Neutrino Event Classification

PHAS0056, Mini-Project Report, Manas Chinthalapati

Contents

List of Figures

Introduction

A fundamental aspect of detector physics is being able to differentiate and identify particles, although this can sometimes be done using only the path of the particle and the track left behind, particularly what part of the detector the track we seen in, we are past the point in which identification is that simple. Tradition methods of identification are often laborious and time-consuming, limiting scalability. Convolution Neural Networks(CNNs) have powerful feature extraction tools and excel at hierarchical representations, they also have spatial in variance allowing them to detect patterns regardless of track position and orientation, this helps make the CNN more robust than other machine learning approaches.

These principles have been implemented in the classification of neutrinos from the images in a hypothetical detector that resembles that of the NOvA experiment.

Background

The project being discussed focuses on simulated data designed to replicate the results from the NOvA detector based at FermiLab in the USA. The particular focuse being on neutrinos. Neutrinos are fundamental particles the belong to the family of leptons. They do not compose of any smaller constituents and are therefore classified as elementary particles. Neutrinos have 3 flavours; electron neutrinos, muon neutrinos, and tau neutrinos, each flavour corresponds to a respective lepton. Normally it was thought that flavour was conserved however neutrino oscillations is a phenomenon where neutrino's oscillate in flavour as they travel through space. This is the only process that's been observed to break the Standard Model for particle physics.
In order to detect a particle it must first interact with something. The neutrino interaction with matter is weak, the mediators for the weak force is the W^+, W^-, and Z^0. The neutrino has no charge and so the only forces it experiences is the weak force and gravity. However due to the neutrino's extremely low weight the effect of gravity is ignored. This gives neutrinos 3 main interactions with matter that can be observed in a detector. Charged-Current(CC), due to interaction with the W^+, and W^- bosons and the Neutral-Current(NC), due to the interaction with the Z^0 boson.[1]
The CC interaction involves the neutrino converting into their respective lepton counterparts. This requires high energies. In this case the lepton is the one being detected, and by doing so this confirms the existence of the neutrino. Also the flavour of the created particle will tell us the flavour of the initial neutrino.[1]
The NC interaction involves not the neutrino changing into another particle, but instead transferring it's energy and momentum to a target particle. This interaction can happen due to any flavour electron. However NC interactions have identical amplitude for all 3 flavours, the flavour of the neutrino instead plays a role in the probability of interaction.

Neural networks are a machine learning structure built to imitate the function of the brain. The basic unit of the brains function are neurons. They transport electric signals from one neuron to another. The human brain has over 100 billion neurons, so this level of complexity is incredibly difficult to achieve, however that doesn't mean the structure can't be imitated whilst reducing the complexity. After all Nematodes can survive with just 302 neurons. Neural Networks are organised in layers of 3 types;

Input layer: The layer receives and passes the input data on. Each neuron of the input layer can be mapped onto feature of the data.

Hidden layer: This is where the network actually learns, and is where computations are performed on the data.

Output layer: Gives the final output of the Neural Network. Each neuron acts as an activation function and it can only output 2 inputs, taking in various inputs and assigning weightings, it then feeds this result through a function and if the value is above a certain threshold then the neuron will transfer the signal along.

