PHAS0056 Final Report

Marcel J. Mordarski
SN: 19009471

Neutrino events classification

A machine learning approach

Abstract

Neutrino oscillations defy the framework of the Standard Model. Therefore, any research into the interactions of neutrinos with matter would benefit from a classifier of events whose nature stems from neutrino oscillations. This report reviews literature in the area of neutrino events classification and presents a less computationally intricate approach to the problem. A two-branch network analysing NOvA experiment-esque data reached the accuracy of 67% on the testing dataset. Several approaches to improving its performance were discussed (eventually allowing for 84% accuracy). The focus of the discussion is on the frequency- vs. feature-driven learning of neural networks. The results were then analysed as per the metadata type. Some extensions to the work include the classifier of the final state and energy predictor.

1 Introduction

LIGHT, FAST, ELECTRICALLY NEUTRAL, hardly interacting with particles.

If this was a job description, only neutrinos would be considered for an interview at the *Standard Model Inc.* It should be noted that the non-zero mass characteristic was added to the advertisement only later, as the company originally had not predicted a particle like that in its long-term strategy. This report looks into a case of a "neutrino"-employee that continues to challenge the entire firm to improve, pointing out deficiencies in the organisational structure.

Namely, the instabilities in our knowledge [1, 2] created by, and machine learning approach to the investigation of the neutrino oscillations shall be discussed in the context of the NOvA experiment.

The formal refinement of the neutrino oscillations underlines two of the most powerful principles of Quantum Mechanics – superposition and change of basis. For the SuperKamiokande experiment investigating neutrino oscillations to prove that these particles indeed have rest-masses (by contrast to the predictions stemming from the Standard Model) involved the change of their representation between two bases: mass-constant and flavour-constant neutrinos. [1] The equations

$$|\nu_\alpha\rangle = \sum_i U^*_{\alpha i} |\nu_i\rangle \tag{1}$$

$$|\nu_i\rangle = \sum_\alpha U_{\alpha i} |\nu_\alpha\rangle \tag{2}$$

where

$|\nu_\alpha\rangle$ = a neutrino with definite flavour such that $\alpha \in \{e, \mu, \tau\}$
$|\nu_i\rangle$ = a neutrino with definite mass m_i, such that $i \in \{1, 2, 3\}$,
$\mathrm{U}_{\alpha i}$ = the Pontecorvo–Maki–Nakagawa–Sakata matrix, [3]

say that neutrinos are classified into

1. the electron-, muon-, or tau-flavoured,
2. constant-mass ones.

Each of these basis vectors contributes a differing amount to the resulting superposition. Using the PMNS matrix, both representations can be interchangeably expressed as a linear combination of each other (i.e., the flavour ket can be written as a linear combination of the mass-eigenstates, and vice versa). [4]

Saying flavour is jargon for "type". To this day, nonetheless, we do not know the precise mass of these particles for each flavour. [5] This raises the question of what is actually known about the oscillations of neutrinos.

It is agreed that **neutrino oscillations** are the *process of changing the flavour while moving through matter (during the weak interactions).* [6]

Technically, it is equivalent to the observation that an imaginary phase changing in-flight corresponds to each basis vector. As the result, a beam of neutrinos initially of a single type (say, the μ-flavour) has a finite probability to be measured to contain the remaining two flavours, too, when weakly interacting with matter.

Practically, it means that the measurement involving the detection of neutrinos need to take this phenomenon into account to be precise. Otherwise, the underlying nature of particle physics may remain covered.

Therefore, any community whose work relates to neutrinos would clearly benefit from developing a classifier of neutrino events.

In this context, what makes NOvA experiment important to this report and the developed algorithm is that it is the richest repository of data that can be used for the training of neural networks. [7]

In this experiment, a beam of neutral neutrinos can be detected when passed through a medium. Upon interaction, either neutrino is turned into its charged partner (Charged-Current interaction) or it scatters but remains as a neutrino (Neutral-Current interaction). This report will focus on differentiating these two types of interactions.

NOvA set-up consists of a beam mainly of μ-neutrinos, with energies of a few gigaelectronvolts. The complex multi-step process creates a neutrino beam with a broad spectrum of energies. As described earlier, it will most likely experience the oscillation and include some ν_e component, as well as some antineutrinos, $\tilde{\nu_\mu}$ and $\tilde{\nu_e}$. Depending on composition, there will be either a CC or NC-event.

The NOvA detector is filled with instrumented tubes with a liquid scintillator. This is an organic molecule, which means that this report is caveated to the interaction of neutrinos with carbon atoms. This feature ensures that weak interactions can be detected.

1.1 Why do we care about neutrino oscillations?

The reasons for developing a neutrino events classifier are several and applicable in different contexts.

The first value of investigating neutrino oscillations is in inspiring scientists to be creative in designing experiments.

Nothing shows the sophistication that needs to be achieved in linking simple observations better than the SuperKamiokande experiment. It joined three observations when analysing the results from a detector of neutrinos put on the ground.

Firstly, since neutrinos interact weakly, they could be incoming towards the detector from above or below. [1] That is the Earth would be penetrable for them.

Secondly, the flux of neutrinos is isotropic on Earth, so the flux from above and below should be the same. A small correction should be introduced, though, to account for the Earth's magnetic field. [8]

Thirdly, various testing showed that SuperKamiokande cannot detect τ-neutrinos. [1]

When put together, these observations amounted to the simple explanation why the number of μ-neutrinos entering the detector from below was half the number coming from above, while the number of e-neutrinos did not change. Namely, μ-neutrinos had oscillated into τ-neutrinos, which is a stunning piece of scientific reasoning.

Other useful applications of the neutrino oscillation mechanism could be in astronomy and theoretical physics.

The first area continues to apply this mechanism to solve the solar neutrino problem in the more and more refined contexts. Also, it is useful to characterise some stars and uncover the nuclear reactions of the early universe. [9]

On the other hand, the improved understanding of the mass acquisition mechanism for neutrinos could be gained by considering neutrino oscillations. This debate has not been settled definitely so far. [10]

1.2 Why is machine learning suitable to process it?

Machine learning techniques seem to be particularly suitable for the task of classifying neutrino events in the NOvA experiment because the relevant data is:

1. voluminous,
2. often noisy,
3. to be interpolated.

Developing a machine learning algorithm for such a dataset has many advantages over other methods:

1. the ease of pattern detection,
2. the flexibility of the algorithm adjustment,
3. the robustness against particular quirks of data if coded adequately.

1.3 Examples of similar approaches in the literature

The application of machine learning (and neural networks before) to high-energy physics dates back to the paper by Denby in 1988 [11]. This was a direct application of the neural network model by Hopfield from 1982. [12]

Today, machine learning is a standard processing tool, as indicated by the exponential growth of the papers mentioning machine learning. In particular, not only has machine learning been applied to high energy physics, but the very matter of this report as well. [13] The work by Psihas *et al.* will be looked into to motivate some of the approaches taken in this report.

It has not escaped the author's attention that there are algorithms from outside the high-energy physics that resemble some similarities with the NOvA experiment data. Most notably, those for the classification of the MNIST dataset could be applied. [14]

1.4 The plan of report

To address the need for a neutrino event classifier as motivated in previous sections, this report will investigate charged-current neutrino events. The type of the particle is dependent of the phases of eigenstates that will have been acquired by an oscillating neutrino before the interaction.

The following scientific aims are set out:

1. Reconfirming that machine learning algorithms can distinguish the ν_μ CC-events from others.

2. Analysing the influence of the contents of the metadata on the accuracy of training and prediction.

3. Develop an algorithm for the finalstate classification and energy prediction.

These aims will be achieved through addressing the following tasks:

1. Preparing the data adequately.

2. Creating a binary classifier of ν_μ-events.

3. Checking the effectiveness of this new method against the literature.

4. Using these experiences to modify the neural network for other purposes.

2 Machine learning tools

2.1 Comments on data processing

Let it be elaborated on the specification of the dataset that is about to be analysed. This section will cover the manipulations done to ensure the swift work of the classifier.

The data supplied for the training in this task consists of a series of double pictures (views from the xz- and yz-planes) labelled with the type of neutrino interaction, final state, the energy of leptons and neutrinos. In numbers, there were more than 400 files each consisting of roughly 6500 doublets of pictures.

This amount of data raises three issues that need to be taken into account to make the machine learning approach work. They concern RAM, normalisation, and the split into the training, validation and testing datasets.

Given the limitation on RAM in the platform used (Google Colab), a few out of the 400 files were selected to be a sample for training and testing. Further studies could increase the number of pieces of data.

A good coding practice used in similar projects is to normalise the data. There are several ways that one could use to do it; for example, the work on this report included normalisation of every picture with respect to

1. the highest pixel on that particular picture,

2. the highest pixel in the entire dataset used for the training and testing of the neural network.

The former was found to be ineffective, as even the dense neural network could not find underlying patterns in the data and give sensible results (the training resulted in the testing accuracies around 10%). Therefore, the latter approach was chosen resulting in reasonable accuracies.

Finally, concerning the split into the three datasets, the train_test_split() was applied to the connected data from the chosen files to ensure acceptable randomisation and unbiasedness.

