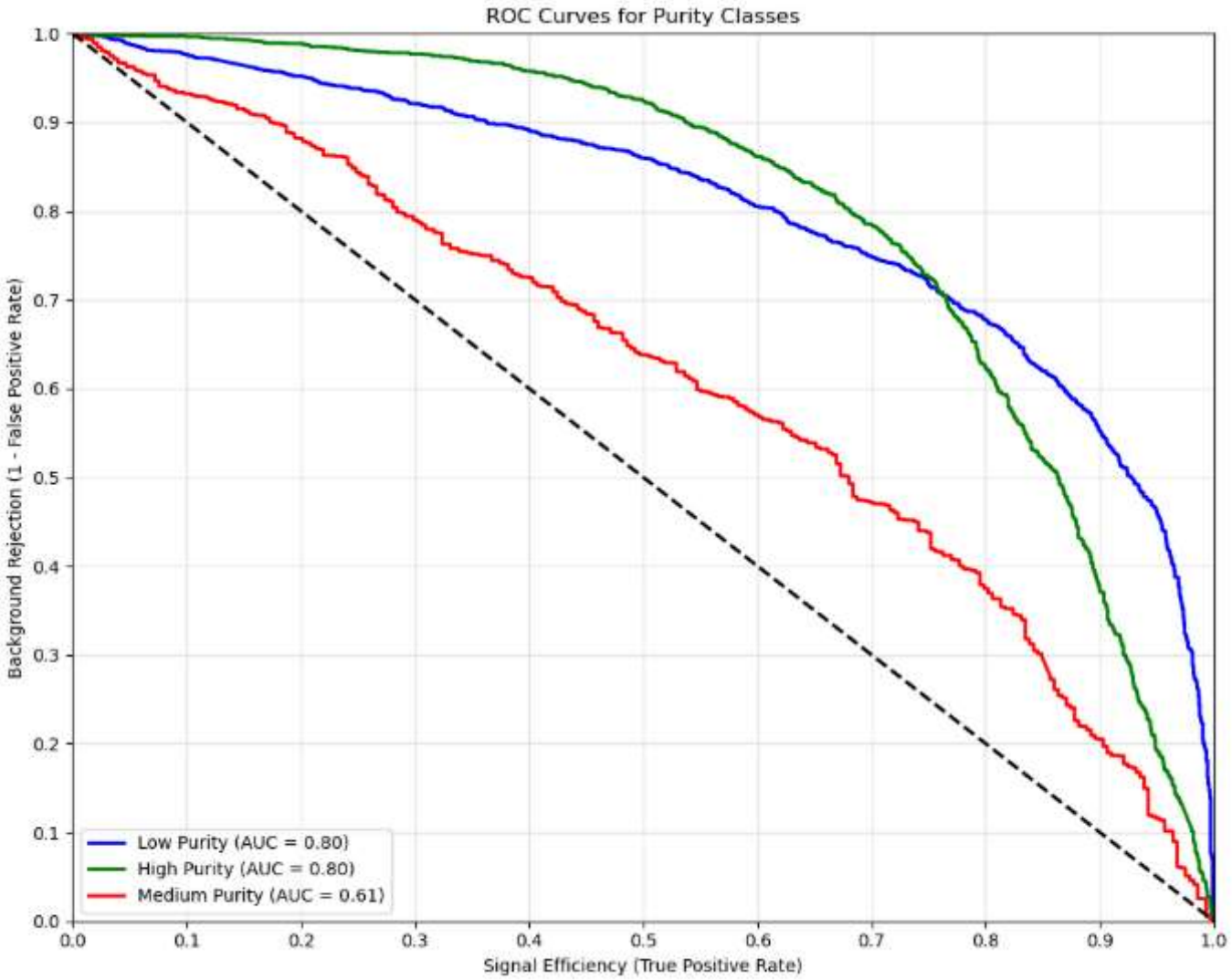
The loss decreasing, as well as the accuracy increasing shows that the neural network is learning. However, the rate at which it learns decreases and plateaus after 5-6 epochs, showing that there is room for improvement.

Secondly, we can consider the confusion matrix. This matrix summarises the performance of this model by comparing the predicted labels to true labels, allowing a useful measure of how well the model performs multi-class classification.



This points towards an issue with classifying the medium purity range (0.4 to 0.65).

Lastly, the ROC Curve shows how well a classifier distinguishes between classes. For this, we used a One-vs-Rest approach where we plot one ROC curve per class. We compare the true vs false positive rate, aiming for the area under the curve to be 1.0.



We can see that Low and High purity has a reasonably high area, whereas medium purity is much lower.

What's next?

We can diagnose the problems with the medium purity, possibly by adding weights. Also, I will run this with the full dataset (instead of around 3000) to benefit from the increased sample size.