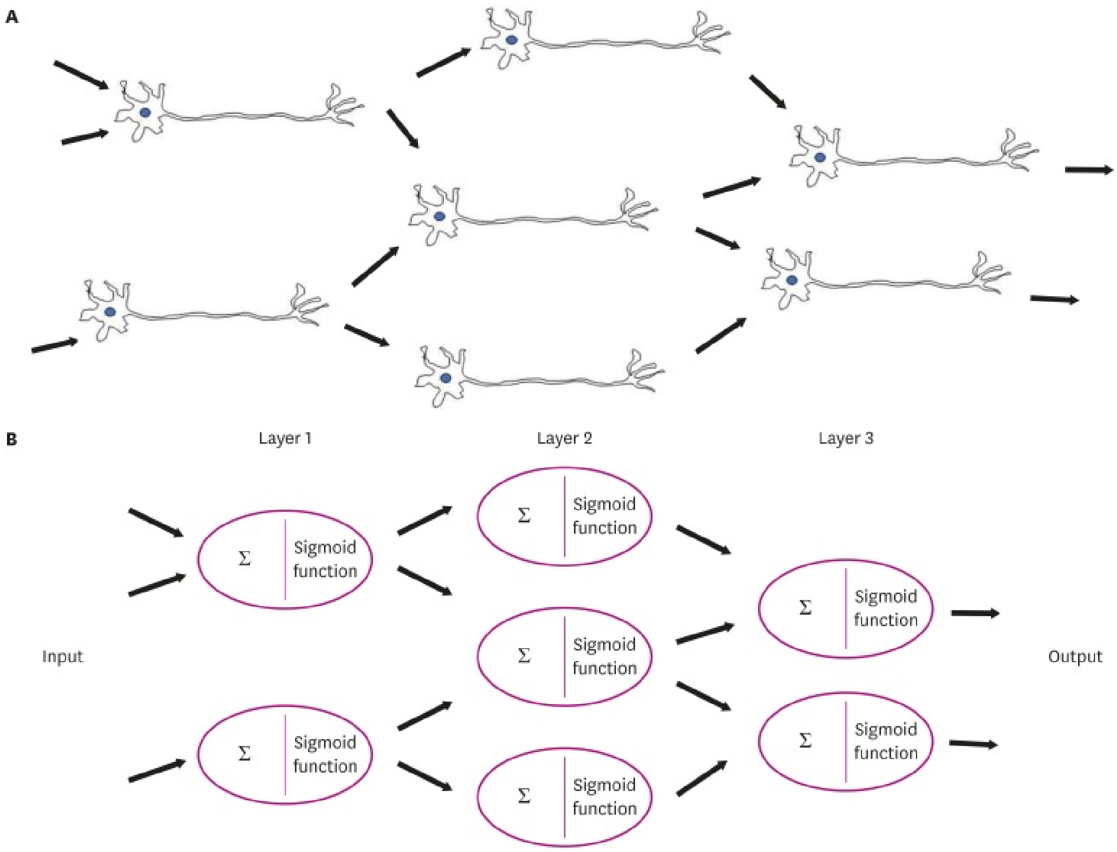

Figure 1 ***Comparison of a biological and artificial neural network:*** From Artificial Neural Network: Understanding the Basic Concepts without Mathematics[2]

The network learns by comparing the calculating the error between the predicted value and the actual value and then updating the weightings of the connections between nodes. Convolution Neural Networks are specifically designed to process image data, or grid-like data. As a simple 64x64 image has a minimum of 4096 weights(if every pixel is black and white), and this can go up to 12,228. While we could increase the number of hidden layers this comes along with 2 major problems. First this would increase the computational time and hardware required, but it would also cause over-fitting. This is when the network, or any machine learning program for that matter, is able to perform very well on training data, but poorly on unseen data. This can be due to a number of reasons but often comes down to the algorithm picking up on the random noise seen and not the fundamental features and patterns of the data.[3]

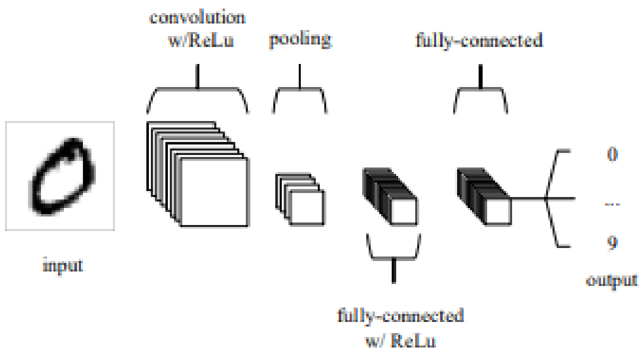

Figure 2 ***Simplified diagram of the architecture of a Convolution Neural Network:*** From An Introduction to Convolutional Neural Networks[3]