Additionally, building a binary classifier now came in handy to simplify some data. Instead of considering 16 different types of CC and NC interactions, ν_μ were ascribed 1, and the remaining ones 0.

There is a complication involved, though. A preliminary look at the dataset proved that reducing the complexity of data to two categories made it imbalanced, i.e., the data for one of the types was more than seven times more numerous than the other one. This may influence the results. Therefore, a case where an equal number of both CC and NC interactions is exposed to the network should be discussed.

2.2 Possible approaches to the task – MNIST-like pictures, noise, and data splitting

Having the picture of the data in mind, one can proceed to develop the machine learning strategies. This section will showcase some possible approaches in the task similar to the one that is being discussed in this report.

The similarity of the MNIST dataset of handwritten digits and the NOvA results suggest two useful machine learning techniques that could be adopted straight away – the use of convolutional neural networks (CNNs) and the decomposition into principal components (PCAs).

CNNs will be valuable for this task due to the translation-invariance properties. They slide convolution kernels through the input features and provides translation equivariant responses known as feature maps. These maps help to classify the collision (of the neutrinos with the scintillator) as they could happen at any position in the considered frame; in other words, their position is not useful for the characterisation of events.

PCAs would provide aid on the memory side of the algorithm, as they find the underlying patterns of data, and hence, they can reduce the training times. However, the trial and error showed that the loss of the information is too significant for this method to work in the considered binary classifier. Therefore, batch training was implemented to make up for this unsuccessful attempt.

During the mini-project, two more strategies were tried out: adding noise to the pictures and processing the doublets of pictures in separate neural networks that were later concatenated to give a single result.

The former is motivated by the imperfection of the experiment set-up — noise is dependent on numerous external factors like the scintillator used, beam parameters, etc., so it is beneficial to make the neural network robust against such complications. One way is to input artificially noised pictures in the neural network so it expects some kind of randomness in them.

The latter has been alluded to in the literature. As the authors of [13] indicate, the two planes that were recorded may contain differing information and hence their separate treatment may improve results. For example, some of the events could happen precisely in one of the planes, so only one picture may contain the correct information about what was the type of interaction. This approach is a double-edged sword, though. Some information is separated from possible quirks in the other plane, making the model more robust. On the other hand, the quirky result is treated as equal to the normal one (because the single input network is told that a particularly unusually looking picture corresponds to an interaction greatly dissimilar to it).

3 Algorithms used

3.1 Psihas network

The authors of [13] approached the classification by creating a two-branched network with a single output layer – see the notebook for the implementation of their network. Their approach was characterised by a high number of trainable parameters, numerously-used inception layers, and occasional use of sigmoid activation functions.

The first is to capture the complexity of data by analysing the maximum possible number of features. In other words, the ratio of the amount of information to neurons was minimised to pass all the information between subsequent layers.

The second is to tune the network training to the quirks of the data by fining the best pooling kernel. In theory, it helps to adjust the weights of connections.

The third is to make the results sharp by nonlinearly amplifying them.

The following neural network aims at reducing the number of trainables to speed up the training while maintaining accuracy.

3.2 ν_μ-classifier

The scheme of the network can be seen in appendix A.

The network has two branches that later merge into a single one.

The sequence of layers, CNN–max-pooling–normalisation, starts each branch and is then repeated. What follows is a drop-out layer, flattening, and two relu-activated dense layers before the branches are connected.

Already in the single-layer, there is a dense layer with two neurons to give two outputs (i.e., 1 if the event is CC, and 0 if it is NC).

Let the sequence *CNN–max-pooling–normalisation* be discussed.

The CNN layer detects the features of the events focusing on their shapes and other characteristics like pixel brightness instead of the position. As described previously, the spatial invariance of the shape of the events is thus taken into account in the training. Also, the training is sped up as CNNs are not densely connected. Resultantly, some information is lost, but the number of trainable parameters is reduced while maintaining good accuracy scores.

Such data is then customarily followed by the max pooling. The way PCAs bottleneck the data to find the underlying patterns, max-pooling strengthens the most prominent (here the largest/brightest), value in each patch of each feature map created by CNN. The resulting downsampled feature maps highlight the most present feature in the patch. Conceptually, it means that the spikes are prioritised over the average features when minimising the loss function value.

Finally, this is all followed by a normalisation layer to ensure that the data stays normalised as in the beginning. In general, it may not be the case because the max-pooling layers may amplify some values beyond unity.

The kernel sizes were kept small (e.g., 3×3) to keep high number of parameters.

Having created an edgy picture of the interactions, one needs to ensure that it is not over-trained on a particular set of data. Drop-out layers may be helpful in so doing as they randomly set input units/weights to a null value for some weights. A frequency of rate set to 0 at each step during training time prevents overfitting to some extent. As it was the case for the MNIST set, 0.2 was chosen to give the best results in terms of the overall accuracy.

Another way to reduce the inevitably growing amount of data is to use the flattening layers. When analysing the collisions of neutrinos, all of the layers are merged into the background layer to reduce the file size. This only makes sense after the max-pooling layers were applied because otherwise, the processed snapshots of the collisions in the two planes would be overwhelmingly biased towards the highlights.

What is got after such processing in each branch are signals with the main features more pronounced but not caricatural. It is desirable to pass them through dense layers at this stage to expose the network to the details of these signals. In other words, all the information from the previous layers needs to be analysed to return decent learning results.

Therefore, two layers of fully-connected layers follow in each branch. The number of neurons is lowered from 32 to 16 nevertheless to keep the number of trainable parameters low and prepare the branches for concatenation.

Finally, the two branches are joined, as the output has to be a one or a zero – this is a requirement for a binary classifier. Consequently, there are two neurons in the last layer/the softmax was used.

It is important to eventually discuss some features important for the whole network.

Firstly, the loss function was BinaryCrossentropy() which is a suitable for the binary classification. This is because it compares each of the predicted probabilities to the labels (either 0 or 1). Next is the calculation of the score that penalises the probabilities based on the distance from the expected value.

Secondly, the optimiser was Adam. It dynamically changes the learning rate taking into account the gradient (from the gradient descent algorithm) and has been proven to be efficient in the stochastic setting (which is the case for this task). Importantly, this automatisation provides the advantage of easier use over other optimisers also because the learning rate does not have to be set before the training.

Thirdly, to ensure that what is got is a probability of assignment a particular label to the passed picture, a non-linear activation is used. This is in contrast with the practice for the previous layers, as relu was chosen there over softmax or sigmoid.

Fourthly, the set of metrics also determines how well the training goes. The suitable choice seemed "accuracies" because it evaluates the fraction of right predictions made by the model.

3.3 Improvements to the ν_μ-classifier

3.3.1 More convolutional layers

One way to improve the results of the ν_μ-classifier is to add more convolutional layers. This would possibly result in the extraction of a greater number of distinctive features for the CC-events. That said, one needs to remain cautious as adding too many layers (i.e., over-extraction of features) carries some risk. Teaching the network some features which are specific to the considered dataset (and not to the CC-events in general), such as the noise in a particular detector, may give poorer results on a different dataset.

3.3.2 More dense layers

If the results are unsatisfactory, improvement may be made by adding more dense layers. By contrast to adding more convolutional layers, this strategy suggests that the features were correctly identified but not exposed to the network in the adequate depth.

3.3.3 Noise

The addition of noise for the training mentioned earlier is straightforward, but it gets tricky when deciding its size. Two methods can be tried out: adding noise from a Gaussian distribution before or during the processing.

According to the first standard practice, the uniform distribution was chosen to generate noise. Its range was capped at 0.005 due to the normalisation and the brightness of the pictures before the normalisation. It was added to the pictures before passing them into the network

When it comes to the noise layers in the network, they have the advantage of doubling the size of the dataset in a way. Because the network is first shown the pictures without noise, and then the later layers have to process noisy ones, the network learning is improved.

3.3.4 More dropouts

Although the intuition about the dropout from the MNIST classification applies to a variety of problems (it was used here by setting the dropout rate to 0.2), it may be deceptive for the ν_μ-classifier. Hence, increasing the rate of dropout or adding more dropout layers may lead to greater robustness of the model against the special features of the data.

It should be noted that an improvement of the result as the effect of this action (and likewise the addition of noise) indicates that the general features were extracted acceptably.

3.4 Extensions

3.4.1 Final state classifier

Conceptually speaking, the development of the final state classifier is a more refined version of the ν_μ-classifier. The difference is that there are more labels now that the model needs to discriminate against.

Let it be assumed that the extraction of features by the ν_μ-classifier was acceptable (mentioned in 3.3). Thus, it can be inferred that incorporation of more dense layers to the model would be sufficient to cope with the increased complexity of the classification.

A logical change in the structure of the model relates to the number of labels – their number and the number of neurons in the output layer must match. The loss function also needed to adjust (it was change from binary to categorical).

All the ways to improve the classification described in the previous section may apply here.

3.4.2 Energy predictor

Last but not least in terms of the trickiness was the machine learning tool for the energy prediction. Even though the data makes a distinction between the lepton and neutrino energy, the algorithm for this task should be similar. This is because it is tasked with assigning a real number (from two slightly varied ranges) to a doublet of pictures.

