Author:
Greer, Joel A T

Title:
Neutrino interaction identification for the DUNE trigger

Neutrino Interaction Identification for the DUNE Trigger

By

Joel Andrew Thomas Greer

School of Physics
University of Bristol

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of Doctor of Philosophy in the Faculty of Science.

March 2022

Word count: 43000

Abstract

The Deep Underground Neutrino Experiment (DUNE) will be a long baseline neutrino oscillation experiment using a high purity muon neutrino beam and near detector, both located at the Fermilab National Accelerator Laboratory, and a far detector hosted 1300 km downstream at the Sanford Underground Research Facility. The 10 kt fiducial mass of liquid argon will allow DUNE to have a rich off-beam neutrino physics programme, including the study of neutrino signals from core collapse supernovae.

The single phase DUNE far detector module will read out ionisation data at a rate of $1.2\,\mathrm{TB\,s^{-1}}$ whilst only a total data volume of 30 PB per year can be permanently stored. DUNE will make use of field programmable gate array (FPGA) resources in the front-end of the data acquisition (DAQ) as part of the necessitated triggering system.

This thesis presents a validation study of the FPGA-based trigger primitive generation (TPG) in the front-end DAQ using data collected by the ProtoDUNE experiment hosted at the European Council for Nuclear Research. The FPGA-based TPG was utilised as the first stage of a baseline supernova burst (SNB) trigger whose performance was evaluated using simulated neutrino interactions for a $11.2m_\odot$ progenitor star. The efficiency of the baseline SNB trigger was determined to have a lower limit of $97.7^{+0.2}_{-0.3}\%$ for supernovae at a distance 20 kpc, achieving the technical requirements set out for DUNE. To improve the performance of the SNB trigger at greater SNB distances, the use of a bounding box proposal network, You-Only-Look-Once version 3 (YOLOv3), was explored. This was found to improve the efficiency of the SNB trigger to $100^{+0.0}_{-0.0}\%$ up to the far side of the Milky Way galaxy and to $92.5^{+0.5}_{-0.5}\%$ at the Large Magellanic Cloud.

Dedication and acknowledgements

I would like to dedicate this thesis to my parents, who have always been a constant source of support, encouragement and inspiration. I would also like to thank my brother for showing me an excellent example to follow and my sister for keeping me sane throughout the years. Thanks are also due to my friends for their support and for pushing me to keep working hard.

Above all other collaborators, I would like to thank my supervisor, Dr. Jim Brooke for his invaluable advice and direction, his utmost patience and his very helpful feedback during the preparation of this thesis. I deeply appreciate the support given by my secondary supervisor, Prof. Jonas Rademacker, and all the members of the Bristol DUNE group through their valued discussion of this work. I would also like to thank the whole Bristol particle physics group for their useful discussions of particle physics, code, provision of computing resources and advice.

This work would not have been possible without the co-operation and open knowledge sharing by many of those who are part of the wider DUNE collaboration, including access to the computing resources at FNAL and CERN. This work was supported by the Fermi National Accelerator Laboratory, managed and operated by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for U.S. Government purposes.

I wish to thank Kunal Kothekar and Ivana Hristova who along with myself were responsible for the development and maintenance of the bitwise validation Python codebase during integration of FPGA-based TPG with ProtoDUNE. I would like to thank the entire firmware development team, in particular Konstantinos Manolopoulos and Kunal Kothekar for their explanations of the idiosyncracies of the TPG firmware and the origins of bugs throughout the development process.

I am grateful to Pierre Lasorak and the Sussex DUNE group who took the time to explain their work on SNB triggering using CPU-based hits and to Philip Rodrigues at Oxford who helped me understand how to use LArSoft to implement simulations of the TPG and provided some code which I customised to convert raw ADC data for the validation of hits from ProtoDUNE.

I thank Geoff Savage and the whole ProtoDUNE team at CERN with whom I enjoyed working to operate and collect data with the ProtoDUNE experiment. My thanks in particular go to Roland Sipos and Alessandro Thea at CERN for explaining to me how the FPGA-based TPG was set up for recording validation data in ProtoDUNE. I would also like to thank the UK Liason Office at CERN and Martisse Foster in Bristol who were absolutely essential in arranging accomodation and solving many of the other administrative matters relating to my LTA at CERN.