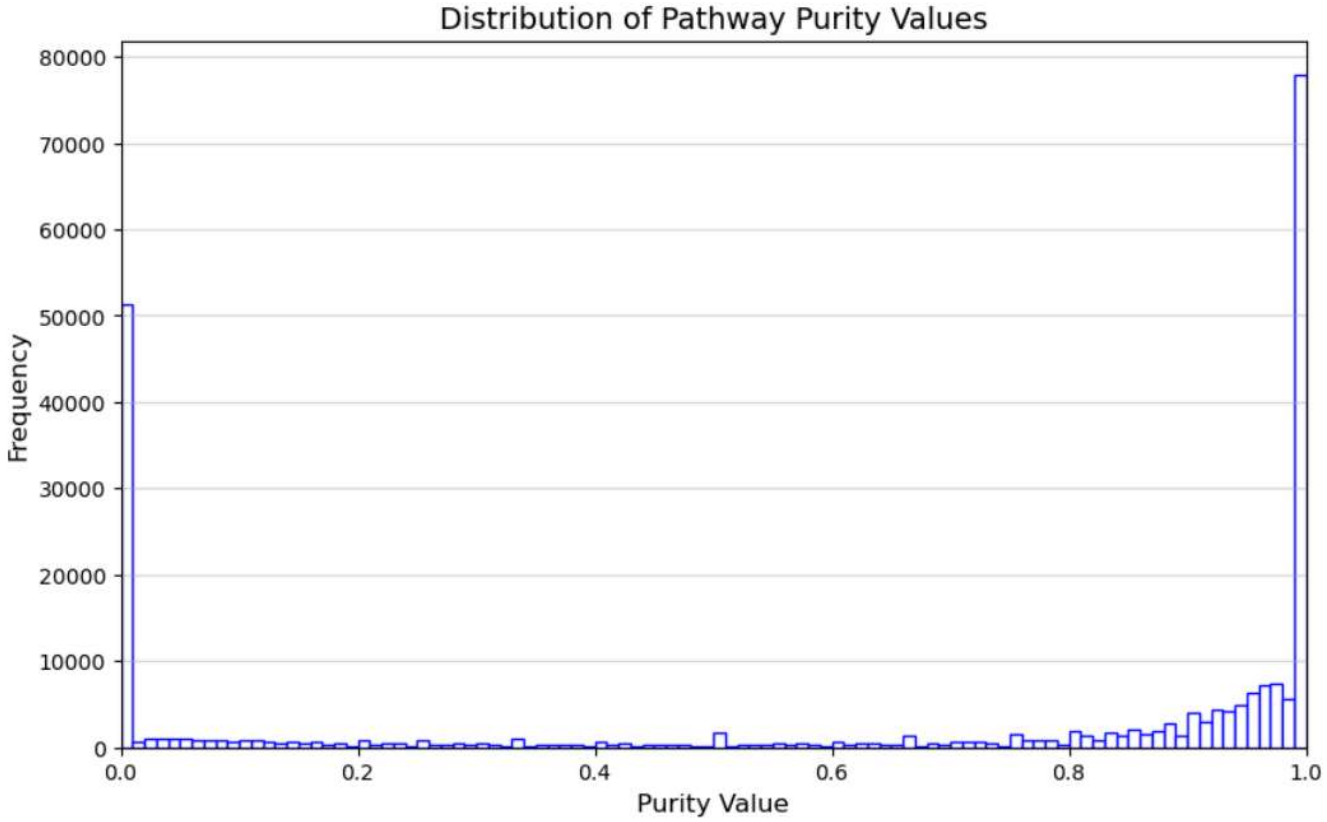
3 July 2025

This week's work:

We began using the full dataset which has about 14000 interactions. As predicted last week, this made it very difficult to run the code, causing a significant amount of delays. On top of this, my anaconda enviornment was apparently extremely messy, causing some significant setbacks, including the need to commute in almost everyday to use the linux machines at the university.

The goal of this week was to train the algorithm again with weighting so that the medium purity class has a higher presence, experiment with different numbers of classes, and then build up the model to handle all the U, V, and W coordinates.

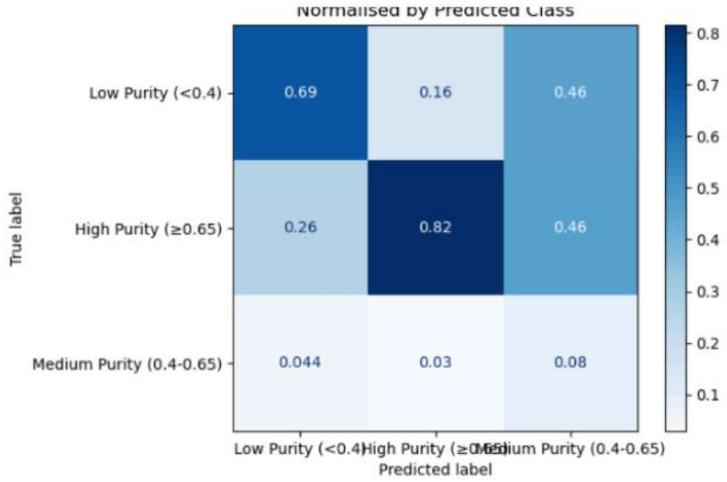
Before implementing weights, we wanted to ensure the quality of the data we also introduced a cut, where any interaction with a <50% shower purity was omitted from the training and testing sets. The following shows the distribution of the remaining pathway purities. This makes it apparent that we have a smaller number of interactions with medium range purity, which further shows the need to add weights to even out the dataset.