This invokes changes in the architecture of the ν_μ-classifier concerning the number of output neurons, the number of convolution and dense layers, the loss function, and the final activation function. The following model will be a *regression model* (although eventually, it is desirable that it performs interpolation rather than just regression).

Firstly, the output layer consists of only one neuron as the output should be a single value.

Secondly, it is anticipated that there should be more both convolution and dense layers. This is to extract and process information better (especially important when there is a continuum of labels).

Thirdly, regression models aim to minimise the error between the true value and those calculated by the model. By comparison to the classification problem, this requires a change of the loss function to mean squared error (MSE) instead of accuracy.

Finally, the output no longer needs to be a probability value. Thus, to get a continuum of possible outputs, linear is chosen as the activation in that last layer.

3.5 Metadata analysis

It is also of interest to see how models work depending on the type of data that is supplied because such analysis may suggest quirks of data that future improvements would need to account for. The following interpretation is chosen as the prompt of the task is ambiguous: the basic ν_μ-classifier trained on the full dataset will be asked to predict labels of a subset of data with particular characteristics, e.g., the final state, range of energy. Therefore, each section of the notebook groups the data into such subsets.

The analysis will be done for the CC/NC type, interaction type, final state, the energy of lepton and neutrino.

Regarding the final state and interaction type, it is useful to differentiate between the following interactions:

- quasi-elastic (the simplest interaction, at lowest energies): normally clear paths,

- resonant (where the nucleon is excited to an unstable ‘resonant’ state, which then decays): fewer clear paths,
- deep inelastic scattering (where the nucleon is broken up, and the neutrino scatters from a quark inside it): many tracks and showers.

A particular attention will be paid to analysing the correlation between the change in accuracy of prediction and the number of shower/paths.

4 Results

EVERY ALGORITHM mentioned in the previous section was run in the attached iPython Notebook. The results are summarised in this section.

4.1 Psihas network

The model had 190 475 parameters in total, 189 995 of which were trainable.

The training was done on a batch size of 25 during two epochs – see Discussion for more information.

The loss was 0.3724. The accuracy was 0.8786.

The confusion matrix showed that 0 (0%) pictures were correctly identified as NC events, and likewise, 2710 (88.6%) were correctly identified as CC events. At the same time, 350 (11.4%) events were classified as NC even though they were CC, and likewise, 0 (0%) NC events were classified as CC.

The model clearly experienced over-training, as the accuracy on the validation set was lower than that on the training set.

4.2 ν_μ-classifier

4.2.1 Imbalanced data

The model had 16 354 parameters in total, 16 226 of which were trainable. This is a reduction of the order of magnitude as compared to the Psihas model.

Two approaches to training were adopted. In the first case, the batch size was 100 and the training lasted for two epochs. In the second case, the batch size was 25 and the training lasted for eight epochs.

While the losses were the same for both and equalled 0.6931, the first model performed significantly better than the one with a smaller batch size in terms of accuracy: 0.6739 v. 0.3925.

Greater insight can be gained by analysing the confusion matrices.

For the model trained on the bigger batch, 110 pictures were correctly identified as NC events, and likewise, 1952 were correctly identified as CC events. At the same time, 266 events were classified as NC even though they were CC, and likewise, 732 NC events were classified as CC.

For the model trained on the smaller batch, 210 pictures were correctly identified as NC events, and likewise, 1051 were correctly identified as CC events. At the same time, 132 events were classified as NC even though they were CC, and likewise, 1627 NC events were classified as CC.

Another difference between the two approaches was the over-fitting: the performance of the model trained on a bigger batch was better and did not show the over-fitting whilst the model trained on a smaller batch over-fitted. This is the case as the accuracy on the training set was lower than that on the validation set for the first model. Despite fluctuations, the opposite was generally true for the model trained on the smaller batch.

4.2.2 Balanced data

The claim that the CC/NC-unevenness of the data influenced the training and testing performance was looked into. The data was down-sampled probabilistically so that the number of CC and NC events supplied for training were equal. In detail, the events corresponding to "one"s were removed from the data put into the training_test_split() in the quantity stemming from the number of "zero"s.

The model had 16 354 parameters in total, 16 226 of which were trainable.

The loss was invariantly 0.6931. As compared to the approach from 4.2.1, the accuracy was improved – it reached 0.4934.

The confusion matrix reflects these changes. Now, 160 (15.1%) pictures were correctly identified as NC events, and likewise, 365 (34.4%) were correctly identified as CC events. At the same time, 401 (37.8%) events were classified as NC even though they were CC, and likewise, 136 (12.8%) NC events were classified as CC.

Over-fitting was not the case.

4.3 Improvements to the ν_μ-classifier

This section scrutinises the attempts to improve the model initially shown to work.

4.3.1 More convolutional layers

The model had 26 968 parameters in total, 26 840 of which were trainable.

Given the results from section 4.2, the training was done on a batch size of 100 during four epochs – see Discussion for more information.

The loss was invariantly 0.6931. As compared to both approaches from 4.2, the accuracy was improved – it reached 0.8422.

The confusion matrix reflects these changes. Now, 20 (0.7%) pictures were correctly identified as NC events, and likewise, 2559 (83.6%) were correctly identified as CC events. At the same time, 322 (10.5%) events were classified as NC even though they were CC, and likewise, 159 (5.2%) NC events were classified as CC.

Again, over-fitting was not the case, as the accuracy on the validation set usually remained higher than that on the training set.

4.3.2 More dense layers

The model had 37 376 parameters in total, 37 248 of which were trainable.

Given the results from section 4.2, the training was done on a batch size of 100 during four epochs – see Discussion for more information.

The loss was invariantly 0.6931. The accuracy was improved only as compared to the ν_μ-classifier trained on the smaller batch. It reached 0.6235.

The confusion matrix indicated that this time 100 (3.3%) pictures were correctly identified as NC events, and likewise, 1808 (59.1%) were correctly identified as CC events. At the same time, 257 (8.4%) events were classified as NC even though they were CC, and likewise, 895 (29.2%) NC events were classified as CC.

Again, over-fitting was not the case, as the accuracy on the validation set usually remained higher than that on the training set.

4.3.3 Noise

The two approaches to adding noise gave differing results. To understand them, let the models be presented. The processing of pictures with pre-added noise was done with the initial ν_μ-classifier described in 4.2 for the imbalanced data. The model with noise layers had 16 354 parameters in total, 16 226 of which were trainable.

As the complexity of the models did not increase as compared to the model trained on the greater batch in 4.2, the training in both cases was done on the batch of 100 for two epochs.

The loss was invariantly 0.6931. The accuracy dropped in both cases as compared to the ν_μ-classifier trained on the greater batch. For the case of pre-noised pictures, it reached 0.4895, while for the model with the noise layers, it reached 0.1144.

The confusion matrix for the former model indicated that this time, 168 (5.5%) pictures were correctly identified as NC events, and likewise, 1364 (44.6%) were correctly identified as CC events. At the same time, 192 (6.3%) events were classified as NC even though they were CC, and likewise, 1336 (43.7%) NC events were classified as CC. The confusion matrix for the latter model indicated that 0 (0%) pictures were correctly identified as NC events, and likewise, 2710 (88.6%) were correctly identified as CC events. At the same time, 350 (11.4%) events were classified as NC even though they were CC, and likewise, 0 (0%) NC events were classified as CC.

Over-fitting was not the case for the model with pre-noised pictures by contrast to the other model.

4.3.4 More dropouts

The model complexity did not chnage as compared to the initial ν_μ-clasifier in 4.2.

Given the results from section 4.2, the training was done on a batch size of 100 during four epochs – see Discussion for more information.

The loss was invariantly 0.6931. The accuracy was 0.6559.

The confusion matrix indicated that this time 94 (3.1%) pictures were correctly identified as NC events, and likewise, 1940 (63.4%) were correctly identified as CC events. At the same time, 263 (8.6%) events were classified as NC even though they were CC, and likewise, 763 (24.9%) NC events were classified as CC.

Again, over-fitting was not the case.

4.4 Extensions

4.4.1 Final state classifier

The model had 28 552 parameters in total, 28 424 of which were trainable.

Given the results from section 4.2, the training was done on a batch size of 100 during four epochs – see Discussion for more information.

The loss was 3.8953. The accuracy was 0.1520.

Due to the size of the confusion matrix (41 × 41), the readers are encouraged to see the full document in appendix B.

Over-fitting was not the case for the validation data.

4.4.2 Energy predictor

This section contains the results for the lepton data. Readers are encouraged to try the model for the neutrino data.

The majority of energies were less than 20 GeV, which shows the appropriate histogram – see appendix C.

The model had 26 935 parameters in total, 26 807 of which were trainable.

The training was done on a batch size of 100 during five epochs. The RAM limit made it impossible to run the model without the support from the local CPU.

The loss was 117.9006. The accuracy was also 117.9006.

A useful tool to complement the accuracy given by the root mean square measure of the model is histogram of errors - see appendix C. It indicates if the errors are skewed, which can be used to judge the overestimation.

Over-fitting was not the case for the validation data.