I wish to acknowledge the contributions of fellow Bristol PhD candidate Raul Stein who co-contributed to the codebase used to perform the studies on the YOLOv3 work and with whom I had many useful discussions.

I would like to thank Prof. Kate Scholberg for her provision of the SNB energy-time spectrum

which was used to determine the SNB triggering efficiency.

I would like to thank Prof. Joel Goldstein from the University of Bristol and Prof. Alex Tapper from Imperial College London for agreeing to examine this work.

As part of my studies as part of the STFC Data Intensive Centre for Doctoral Training I had a 6 month industrial placement for which I wish to thank DI CDT management board and in particular Rosemary Granger. I had the good fortune to spend the CDT placement at the STFC CCP-EM group with the helpful oversight of Martyn Winn, Tom Burnley and Jola Mirecka which was very enjoyable. I am also most grateful to Tom for his understanding during the last months of my thesis write-up.

Finally, I wish to acknowledge funding of my studies through the STFC Data Intensive CDT.

Author's declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: Joel Andrew Thomas Greer DATE: 13/3/2022

Table of Contents

List of Tables

LIST OF FIGURES

FIGURE **Page**

Chapter 1

Introduction

The Standard Model (SM) of particle physics, consisting of 12 elementary fermions, 4 elementary gauge bosons and the scalar Higgs boson, has proven to be exceptionally accurate at explaining experimentally observed phenomena. However, the SM does not explain a number of observations, including neutrino oscillations [1][2] and the baryon asymmetry problem [3], where more matter than antimatter is observed in the universe. Other failures include the inability to explain experimental observations of gravity [4], dark matter [5] and dark energy [6].

The Deep Underground Neutrino Experiment (DUNE) is a long baseline neutrino oscillation experiment currently under construction which will attempt to address some of these failures. It will be hosted by the Fermi National Accelerator Laboratory (FNAL) and the Sanford Underground Research Facility (SURF). A high purity muon neutrino beam will be sent 1300 km through the Earth from Fermi National Accelerator Laboratory (FNAL) to a set of 4 far detector (FD) modules located 1.5 km underground at the Sanford Underground Research Facility (SURF). Each detector module will have a fiducial mass of 10 kt of liquid argon (LAr). The rate of disappearance of muon neutrinos and appearance of electron neutrinos as they travel is dependent on the baseline of the experiment and the energy of the neutrinos. By probing oscillations of neutrinos over a wide energy band, DUNE will be able to disentangle different contributions to the oscillation rate and provide measurements of neutrino oscillation parameters. A near detector, also based at FNAL will characterise the neutrino beam to avoid biases in the measurements. The main goal of DUNE is to make measurements which determine whether CP violation occurs in the lepton sector and determine the ordering of the neutrino mass hierarchy. Oscillations will also allow a measurement of the θ_{23} mixing angle.

The large volume of liquid argon (LAr) will make DUNE a useful tool in the search for proton decay signals which could provide evidence for grand unified theories [7]. DUNE will also be used to investigate neutrinos from non-beam sources, such as solar neutrinos and atmospheric neutrinos. A high priority goal is to be able to detect the neutrino signal from a core collapse supernova (SN) within the Milky Way with high efficiency. Core collapse supernovae (SNe) release 99% of their energy in a burst of neutrinos. The observation of this signal would allow insights to be made about the formation processes of neutron stars and black holes as well as having the potential to constrain neutrino properties.

A single far detector (FD) module will read out $1.2\,\mathrm{TB\,s^{-1}}$ of data but only 30 PB of data will be able to be stored per year. This means that DUNE will require a system to identify and record interactions in the detector which are of physics interest, such as neutrino interactions, or proton decay. This ‘trigger’ must be able to keep up with the online data collection rate and maintain as high an efficiency as possible whilst keeping background rates below the permitted thresholds. Once triggered upon, collected data can be analysed by more sophisticated algorithms offline. The trigger system in DUNE will utilise field programmable gate array (FPGA) resources for front-end readout and processing to allow data processing to keep up with the input data rate. Subsequent algorithms implemented via central processing unit (CPU) resources will allow each module to trigger independently, collecting data which is then stored for further filtering and offline analysis.