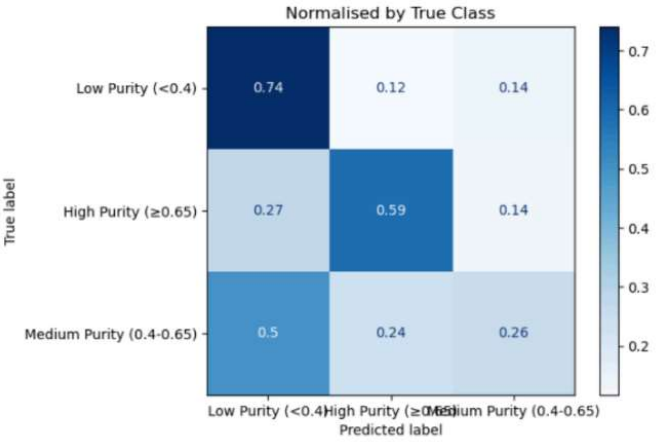



```
Total data points: 234355
Points in [0,1] range: 234355
Outliers: 0
```

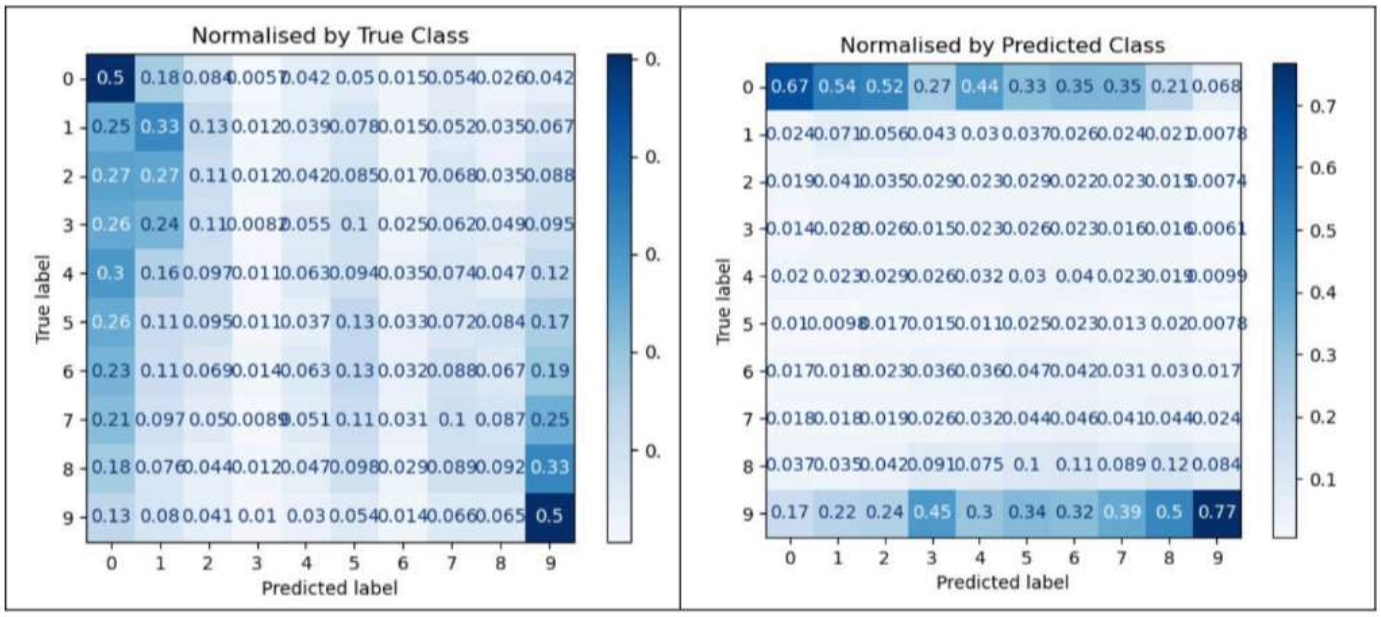
Then, I implemented the weights. This resulted in the identification of the medium purity compared to the complete lack of it previously. However, the model struggled to distinguish between low and medium purity, likely due to the small sample size of the medium purity