4.5 Metadata analysis

4.5.1 Binary classification

The accuracy on all the data for this particular training of the ν_μ-classifier was 0.3046.

The accuracy of classification of a dataset composed of "zero"s only was 0.7432 (better by 144%). The accuracy of classification of a dataset composed of "one"s only was 0.2514 (worse by 17%).

4.5.2 Energy of leptons

For $E_{\text{lepton}} \in [0, 10)$ GeV, the accuracy reached 0.4412 which was better by 4% than 0.4245 for all the data. For $E_{\text{lepton}} \in [10, 20)$ GeV, the accuracy reached 0.5503 which was worse by 2% than 0.5624 for all the data. For $E_{\text{lepton}} \in [20, 30)$ GeV, the accuracy reached 0.5263 which was better by 24% than 0.4245 for all the data. For $E_{\text{lepton}} \in [30, \infty)$ GeV, the accuracy reached 0.4800 which was better by 13% than 0.4245 for all the data.

4.5.3 Energy of neutrinos

For $E_{\text{neutrino}} \in [0, 10)$ GeV, the accuracy of prediction could not be got due to the RAM limitations. For $E_{\text{neutrino}} \in [10, 20)$ GeV, the accuracy reached 0.2702 which was better by 7% than 0.2533 for all the data. For $E_{\text{neutrino}} \in [20, 30)$ GeV, the accuracy reached 0.2697 which was better by 6% than 0.2533 for all the data. For $E_{\text{neutrino}} \in [30, \infty)$ GeV, the accuracy reached 0.2370 which was worse by 6% than 0.2533 for all the data.

4.5.4 Interaction type

In the considered data, only some of the interaction types were present. Hence, the discussion will be done not mentioning ν_τ and "ν on E scattering".

The changes in the accuracy were as follows: ν_μ CC QE: 13.08%, ν_μ CC Resonant: 6.87%, ν_μ CC DIS: 6.74%, ν_μ CC, other: 21.53%, ν_e CC QE: -36.56%, ν_e CC Resonant: -44.69%, ν_e CC DIS: -54.01%, ν_e CC, other: 7.14%, NC -73.72%, τ -90.14%.

See appendix D for more information.

4.5.5 Final state

See appendix E for all the information.

5 Discussion

It is the time to critically evaluate the performance of all the tested methods and tell those useful in application from those incapable of providing insight.

5.1 Binary classifiers

A number of architectures for the binary classifier delivered some indications for finding a better such classifier.

The reexamination of the Psihas model showed that the task involved working with a particularly nasty dataset. Despite having huge number of trainable parameters, the model assigned the same value to all the events. A decent but misleading accuracy of the classification of around 88% was yielded. If it had not been for the unevenness of the data, the results would have dropped dramatically. Overall, the model did not do its job on the considered dataset.

The same was true for the model with the noisy layers. Whether this is due to the intrinsic inapplicability of these layers to this particular task or to too much noise added is to be resolved.

Conclusion is that overcomplication of models and some noise types are not desirable.

A question that has possibly been answered was about the training mode.

Firstly, the training on a bigger batch and for fewer epochs yielded results more than 70% better than for the smaller batch and more epochs. This may be due to a particularly fortunate fixing of the model or the characteristic of the dataset that leads to over-training (hence, poorer results) when exposure is longer.

Secondly, balancing lifted the results of the model trained on a smaller batch. This may be explained by trading off the training focused on the frequency with the training focused on characteristics of data. The former promotes the detection of CC-events (which are more numerous) while the latter recognises NC better (which required better detection of the underlying patterns). Data does show it: the detection of the NC events (210 vs. 110) increased for the model trained on the smaller batch.

However, the trial and error development of the model trained on a greater batch (hard to summarise in this report) seems to be invariant to this observation. Its performance did not improve despite training it on a balanced dataset. Hence, the training with a larger batch size and fewer epochs was generally beneficial.

It was semi-predictable because of the results of the Psihas model that a model performed better if it had more trainable parameters. A distinction required to make is between the types of layers that were added: dense or convolution. The results suggest that the improvement is when bottlenecking is applied (with the caveat being that it reduced the accuracy of NC-events detection) as compared to simply adding more dense layers. This is ascribed to a better ability to detect underlying patterns in the not so graphically rich data.

When it comes to noising the pictures, and hence reducing sometimes-present over-fitting, only the pre-added noise approach was successful (i.e., it is recommended when creating an actual binary classifier). It is speculated that adding too much noise while training may be too big of a complication for the model to overcome.

Not completely counterintuitively to this observation, increasing the dropout layers (hence, adding randomness while training) turned out to be quite effective. It may be a safer and more effective option, as this kind of randomness just allows the network to retrain itself on the remaining data taking some features into account more (instead of random fluctuations that may not necessarily reflect the physics).

In conclusion, this section was to answer if the model is over-trained, needing randomisation, or does not detect the data patterns well enough. The best result was when convolution layers were introduced, but adding dropouts/pre-noised pictures also seems of use.

It should be noted that most of the networks performed appreciably better on the supplied data than the Psihas network which is of interest as they had far fewer trainable parameters.

5.2 Extensions

Given the conclusions from the previous section, the bottlenecking network was employed to the two extensions – identification of the final state and prediction of energy. This is because it performed best of all and focused on frequency-based training. The latter is thought to be especially beneficial as the information about the distribution of events is lost in other models (but it can be used to make predictions).

Sadly, this model was not enough to predict the final state with an overall accuracy higher than 15%. The central reason for that is possibly the high number of output classifications to reconcile (more than forty, while the network was developed for just two).

By contrast, the energy predictor could have an actual use in the research on neutrino interactions. As roughly 75% of the energies were predicted with the precision of ±10 GeV, the model could serve as an estimator of the order of energy for a particular interaction.

5.3 Metadata analysis

5.3.1 Binary classification

"Zero"s were classified superiorly better than "one"s in the simulation performed. As explained better, their energy might have been better captured by the model.

5.3.2 Energy of leptons

There is a lack of a clear pattern in the percentage changes of the accuracies. While the small energies tended to be captured slightly better than the average, those from the $[10, 20)$ GeV range were a struggle for the classifier. The unsteady increase for the higher energies seems puzzling and should be ascribed to the peculiarities of data and features that the model might have caught during the training.

5.3.3 Energy of neutrinos

By contrast to the "Energy of leptons" section, there is a downwards trend in the accuracies depending on the neutrino energies. It should be noted that it plateaus for the higher energies. As the rate of oscillation changes with energy, the oscillations may hinder the types of neutrinos involved in the interactions for the classifier.

5.3.4 Interaction type

Intuitively, the performance check with regard to interaction type can elucidate the neural network's mechanisms if it is considered in the categories of quasi-elastic/resonant/deep inelastic scattering/CC/NC interactions, but also flavours.

As it turned out, the latter were a good indication of the improvement in performance: for the ν_μ neutrinos, the ν_μ-classifier, unsurprisingly, performed (6%-13%) better than the reference.

On the other hand, the classification accuracy of the other neutrinos by the ν_μ-classifier was considerably lower: from 36% to 54% worse.

In these two cases, a mass dependence relation seems plausible. The scintillator is an organic molecule of a given mass, so a more massive particle has higher interaction energy (as the kinetic energy is proportional to mass), and hence, the lines are more vivid (especially in the case of clear pictures without shower)

The classification accuracies of NC and τ were decreased even more (by 73% and 90%, respectively). More analysis would be needed to entitle the author to speculate on these interactions.

For the ν_μ interactions exclusively, that the improvement is almost twice as big for the elastic interaction as for the semi-inelastic and inelastic ones may suggest that the clarity of the picture has higher importance for classification than the number of paths/showers.

For the ν_e interactions, a similar downwards trend was observed, which further corroborates the statement of the previous paragraph.

5.3.5 Final state

It is again useful to categorise the data into that corresponding to ν_μ, ν_e, and NC. The interplay between the numbers of showers and tracks will be a guiding tool.

For the first one, the most insightful result happened for the left red bars. Initially, for no track, it is seen that the more showers, the poorer the accuracy (drop from -2% to -47%). However, it was sufficient that one track appeared in the patterns – as the table was scrutinised toward the right – for the changes to get positive (even 16%). Later, as there were two and more tracks, the accuracies did not diverge much from the reference (between 0% to 11%).

Predictibly, it means that tracks are easily recognisable by the network as long as there are not too many of them. Moreover, they can counterbalance the "chaos" introduced by the showers.

Sadly, this only applied to ν_μ. When looking at the interactions of lighter neutrinos and those leading to NC, the lighter the particle, the more dominant the showers were.

For example, a drop from -16% to -100% for the ν_e when there are three or more tracks proved it. Though the last statement is suggested by the data (and the author refrains from speculating further on that topic), this may be profoundly misleading as the data for ν_e oftentimes counted a few pictures.

Regarding the final states of the NC particles, the data for zero and one tracks suggest the reversal of the principle derived for the ν_μ, i.e., this time, the more showers, the easier it is to get improved accuracies. The results restart to agree with the observations for ν_μ only for the cases with two or more tracks.