1.1 Thesis Overview

This thesis describes validation of data processing algorithms implemented in the front-end FPGA hardware and studies of SNB triggering in the DUNE FD. The validation studies culminated in the validation of the FPGA data processing algorithms when implemented in the ProtoDUNE detector. An analysis of a baseline supernova burst (SNB) trigger algorithm which utilised these data processing algorithms is then presented. This is followed by studies which made use of a machine learning algorithm, YOLOv3, to attempt to improve the performance of the SNB trigger.

Firstly, the physics of neutrino oscillations and the motivation for their study are described in chapter 2. The interactions of neutrinos in a liquid argon time projection chamber (LArTPC), such as DUNE, are introduced followed by an introduction to core collapse SNe and a description of the neutrino signal which has been used for the SNB trigger studies presented.

In chapter 3 DUNE is introduced. The FD detector and DAQ system, including the functional blocks studied in validation work, are described. The technical requirements for a SNB are set out, followed by the baseline SNB trigger algorithm. ProtoDUNE is also introduced, including the setup in which it was used to validate the FPGA-based data processing algorithms. These validation studies are described in chapter 4.

An analysis of the performance of the baseline SNB trigger is presented in chapter 5. Details of data simulation are described and the objects generated at each stage of the trigger are characterised before the computed SNB trigger efficiency is presented.

Chapter 6 explains the theory behind machine learning (ML) including convolutional neural networks (CNNs) and sets out the YOLOv3 network whose implementation as part of the trigger chain was studied. These studies are presented in chapter 7 and indicate whether this ML algorithm could improve upon the baseline SNB trigger algorithm by more accurately identifying the signals from the low energy neutrino interactions characteristic of a SNB. The possible areas where ML could be introduced into the DAQ are discussed and the reasons behind the choice of YOLOv3 are communicated. The preparation of data and choices of hyperparameters to train the model are set out. The training metrics are evaluated followed by an analysis of the network predictions and the impact that the use of the YOLOv3 network can make to a SNB trigger.

Chapter 8 looks back over the main findings presented in this thesis. The outcomes of the studies are summarised, including the performance of the baseline and ML approaches to a SNB trigger. Recommendations are made for areas of future study.

Chapter 2

Physics

The SM treats neutrinos as massless and therefore does not explain the observations of neutrino flavour oscillation for which Kajita and McDonald were awarded the 2015 Nobel Prize in Physics. The SM also fails to explain the baryon asymmetry problem, where more matter than antimatter is observed in the universe. This asymmetry may have been produced in the early universe if the three Sakharov conditions [3] required for a baryon-generating interaction to produce matter and antimatter at different rates from an initial balanced state are satisfied. The first of these is baryon number violation, resulting in the creation of more baryons than antibaryons. The second is that charge conjugation (C) and charge conjugation parity (CP) symmetry must both be violated in order for processes which produce an excess of baryons to occur at a different rate than complementary processes which produce an excess of antibaryons. The final condition is that there must be a departure from thermal equilibrium. The first two Sakharov conditions can be tested within the SM of particle physics.

The SM is observed to have charge, parity and time reversal (CPT) symmetry, though C, P and CP (and therefore T) symmetry violation have been observed individually. To date no baryon number violation has been experimentally observed and, though theoretically possible, no significant CP violation due to strong force interactions has been observed either.

In 1957 parity violation in weak interactions was first detected in beta decay of ^{60}Co nuclei [8]. This was followed by the detection of indirect CP violation in kaon decays in 1964 [9]. Direct CP violation has since been observed in kaon decays [10] [11], B mesons [12] and D mesons [13]. However, the CP violation in the quark sector is not significant enough to explain the matter asymmetry problem [14]. The T2K experiment has shown hints that CP violation may occur in the lepton sector but the results do not yet constitute a discovery [15].

The first of DUNE's primary science goals is to detect oscillations of beam-produced neutrinos.