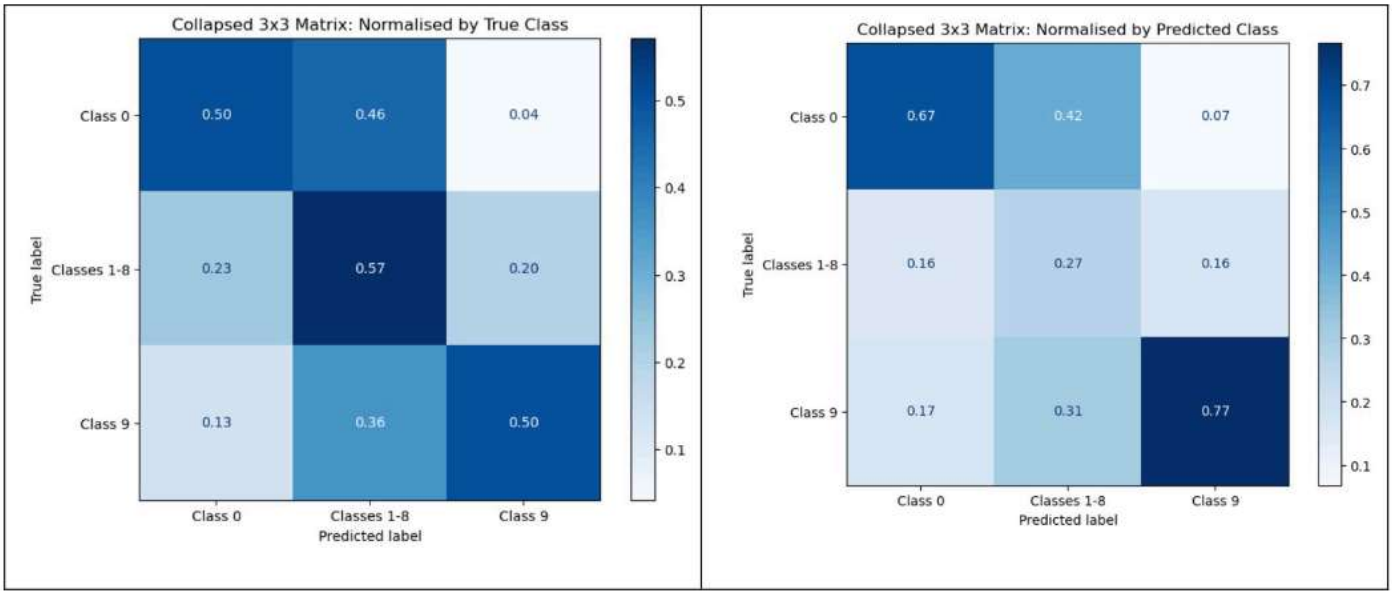
interactions.





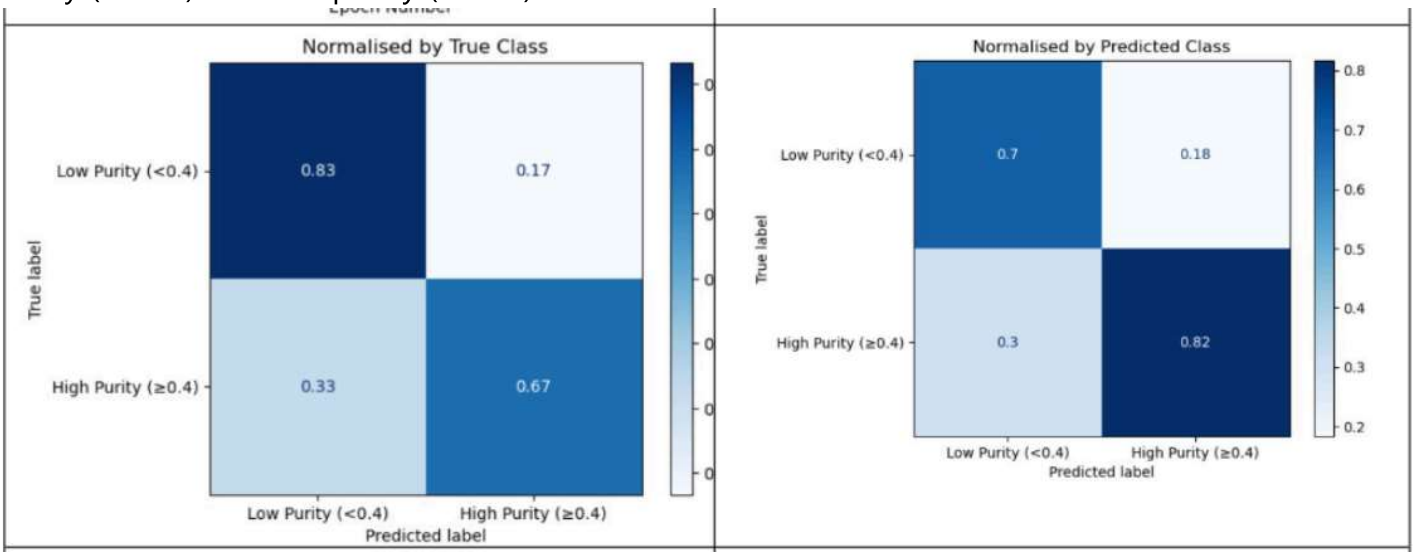
To rectify this, I experimented with making 10 classes which would distinguish the former block of ‘medium purity’ into 8 different classes, each with a 10% range. This would allow for a greater differentiation of each purity range; especially since these would be heavily weighted due to the smaller sample size. I built a 10x10 confusion matrix, and subsequently grouped the former middle purity class together.