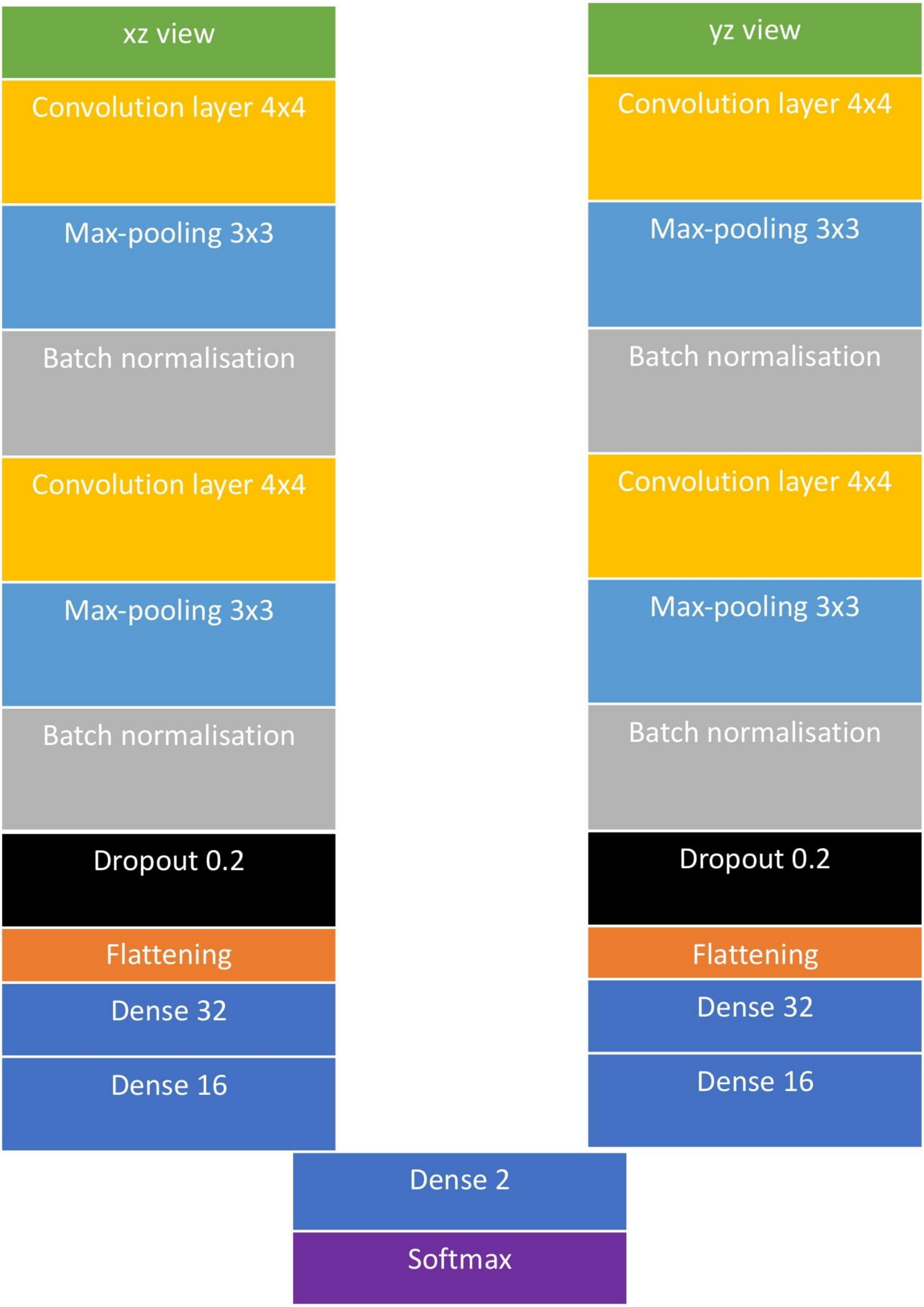
These characteristics could serve as a tool to predict the final state, and hence the resulting particles, as they differentiate ν_μ, ν_e, and NC relatively strictly.

6 Limitations and future work

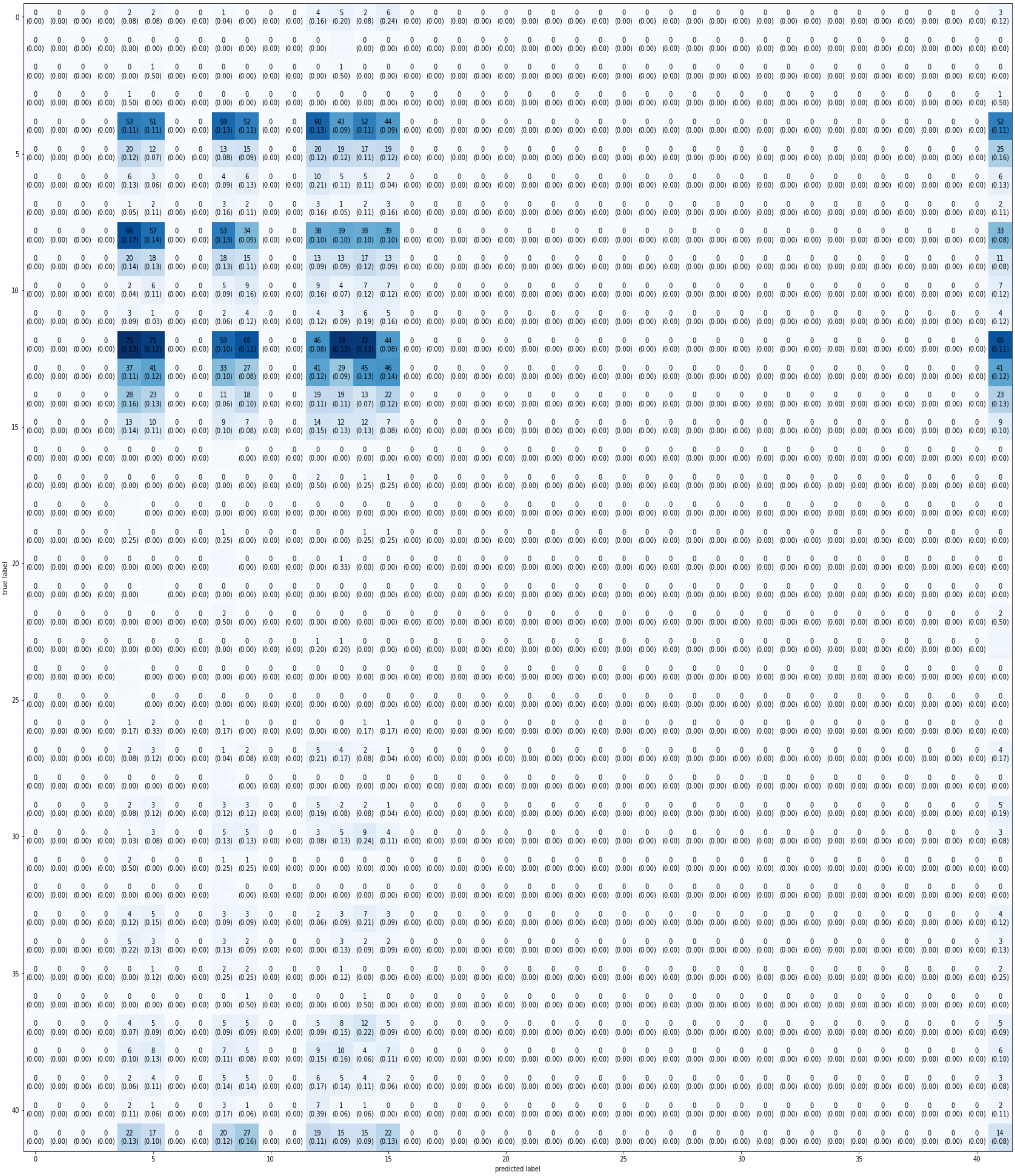
As indicated throughout this report, it is of hope that some of the approaches shown in the literature regarding the neutrino events classification have been challenged. For example, the architectures focused on frequency and pattern recognition were presented. The author wants to reemphasise that this work was limited in time and resources, e.g., by the processing capabilities of the software and hardware used. Therefore, future work in the area of neutrino events classification should aim to

- create neural networks balancing dropouts and bottlenecking more efficiently,
- prepare better and process more efficiently the supplied data,
- and for the final state prediction, consider some types of final states together the very same way the binary classification was envisioned.