This will allow measurements of the CP phase term and allow the neutrino mass hierarchy to be determined. Oscillations will also allow a measurement of the θ_{23} mixing angle. The large active detector volume and use of LAr as a detector medium will also allow searches for proton decay, a secondary main science goal. Proton decay may provide evidence for Grand Unified Theories of physics which break baryon number symmetry and satisfy the first Sakharov condition. The final main goal for DUNE is to measure the neutrino flux from a core collapse supernova (SN) within our galaxy or its satellites. These neutrinos would provide constraints on neutrino properties and valuable insights into the internal mechanisms of SNBs to allow the validation of theoretical models.

In this chapter neutrino oscillations and the oscillation matrix are introduced, along with existing measurements. The measurement of δ_{CP} and the determination of the mass hierarchy in DUNE is then described, followed by the physics of particle interactions in LAr. The DUNE oscillation physics projections are then presented before a discussion of SNB physics and the signals they are expected to produce in DUNE.

2.1 Neutrino Physics

Neutrinos were postulated as massless particles by Fermi [16] to solve the problem of the continuous β-decay energy spectrum observed by Chadwick [17]. The first observations of neutrinos produced by the Savannah River nuclear reactor were made in 1956 via inverse β-decay [18]. The discovery of these (anti) electron neutrinos preceded the subsequent discovery of muon neutrinos in 1962 at the Brookhaven National Laboratory [19] and the later observation of the tau neutrino by the 'Direct observation of the nu tau' (DONUT) experiment in 2001 [20]. The Z mass and width provide experimental evidence which suggests that the electron, muon and tau neutrinos are the only 3 active flavours [21] but it is possible for there to be additional 'sterile' neutrino flavours which do not interact via the weak, strong or electromagnetic forces. In the case that sterile neutrinos do exist, it is possible to extend the SM to account for neutrino oscillations via the seesaw mechanism [22].

In this section the history of neutrino oscillation measurements is introduced, followed by the parameterisation of neutrino oscillations and an explanation of the parameters involved. Next, the current best fit values for these parameters are presented followed by the expected sensitivity of DUNE.

2.1.1 History of Neutrino Oscillations

Neutrino oscillations were anticipated by Pontecorvo who devised the first phenomenological model of neutrino oscillations in 1967 [23], predicting that if the oscillation length was smaller than the radius of the sun, the flux of electron neutrinos from the Sun would be half that of the total neutrino flux (assuming only two neutrino flavours). The first indication of oscillations was in the Homestake experiment, designed to measure the flux of electron neutrinos produced in the Sun. Theoretical calculations of this using standard solar models (SSMs) [24] predicted a large flux of electron neutrinos from fusion processes in the stellar core, with the majority of these having sub-MeV energies. The Homestake experiment was mainly sensitive to neutrinos produced from

$$^{8}\mathrm{B} \rightarrow {}^{8}\mathrm{Be} + e^{+} + \nu_{e} \tag{2.1}$$

decays which have an average energy of around 7 MeV but only form 10^{-4} of the solar neutrino flux. The neutrinos were detected using a large tank of tetrachloroethylene via the charged current (CC) interaction

$$\nu_{e} + {}^{37}\mathrm{Cl} \rightarrow e^{-} + {}^{37}\mathrm{Ar}, \tag{2.2}$$

where the subsequent radioactive decays from the ^{37}Ar atoms were observed. The observed neutrino flux was around 30% of that estimated by the SSM over the operation of the detector from 1970 to 1998 [25]. Subsequent gallium-based detectors, the Gallium Experiment [26] and the Soviet-American Gallium Experiment [27] were sensitive to ~0.23 MeV neutrinos produced in the

$$p + p \rightarrow d + e^{+} + \nu_{e} \tag{2.3}$$

deuterium-producing fusion reaction (which produces 90% of solar neutrinos). They measured electron neutrino fluxes 60-70% of that expected in the absence of oscillations.