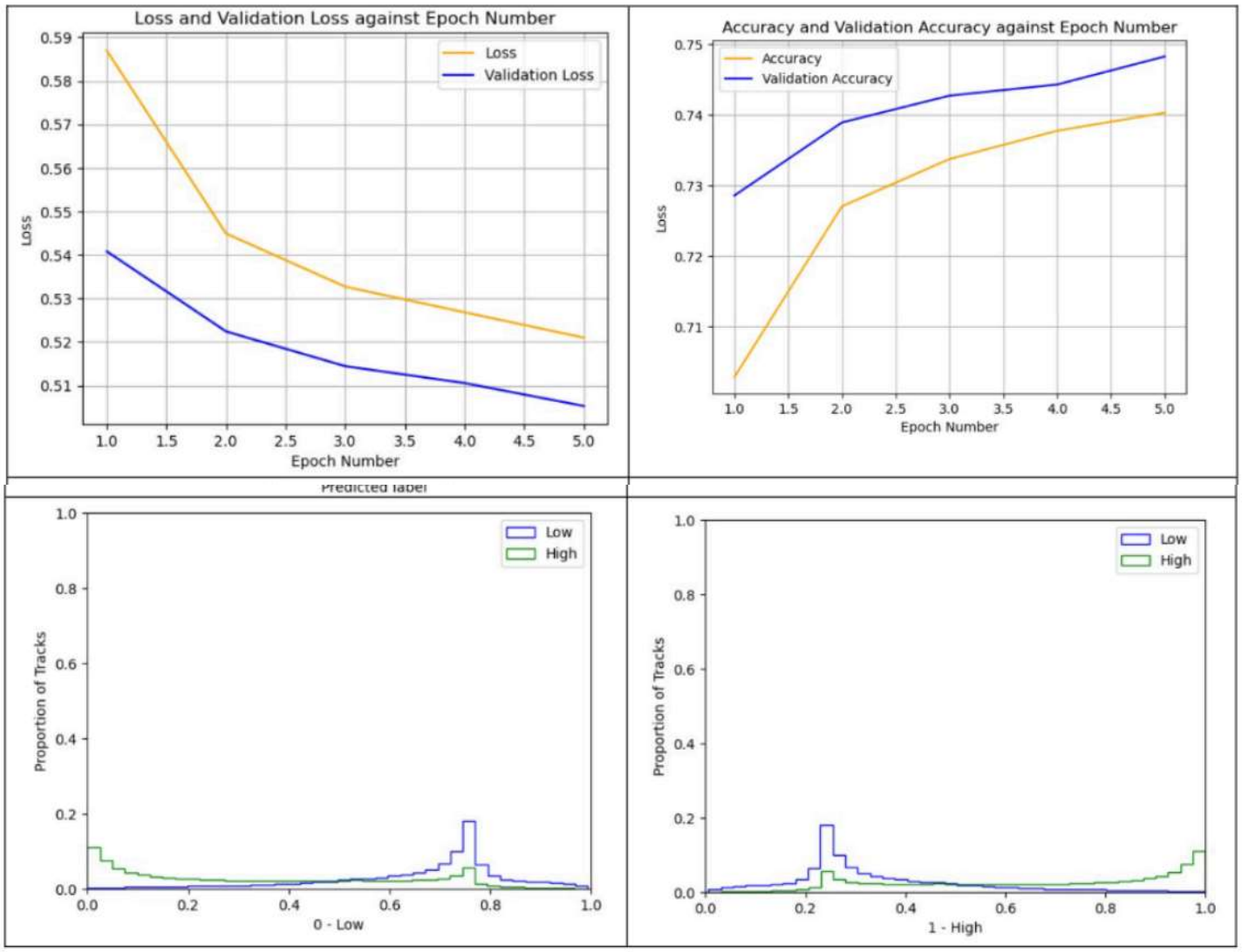




This shows some improvements in the medium class identifications, but also increases the misidentification of the medium purity with the high and low purities. So, we determined that this would not be a suitable way to separate the classes.

The last setting that we experimented with was building a binary model, sorting between high purity (>40%) and low purity (<40%). This resulted in a more accurate classification.





And so, this justified extending the model to 3D using a binary model.

What's next?

Extend the model to 3D.