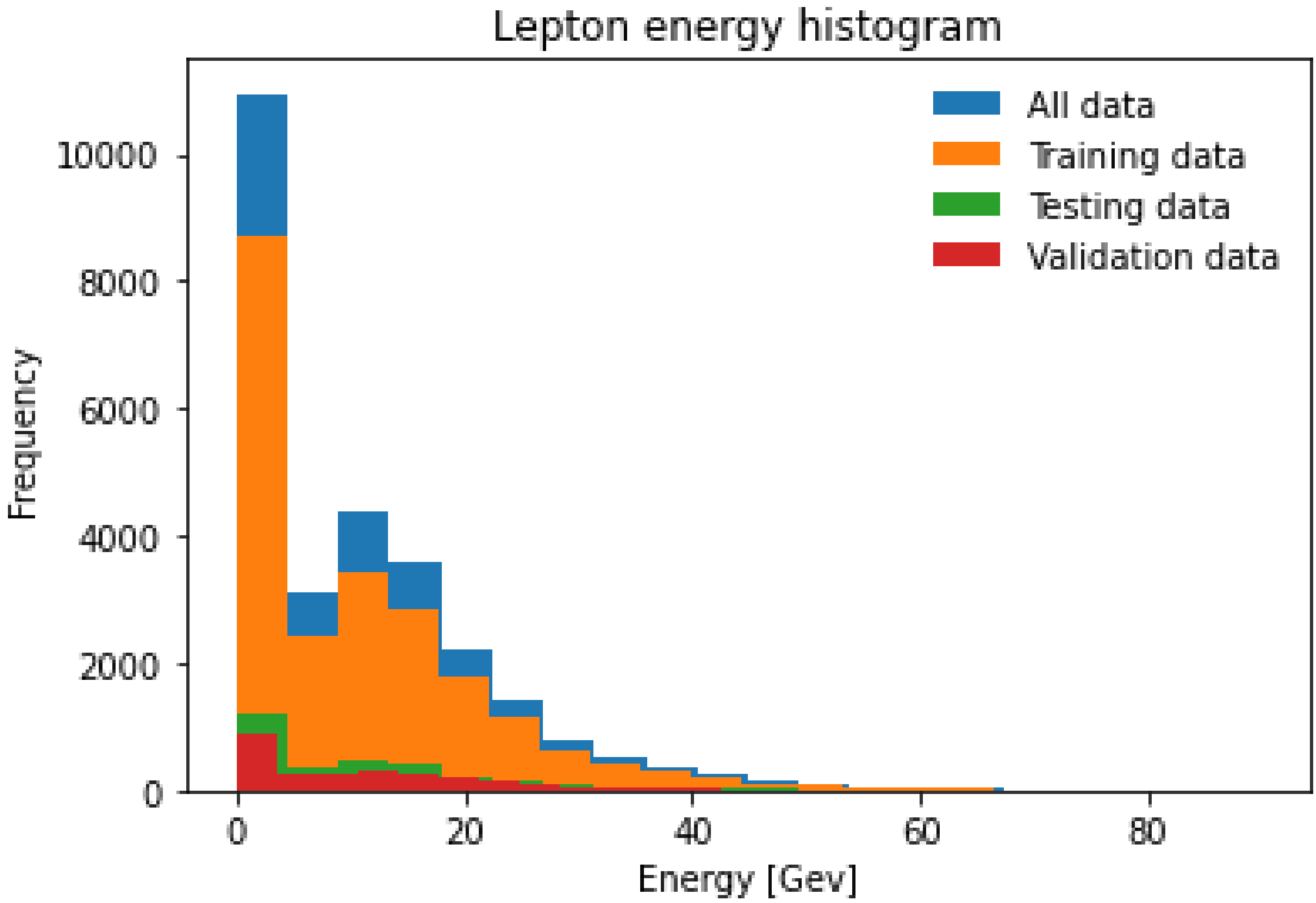
A Neural network scheme

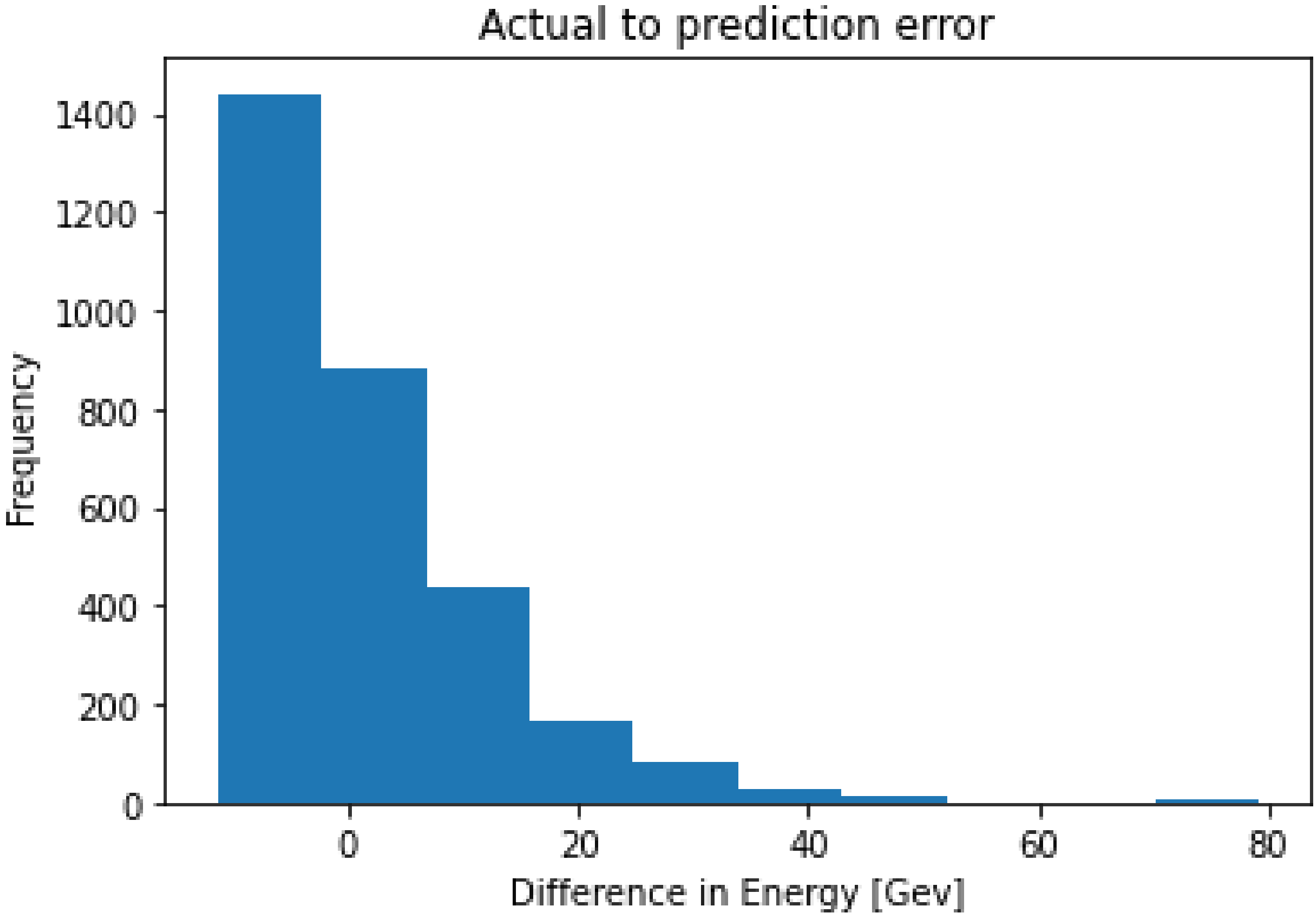


B Final state prediction – confusion matrix

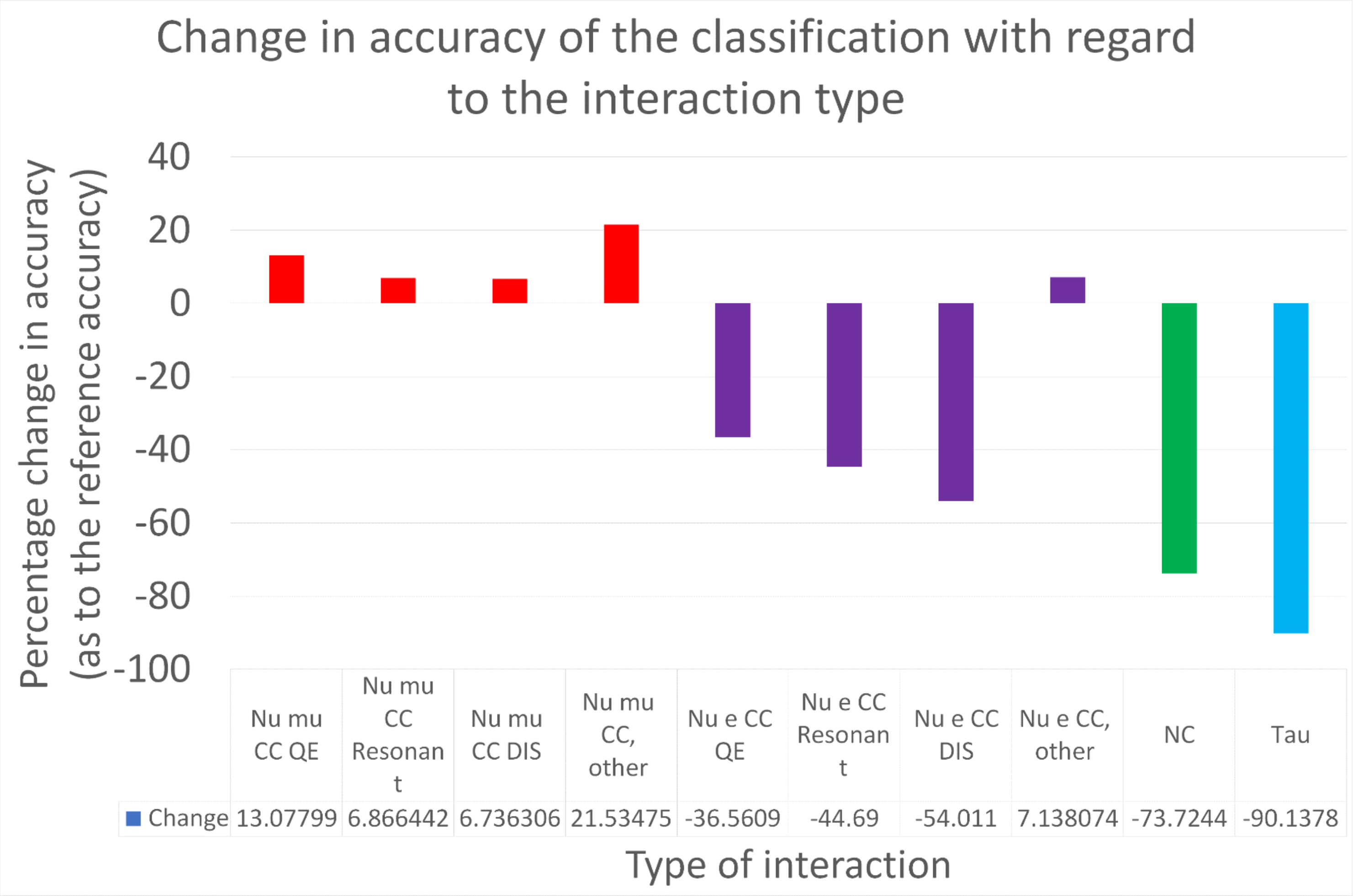


C Lepton energies



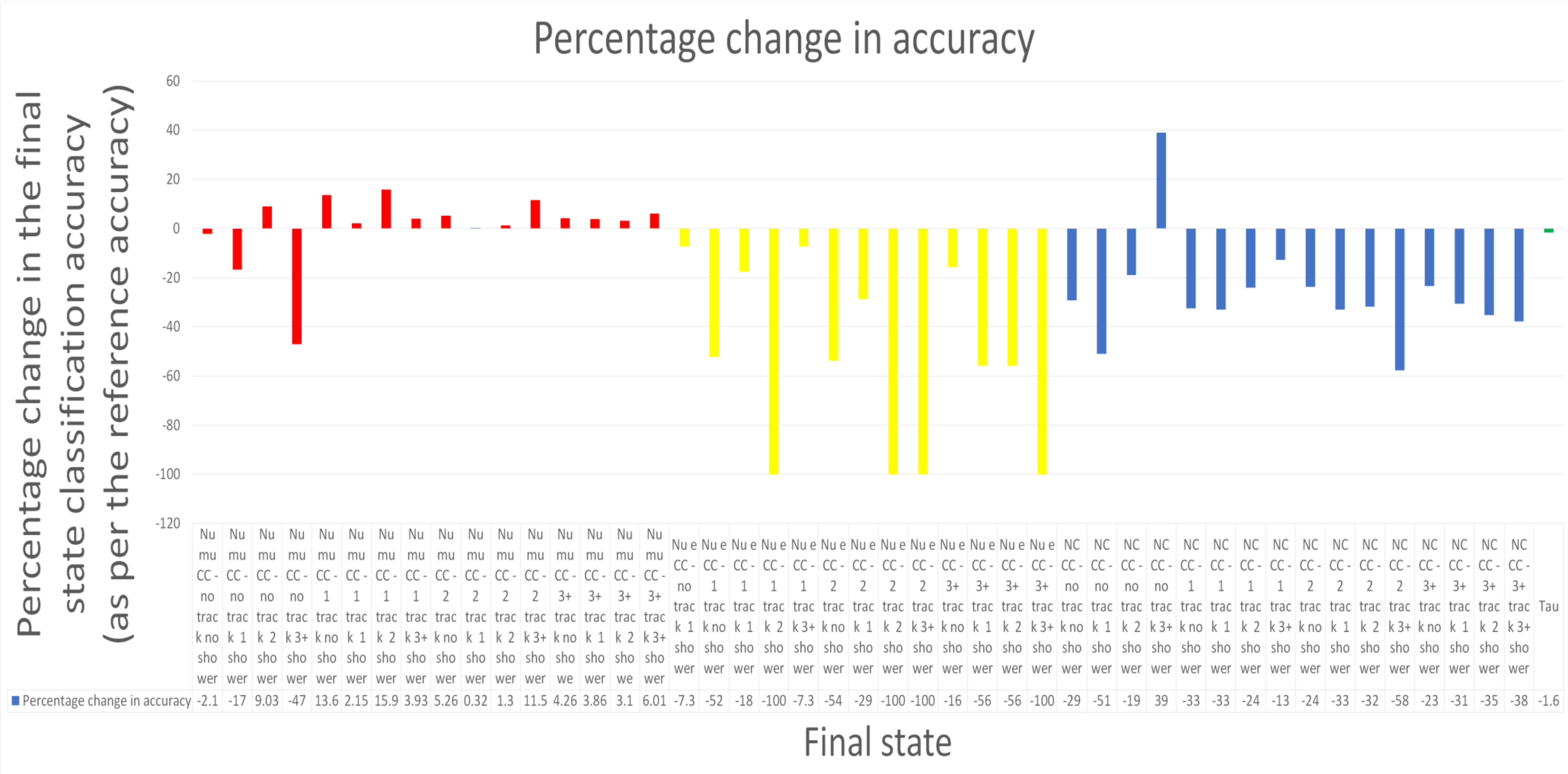


D Accuracy changes as per interaction type



	Nu mu CC QE	Nu mu CC Resonan t	Nu mu CC DIS	Nu mu CC, other	Nu e CC QE	Nu e CC Resonan t	Nu e CC DIS	Nu e CC, other	NC	Tau
🟦 Change	13.07799	6.866442	6.736306	21.53475	-36.5609	-44.69	-54.011	7.138074	-73.7244	-90.1378

E Accuracy changes as per final state



References

[1] Y. Fukuda, T. Hayakawa, E. Ichihara, K. Inoue, K. Ishihara, H. Ishino, Y. Itow, T. Kajita, J. Kameda, S. Kasuga, K. Kobayashi, Y. Kobayashi, Y. Koshio, M. Miura, M. Nakahata, S. Nakayama, A. Okada, K. Okumura, N. Sakurai, M. Shiozawa, Y. Suzuki, Y. Takeuchi, Y. Totsuka, S. Yamada, M. Earl, A. Habig, E. Kearns, M. D. Messier, K. Scholberg, J. L. Stone, L. R. Sulak, C. W. Walter, M. Goldhaber, T. Barszczxak, D. Casper, W. Gajewski, P. G. Halverson, J. Hsu, W. R. Kropp, L. R. Price, F. Reines, M. Smy, H. W. Sobel, M. R. Vagins, K. S. Ganezer, W. E. Keig, R. W. Ellsworth, S. Tasaka, J. W. Flanagan, A. Kibayashi, J. G. Learned, S. Matsuno, V. J. Stenger, D. Takemori, T. Ishii, J. Kanzaki, T. Kobayashi, S. Mine, K. Nakamura, K. Nishikawa, Y. Oyama, A. Sakai, M. Sakuda, O. Sasaki, S. Echigo, M. Kohama, A. T. Suzuki, T. J. Haines, E. Blaufuss, B. K. Kim, R. Sanford, R. Svoboda, M. L. Chen, Z. Conner, J. A. Goodman, G. W. Sullivan, J. Hill, C. K. Jung, K. Martens, C. Mauger, C. McGrew, E. Sharkey, B. Viren, C. Yanagisawa, W. Doki, K. Miyano, H. Okazawa, C. Saji, M. Takahata, Y. Nagashima, M. Takita, T. Yamaguchi, M. Yoshida, S. B. Kim, M. Etoh, K. Fujita, A. Hasegawa, T. Hasegawa, S. Hatakeyama, T. Iwamoto, M. Koga, T. Maruyama, H. Ogawa, J. Shirai, A. Suzuki, F. Tsushima, M. Koshiba, M. Nemoto, K. Nishijima, T. Futagami, Y. Hayato, Y. Kanaya, K. Kaneyuki, Y. Watanabe, D. Kielczewska, R. A. Doyle, J. S. George, A. L. Stachyra, L. L. Wai, R. J. Wilkes, and K. K. Young, *Phys. Rev. Lett.*, 1998, **81**, 1562–1567.

[2] C. Giunti and A. Studenikin, *Reviews of Modern Physics*, 2015, **87**(2), 531–591.

[3] Z. Maki, M. Nakagawa, and S. Sakata, *Progress of Theoretical Physics*, 1962, **28**(5), 870–880.

[4] F. Capozzi, E. Lisi, A. Marrone, D. Montanino, and A. Palazzo, *Nuclear Physics B*, 2016, **908**, 218–234.