The mystery of this 'solar neutrino deficit' persisted until the Sudbury Neutrino Observatory (SNO) experiment published results in [2]. SNO was a Cerenkov light detector filled with 1000 t of D_2O. Whilst SNO had an energy threshold only capable of detecting neutrinos from the reaction shown in equation 2.1, the use of D_2O as a detector medium allowed detection of the CC reaction sensitive only to electron neutrinos:

$$\nu_{e} + d \rightarrow p + p + e^{-}, \tag{2.4}$$

as well as the neutral current (NC) interaction

$$\nu_{x} + d \rightarrow n + p + \nu_{x}, \tag{2.5}$$

where x denotes any lepton flavour. The cross section for equation 2.5 is independent of neutrino flavour. The additional reaction

$$\nu_{x} + e^{-} \rightarrow \nu_{x} + e^{-} \tag{2.6}$$

is the elastic scattering (ES) reaction which is primarily sensitive to electron neutrinos via the exchange of either a W or a Z boson, but also has sensitivity to the muon and tau neutrinos via Z

boson exchange. The flux of the CC reaction was measured and compared to the flux from NC interactions. The ES flux was observed to be consistent with that observed by the Homestake detector, providing further evidence for neutrino flavour oscillation.

In addition to electron neutrino oscillations, strong evidence for muon neutrino oscillations was detected by the Super-Kamiokande (SK) experiment in 1998 [1]. Filled with 50 kt of water, Super-Kamiokande (SK) was another Cerenkov light detector, capable of detecting the characteristic signals produced when electron or muon neutrinos with energies greater than ~1 GeV interact with the water nuclei to produce an electron or muon. Pions created from the interaction of a cosmic ray proton with nuclei in the upper atmosphere decay to produce electron and muon neutrinos. SK was able to detect the azimuthal angle (and therefore the distance, L which the neutrino had passed through the Earth before detection) and the energy, E, of the interaction signals. When plotting the proportion of muon neutrinos observed to muon neutrinos expected, ν_μ^o/ν_μ^e, as a function of L/E it was observed that ν_μ^o/ν_μ^e dropped to around 0.5 for $L/E = 10^3$ km GeV^{-1}, a clear indication of muon neutrino oscillation. The SK detector was also used in the first long baseline neutrino experiment, KEK to Kamioka (K2K), where 112 beam neutrinos were detected from an expected 158±9 [28].

2.1.2 Neutrino Oscillations

Experiments including those described in section 2.1.1 have produced evidence for oscillations between the 3 active neutrino flavours, ν_e, ν_μ and ν_τ. These interactions can proceed via a weak CC interaction through W boson exchange. In order to oscillate between flavours, a neutrino flavour eigenstate, ν_α must be composed of a superposition of mass eigenstates, ν_i and can be expressed as such through the Pontecorvo-Maki-Nakagawa-Sakata (PMNS) matrix, U [29]:

$$|\nu_\alpha\rangle = \sum_{i=1}^{3} U_{\alpha i} |\nu_i\rangle . \tag{2.7}$$

The PMNS matrix is a 3x3 unitary parametrised by 4 free parameters. These are the 3 neutrino mixing angles, θ_{ij} where $i < j$ and the CP-violating phase δ_{CP} and are expressed as:

$$U = \begin{pmatrix} U_{e1} & U_{e2} & U_{e3} \\ U_{\mu 1} & U_{\mu 2} & U_{\mu 3} \\ U_{\tau 1} & U_{\tau 2} & U_{\tau 3} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_{23} & s_{23} \\ 0 & -s_{23} & c_{23} \end{pmatrix} \begin{pmatrix} c_{13} & 0 & s_{13}e^{-i\delta} \\ 0 & 1 & 0 \\ -s_{13}e^{i\delta_{CP}} & 0 & c_{13} \end{pmatrix} \begin{pmatrix} c_{12} & s_{12} & 0 \\ -s_{12} & c_{12} & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.8}$$

where c_{ij} is $\cos\theta_{ij}$ and s_{ij} is $\sin\theta_{ij}$.