10 July 2025

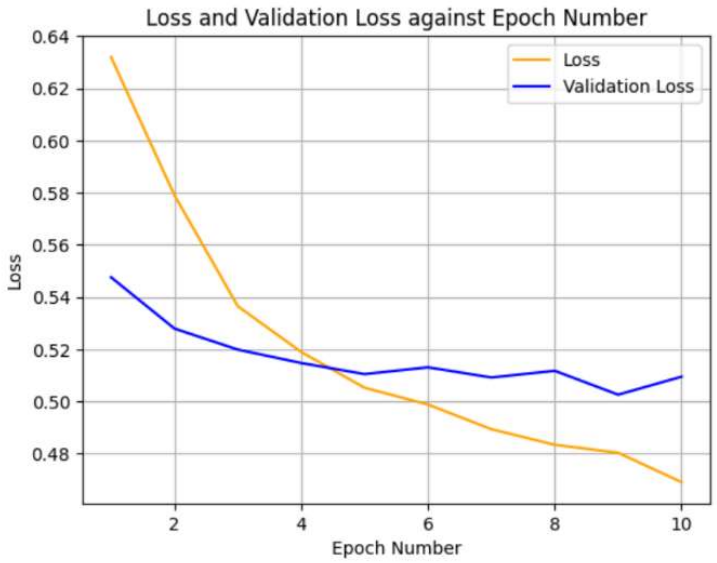
This week's work:

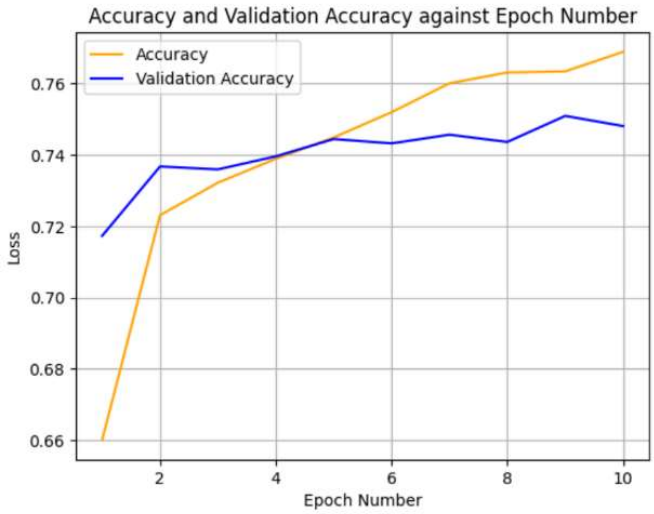
I finished my final day on 4 July 2025 by extending the model to 3D. The Lancaster machines were down in the morning, so I had to build this between 1pm to 6pm. The steps to do so was as follows:

- Change network to take three inputs, where we feed all three views, with a common normalisation

- Create new truth labels according to the truth label of the triplet. Ensure weights are still in place.
- Pass each view into the network one at a time, then flatten the outputs before passing into a sigmoid layer which accounts for the binary classification

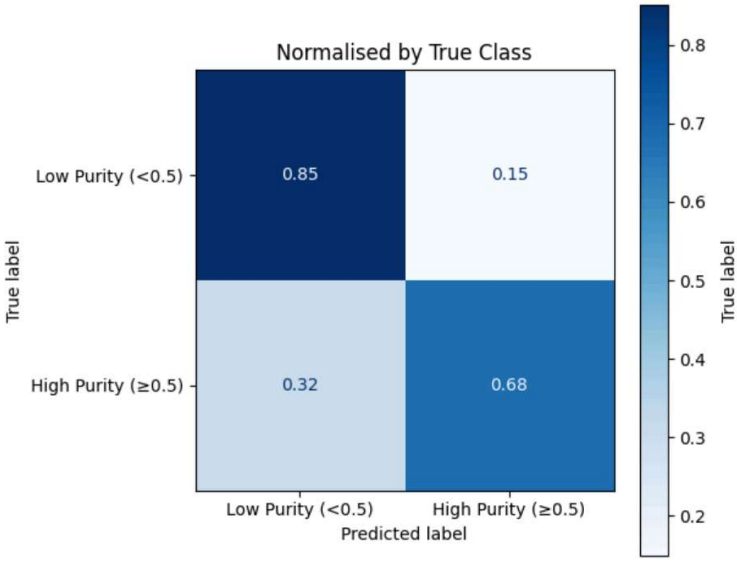
I ran this with a small dataset of 100 of each interaction, just so that I would not face any computational issues under the time constraint. But Isobel has kindly agreed to run this with the full dataset in some time. For now, here are the results from the smaller dataset:

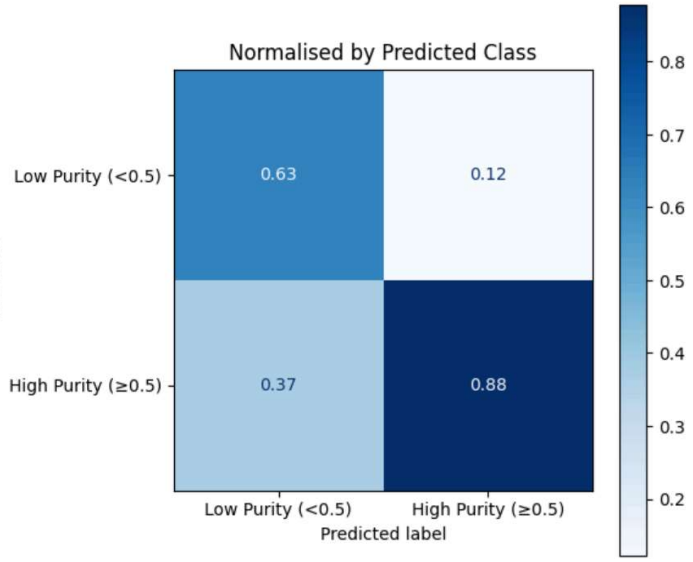




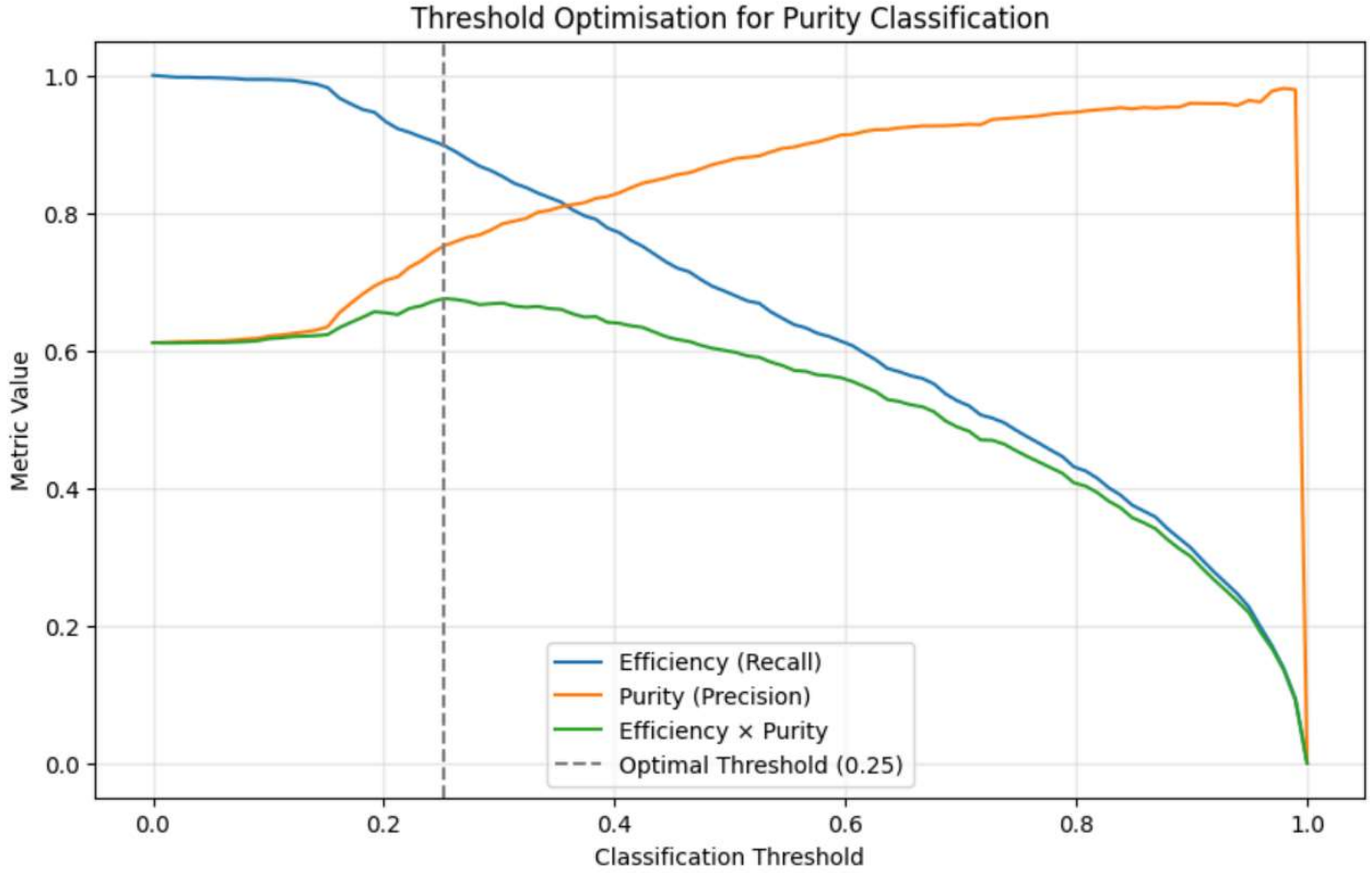
Inevitably there was overfitting due to the small sample size applied to a network that was handling thousands of interactions, but the training shows that the network is effectively learning even when the model is built for all 3 views.