The input layer stores the input data. The convolution layers apply multiple filters on the input, called kernels, these filters are designed to extract notable features from the data. One key advantage is that the weightings used by each kernel is the same, in traditional neural networks the weightings are unique to each node, however for convectional neural networks all kernals share the same weights. This drastically reduces the complexity of the system[3]. In Fig 2 the activation function used is the rectified linear unit(or ReLu) for short. The following pooling layers perform downsizing, reducing the number of parameters. The fully-connected layers will then do the same thing found in the hidden layers of traditional neural networks.

We previously went over CC and NC interactions, however CC interactions are split into a further 3;

Quasi-elastic(QE) interactions: The target nucleon only changes it's charge, and no pions are directly produces. These interactions usually produce clean events, normally just 2 tracks.

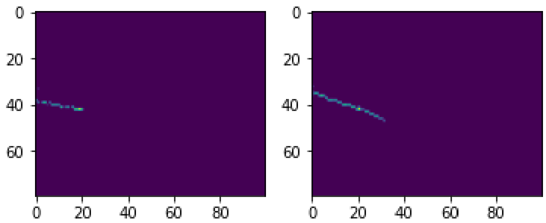

Figure 3 ***Image of QE interaction*** The 2 graphs combine to make a 3D image, the graph on the left is X against Z, and on the right is Y against Z

Resonance(Res) production: The neutrino excites the target nucleons to a resonance state, this then produces pions. This results in slightly more messy data than QE, but not as messy as Dis which we will get into now.

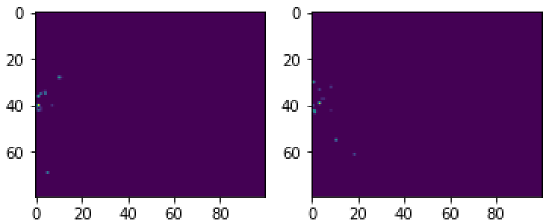

Figure 4 ***Image of Res interaction*** The 2 graphs combine to make a 3D image, the graph on the left is X against Z, and on the right is Y against Z

Deep inelastic scattering(Dis): A high energy neutrino penetrates deep into the nucleon and scatters a quark. This interaction results in many tracks and showers, and an overall messy event.

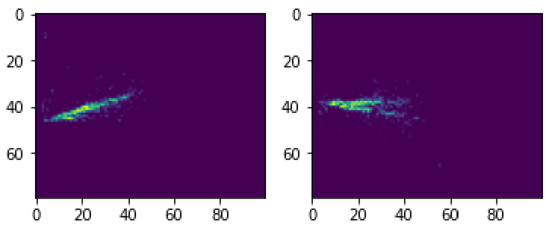

Figure 5 ***Image of Dis interaction*** The 2 graphs combine to make a 3D image, the graph on the left is X against Z, and on the right is Y against Z

1 Task 1, Binary Classifier

1.1 Normalisation

Normalisation is a standard practice in machine learning, it essentially translates data into the range [0,1]. This transforms all the data such that they are the same scale. Normalisation aids model performance on algorithms that rely on distance metrics by preventing larger scaled features from dominating. This will also aid in preventing over-fitting and improves the networks ability of generalization. In this task to create a binary classifier normalisation was implemented. Although all of the data used was of the same dimensions and size there are still some benefits to using normalisation. Including Scale Sensitivity, Regularization, and Interpretability.

1.2 Architecture

The architecture used to create a binary classifier to differentiate between neutrino's and other particles composes of two encoder blocks. The first block is consisting of a convolution layer of 32, 3x3 kernels with the ReLu activation function, followed by a 2x2 MaxPooling layer and ending with a 0.25 Dropout layer. The second encoder block is very similar but instead has convolution layer with 64, 3x3 kernels.