[5] M. Aker, K. Altenmüller, M. Arenz, M. Babutzka, J. Barrett, S. Bauer, M. Beck, A. Beglarian, J. Behrens, T. Bergmann, U. Besserer, K. Blaum, F. Block, S. Bobien, K. Bokeloh, J. Bonn, B. Bornschein, L. Bornschein, H. Bouquet, T. Brunst, T. S. Caldwell, L. La Cascio, S. Chilingaryan, W. Choi, T. J. Corona, K. Debowski, M. Deffert, M. Descher, P. J. Doe, O. Dragoun, G. Drexlin, J. A. Dunmore, S. Dyba, F. Edzards, L. Eisenblätter, K. Eitel, E. Ellinger, R. Engel, S. Enomoto, M. Erhard, D. Eversheim, M. Fedkevych, A. Felden, S. Fischer, B. Flatt, J. A. Formaggio, F. M. Fränkle, G. B. Franklin, H. Frankrone, F. Friedel, D. Fuchs, A. Fulst, D. Furse, K. Gauda, H. Gemmeke, W. Gil, F. Glück, S. Görhardt, S. Groh, S. Grohmann, R. Grössle, R. Gumbsheimer, M. Ha Minh, M. Hackenjos, V. Hannen, F. Harms, J. Hartmann, N. Haußmann, F. Heizmann, K. Helbing, S. Hickford, D. Hilk, B. Hillen, D. Hillesheimer, D. Hinz, T. Höhn, B. Holzapfel, S. Holzmann, T. Houdy, M. A. Howe, A. Huber, T. M. James, A. Jansen, A. Kaboth, C. Karl, O. Kazachenko, J. Kellerer, N. Kernert, L. Kippenbrock, M. Kleesiek, M. Klein, C. Köhler, L. Köllenberger, A. Kopmann, M. Korzeczek, A. Kosmider, A. Kovalík, B. Krasch, M. Kraus, H. Krause, L. Kuckert, B. Kuffner, N. Kunka, T. Lasserre, T. L. Le, O. Lebeda, M. Leber, B. Lehnert, J. Letnev, F. Leven, S. Lichter, V. M. Lobashev, A. Lokhov, M. Machatschek, E. Malcherek, K. Müller, M. Mark, A. Marsteller, E. L. Martin, C. Melzer, A. Menshikov, S. Mertens, L. I. Minter, S. Mirz, B. Monreal, P. I. Morales Guzmán, K. Müller, U. Naumann, W. Ndeke, H. Neumann, S. Niemes, M. Noe, N. S. Oblath, H.-W. Ortjohann, A. Osipowicz, B. Ostrick, E. Otten, D. S. Parno, D. G. Phillips, P. Plischke, A. Pollithy,

A. W. P. Poon, J. Pouryamout, M. Prall, F. Priester, M. Röllig, C. Röttele, P. C.-O. Ranitzsch, O. Rest, R. Rinderspacher, R. G. H. Robertson, C. Rodenbeck, P. Rohr, C. Roll, S. Rupp, M. Ryšavý, R. Sack, A. Saenz, P. Schäfer, L. Schimpf, K. Schlösser, M. Schlösser, L. Schlüter, H. Schön, K. Schönung, M. Schrank, B. Schulz, J. Schwarz, H. Seitz-Moskaliuk, W. Seller, V. Sibille, D. Siegmann, A. Skasyrskaya, M. Slezák, A. Špalek, F. Spanier, M. Steidl, N. Steinbrink, M. Sturm, M. Suesser, M. Sun, D. Tcherniakhovski, H. H. Telle, T. Thümmler, L. A. Thorne, N. Titov, I. Tkachev, N. Trost, K. Urban, D. Vénos, K. Valerius, B. A. VanDevender, R. Vianden, A. P. Vizcaya Hernández, B. L. Wall, S. Wüstling, M. Weber, C. Weinheimer, C. Weiss, S. Welte, J. Wendel, K. J. Wierman, J. F. Wilkerson, J. Wolf, W. Xu, Y.-R. Yen, M. Zacher, S. Zadorozhny, M. Zbořil, and G. Zeller, *Phys. Rev. Lett.*, 2019, **123**, 221802.

[6] B. Pontecorvo, *Soviet Journal of Experimental and Theoretical Physics*, 1968, **26**, 984.

[7] M. A. Acero, P. Adamson, L. Aliaga, N. Anfimov, A. Antoshkin, E. Arrieta-Diaz, L. Asquith, A. Aurisano, A. Back, C. Backhouse, M. Baird, N. Balashov, P. Baldi, B. A. Bambah, S. Bashar, K. Bays, R. Bernstein, V. Bhatnagar, D. Bhattarai, B. Bhuyan, J. Bian, J. Blair, A. C. Booth, R. Bowles, C. Bromberg, N. Buchanan, A. Butkevich, S. Calvez, T. J. Carroll, E. Catano-Mur, B. C. Choudhary, A. Christensen, T. E. Coan, M. Colo, L. Cremonesi, G. S. Davies, P. F. Derwent, P. Ding, Z. Djurcic, M. Dolce, D. Doyle, D. D. Tonguino, E. C. Dukes, H. Duyang, R. Ehrlich, M. Elkins, E. Ewart, G. J. Feldman, P. Filip, J. Franc, M. J. Frank, H. R. Gallagher, R. Gandrajula, F. Gao, A. Giri, R. A. Gomes, M. C. Goodman, V. Grichine, M. Groh, R. Group, B. Guo, A. Habig, F. Hakl, A. Hall, J. Hartnell, R. Hatcher, H. Hausner, M. He, K. Heller, J. Hewes, A. Himmel, A. Holin, J. Huang, B. Jargowsky, J. Jarosz, F. Jediny, C. Johnson, M. Judah, I. Kakorin, D. M. Kaplan, A. Kalitkina, R. Keloth, O. Klimov, L. W. Koerner, L. Kolupaeva, S. Kotelnikov, R. Kralik, C. Kullenberg, M. Kubu, A. Kumar, C. D. Kuruppu, V. Kus, T. Lackey, K. Lang, P. Lasorak, J. Lesmeister, S. Lin, A. Lister, J. Liu, M. Lokajicek, S. Magill, M. M. Plata, W. A. Mann, M. L. Marshak, M. Martinez-Casales, V. Matveev, B. Mayes, D. P. Méndez, M. D. Messier, H. Meyer, T. Miao, W. H. Miller, S. R. Mishra, A. Mislivec, R. Mohanta, A. Moren, A. Morozova, W. Mu, L. Mualem, M. Muether, S. Mufson, K. Mulder, D. Naples, N. Nayak, J. K. Nelson, R. Nichol, E. Niner, A. Norman, A. Norrick, T. Nosek, H. Oh, A. Olshevskiy, T. Olson, J. Ott, J. Paley, R. B. Patterson, G. Pawloski, O. Petrova, R. Petti, D. D. Phan, R. K. Plunkett, J. C. C. Porter, A. Rafique, F. Psihas, V. Raj, M. Rajaoalisoa, B. Ramson, B. Rebel, P. Rojas, P. Roy, V. Ryabov, O. Samoylov, M. C. Sanchez, S. S. Falero, P. Shanahan, A. Sheshukov, P. Singh, V. Singh, E. Smith, J. Smolik, P. Snopok, N. Solomey, A. Sousa, K. Soustruznik, M. Strait, L. Suter, A. Sutton, S. Swain, C. Sweeney, A. Sztuc, R. L. Talaga, B. T. Oregui, P. Tas, T. Thakore, R. B. Thayyullathil, J. Thomas, E. Tiras, J. Tripathi, J. Trokan-Tenorio, A. Tsaris, Y. Torun, J. Urheim, P. Vahle, Z. Vallari, J. Vasel, P. Vokac, T. Vrba, M. Wallbank, T. K. Warburton, M. Wetstein, D. Whittington, D. A. Wickremasinghe, S. G. Wojcicki, J. Wolcott, W. Wu, Y. Xiao, A. Y. Dombara, A. Yankelevich, K. Yonehara, S. Yu, Y. Yu, S. Zadorozhnyy, J. Zalesak, Y. Zhang, and R. Zwaska, An improved measurement of neutrino oscillation parameters by the nova experiment, 2021.

[8] M. Honda, T. Kajita, K. Kasahara, and S. Midorikawa, *Phys. Rev. D*, 1995, **52**, 4985–5005.

[9] V. Barger, D. Marfatia, and K. Whisnant, *The physics of neutrinos*, Princeton University Press, 2012.

[10] R. N. Mohapatra and J. W. F. Valle, *Physical Review D*, 1986, **34**(5), 1642–1645.

[11] B. Denby, *Computer Physics Communications*, 1988, **49**(3), 429–448.

[12] J. J. Hopfield, *Proceedings of the National Academy of Sciences*, 1982, **79**(8), 2554–2558.

[13] F. Psihas, N. Collaboration, et al. In *Journal of Physics: Conference Series*, Vol. 898, p. 072053. IOP Publishing, 2017.

[14] L. Deng, *IEEE Signal Processing Magazine*, 2012, **29**(6), 141–142.

Finally, it is of the utmost importance to indicate that the core concepts used in this project stemmed from the material of the 2021/22 module "PHAS0056: Practical Machine Learning for Physicists" taught at UCL by Prof. Ryan Nichol.