The confusion matrix unfortunately shows some confusion that was previously not existent, since we see a high rate of high purity interactions being identified as a low purity one. This is again a consequence of running the smaller sample, where there are less low purity data available (as seen on the histogram from last week).

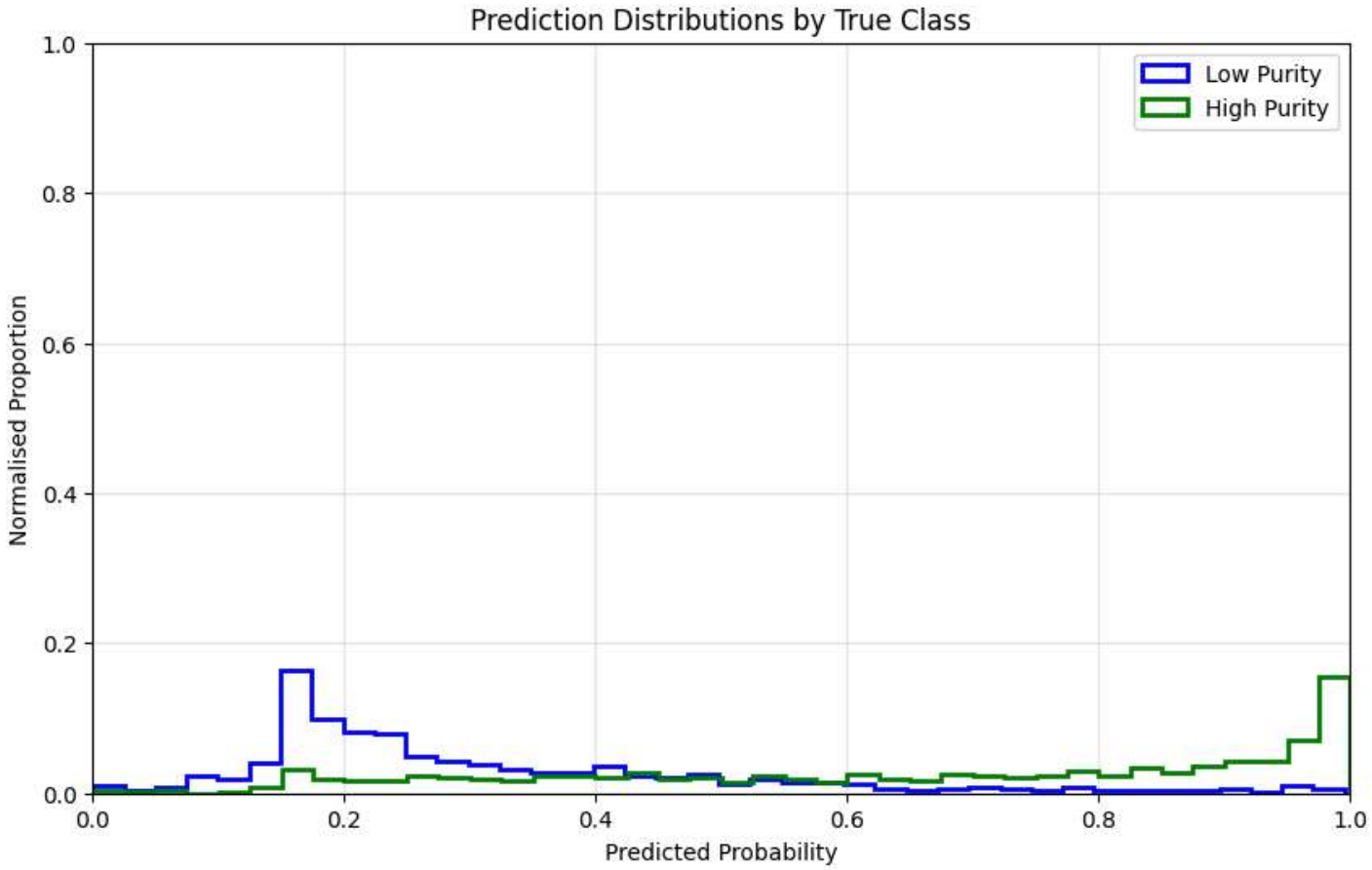




We also introduced a threshold optimisation, which determines the optimal classification threshold to identify one class to the other. The best choice for a boundary can be seen by obtaining this metric, as we did below.



The final metric was to see the prediction distribution. The goal was to see the two purities with a peak at opposite ends, showing that the predicted distribution is in line with the shape of the true distribution. In the plot below, we can see that that is indeed the case.



And so, this concludes my work in building a CNN that aids in identifying whether the reconstructed particle's pathway belong to the same particle, allowing us to catch interactions with false positives/negatives.

What's next?

See how this model runs with the full dataset. Present this information on a poster.