This is then fed into a Flatten, Dense, 0.5 Dropout and Dense Layer, with a

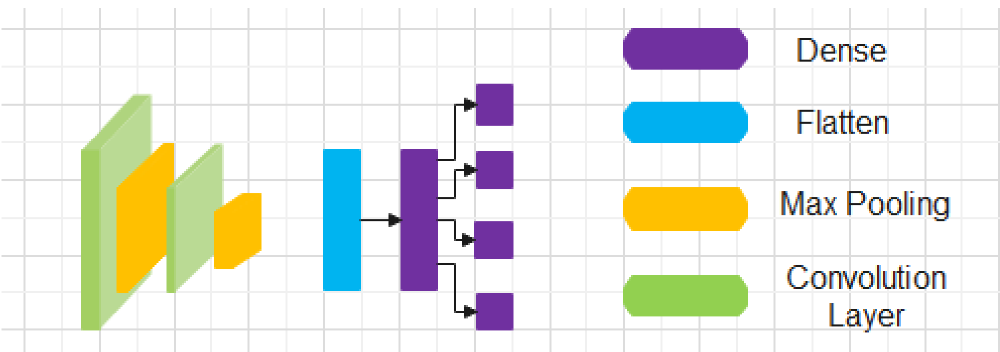

Figure 6 ***CNN architecture*** CNN architecture of the model used.

binary output and sigmoid activation. These final layers have a very similar structure to that of a tradition neural network. The convolution layers work to perform feature extraction, the Max pooling layers reduce dimensionality and the Dropout layers help prevent over-fitting. Note, the dropout layers have been excluded from the above diagram as they're job isn't so easily visually represented. The final dropout layer hover is instead shown as the splitting of the Dense layer.

1.3 Training and Testing

The total number of data points used was roughly 21000. This was then split into train and test data, initially a split of 80:20 was used, however when testing on a smaller data set this came with the issue of possible overfitting, therefore, the split was change to 75:25. However the distribution of categories is 88:12, this meant there was a chance of the network being extremely biased. If testing was done without any balancing this could give very misleading results, as an accuracy of 88% could just be the network predicting everything as a neutrino. So both a unbalanced and balanced testing set was made, and confusion matrices would be constructed as well.

When fitting the model a batch size of 100 was used, this was primarily due to the large data set. A range of epoch's were tested Although more Epoch's does

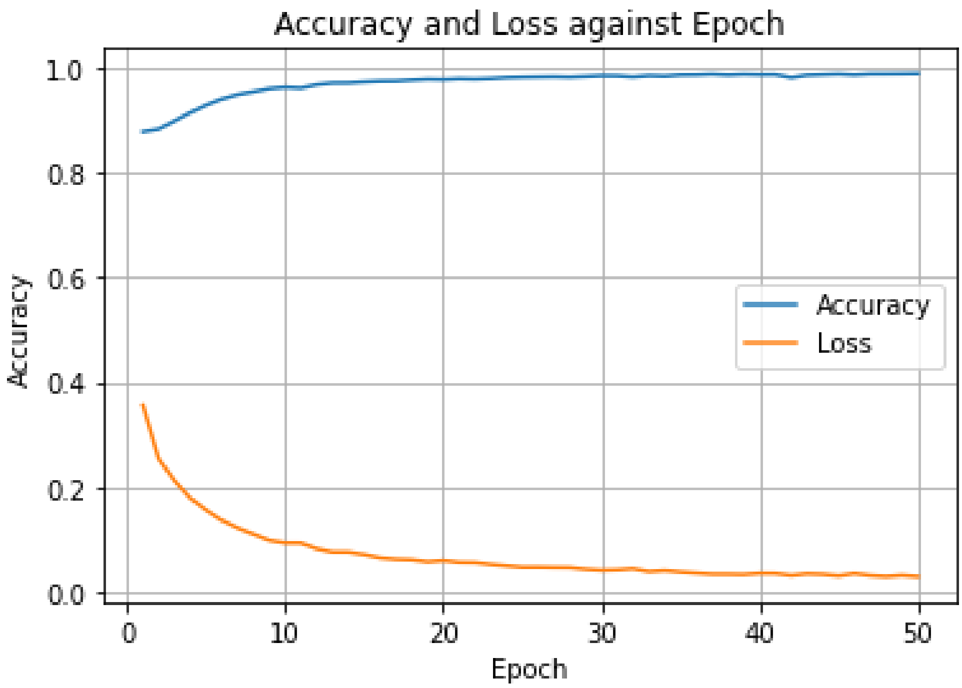

Figure 7 ***Performance of the model depending on no.of Epoch's used***

show an increase in accuracy and a decrease in loss all the way up to 50 epoch's, past 20 the effectiveness drastically diminishes. However the time taken the run the model increases linearly with no.of epoch's. Therefore 20 epoch's would give most of the model's performance and anything above 50 was deemed unnecessary. Followed by this a test was done to find the optimal threshold function, initially this would be set as 0.5 as the default by a range from 0.1-0.9 as the threshold value was tested. Fig 8 shows the results of the previously stated analysis. This

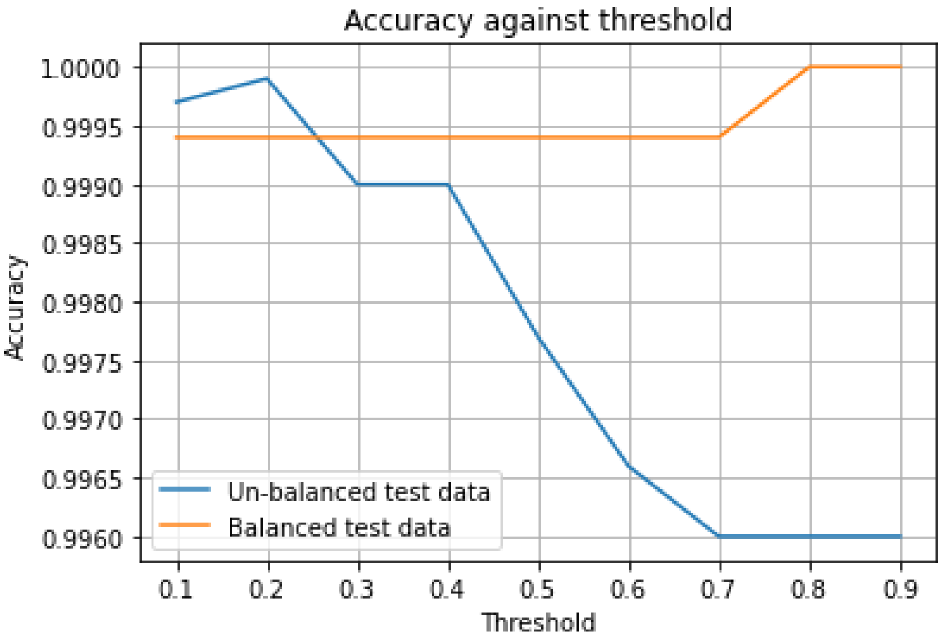

Figure 8 ***Performance of the model depending on value of threshold***

would indicate the perhaps a value of 0.2 would be best suited, however this data is incredibly misleading, as with a value of 0.7 and above the network would predict every track as not a neutrino, and with a threshold of 0.3 and below the network would predict every track as a neutrino. It was for this reason it was decided to leave the threshold value at 0.5 as changing it would lead to marginal changes in accuracy but along with many issues with reliability of data.

When testing our model an accuracy of roughly 98% was found for 20 epoch's. For the case of this study we will be considering the best case of 50 epoch's. This gave an accuracy of 99.86% for the un-balanced test data, and 99.66% for the balanced test data. These are incredibly good results but do indicate a slight bias in the model. Fig 9 and 10 show that the model has a good accuracy without producing biased predictions.

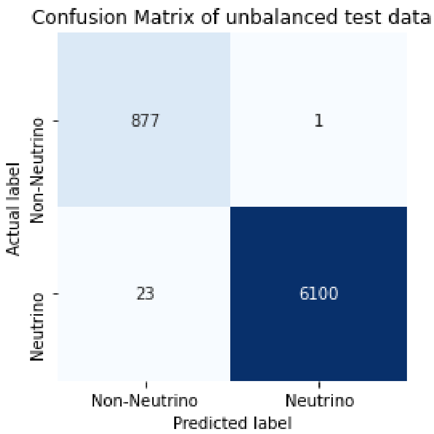

Figure 9 ***Analysis using unbalanced test data***

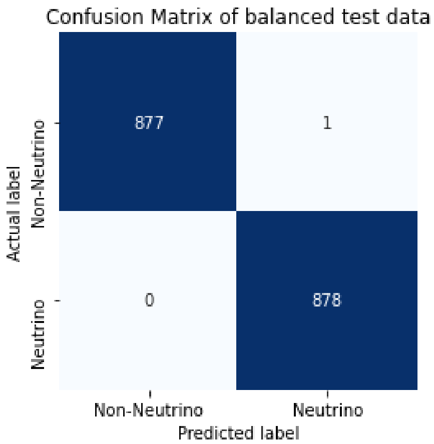

Figure 10 ***Analysis using balanced test data***

2 Task 2, Analysis on Meta Data Variables

Then next task was to analyse the model an a variety of meta data, these being the type of interaction, QE, RES, DIS. The accuracy depending on these 3 interactions was;

QE: 0.9954

Res: 0.9979

Dis: 0.9962

Other: 0.9954

This is somewhat unexpected as since QE interactions have the clearest tracks it would be expected that the accuracy on predicting them would be the highest, and hence the accuracy on predicting Dis be the lowest. The reason for this was due to the abundance of the 3 categories in the data. The abundance in the data was 13:71:16 for QE, Res, Dis respectively. This shows that the reason for the unexpected accuracy was likely due to a surplus in Res data, allowing the model to better predict them. The training and test data were therfore balanced and the new accuracies were found to be;

QE: 0.9989

Res: 0.9956

Dis: 0.9991

This shows the accuracy of QE and Dis being incredibly similar, however Res now seems to be lagging behind. When looking at the typical Res track and comparing them with QE and Dis tracks, fig 3,4,5 we see that Res interactions have considerably less notable features. This is likely the cause of the lower accuracy of Res predictions.

3 Conclusion

The use of machine learning, in the form of a CNN as a binary classifier has been theoretically justified and practically shown. The created model had an incredibly good accuracy of 98%, which could go up to 99% depending on the computational time given. The model also showed no signs of biased prediction on classes, despite a largely biased training data set. The model accuracy on specific interaction types however was more dependent on the size of said interaction data given. When training and testing data was balanced it was found their was likely an intrinsic drop in performance when regarding Res interactions, likely due to the lack of discernible feature present in track images.

References

1. Saha, S. & Das, P. K. Neutrino Interactions and detections: Review and future potential. *International Journal of Modern Physics A* **37** (Aug. 2022).
2. Han, S.-H., Kim, K. W., Kim, S. & Youn, Y. C. Artificial Neural Network: Understanding the Basic Concepts without Mathematics. eng. *Dementia and neurocognitive disorders* **17,** 83–89. ISSN: 1738-1495 (2018).
3. O'Shea, K. & Nash, R. An Introduction to Convolutional Neural Networks. eng. *arXiv.org.* ISSN: 2331-8422 (2015).