The probability that a neutrino generated as a particular flavour, ν_α, oscillates into another, ν_β, as it travels through a vacuum is dependent on the ratio of the distance it has travelled, L, to its energy, E_ν [30]:

$$\begin{aligned}\mathrm{P}(\nu_\alpha \to \nu_\beta) = \delta_{\alpha\beta} - 4\sum_{j>k} \Re(U_{\alpha j}U^*_{\beta j}U^*_{\alpha j}U_{\beta k})\sin^2\left(1.267\frac{\Delta m^2_{jk}[\mathrm{eV}^2]L[\mathrm{km}]}{E_\nu[\mathrm{GeV}]}\right) \\ +2\sum_{j>k}\Im(U_{\alpha j}U^*_{\beta j}U^*_{\alpha j}U_{\beta k})\sin\left(1.267\frac{\Delta m^2_{jk}[\mathrm{eV}^2]L[\mathrm{km}]}{E_\nu[\mathrm{GeV}]}\right),\end{aligned} \tag{2.9}$$

with the final term changing in sign for the case of antineutrino flavour transitions $\overline{\nu_\alpha} \to \overline{\nu_\beta}$. Evidently the transition probability also depends on the mass difference between the different mass eigenstates. As $\Delta m^2_{32} + \Delta m^2_{21} - \Delta m^2_{31} = 0$, neutrino oscillations in a vacuum are only parameterised by 6 free parameters.

Experiments sensitive to '1-2 sector', have measured solar neutrino oscillations (Homestake [25], GALLEX [26], SAGE [27], SK [1], SNO [2]) and reactor neutrinos (KamLAND [31]) to determine θ_{12} and Δm^2_{21}. The '2-3' sector has been measured via atmospheric neutrino oscillations (Kamiokande [32], SK, IMB [33], IceCube [34]) and long baseline neutrino beams (K2K [28], NOvA [35], OPERA [36]). θ_{13} and $|\Delta m^2_{31}|$ have been studied by reactor experiments (Chooz [37], Palo Verde [38], Daya Bay [39], Double Chooz [40], RENO [41]).

From these experiments it has been determined that $\Delta m^2_{21} \ll |\Delta m^2_{31}| \simeq |\Delta m^2_{32}|$. However, the sign of Δm^2_{32} is not yet known. If $m^2_1 < m^2_2 \ll m^2_3$, then the mass eigenvalues are said to follow the 'normal hierarchy'. If instead $m^2_3 \ll m^2_1 < m^2_2$, then they have an 'inverted hierarchy'.

A precise measurement of $P(\nu_\mu \to \nu_e)$ and $P(\overline{\nu_\mu} \to \overline{\nu_e})$ via a long baseline neutrino experiment would allow the remaining unknowns in the neutrino oscillation parameters to be constrained [42]. These are the octant of θ_{23}, δ_{CP}, the mass hierarchy and the independent determination of θ_{13}.

Neutrinos in this experiment would propagate through the Earth rather than a vacuum. This requires the effects of neutrinos interacting with matter to be taken into account in order to disentangle the contribution of δ_{CP} from that of matter effects [43] and allow a measurement of it. The asymmetry, A, between the oscillation probability of $P(\nu) \equiv P(\nu_\mu \to \nu_e)$ and $P(\overline{\nu}) \equiv P(\overline{\nu_\mu} \to \overline{\nu_e})$ is [44]:

$$\begin{aligned}A(\nu_\mu \to \nu_e) \equiv (P(\nu) - P(\overline{\nu})) \equiv \Delta P_1 + \Delta P_2 + \Delta P_3 \\ = 16\frac{a}{\Delta m^2_{31}}\sin^2\frac{\Delta m^2_{31}L}{4E_\nu}c^2_{13}s^2_{13}s^2_{23}(1-2s^2_{13}) \\ -4\frac{aL}{2E_\nu}\sin\frac{\Delta m^2_{31}L}{2E_\nu}c^2_{13}s^2_{13}s^2_{23}(1-2s^2_{13}) \\ -8\frac{\Delta m^2_{21}L}{2E_\nu}\sin^2\frac{\Delta m^2_{31}L}{4E_\nu}s_\delta c^2_{13}s_{13}c_{23}s_{23}c_{12}s_{12},\end{aligned} \tag{2.10}$$

where $s_\delta = \sin\delta_{CP}$ and $a = 2\sqrt{2}G_F n_e E_\nu$ is the matter effect factor. This is composed of the Fermi constant, G_F, and the electron density of the medium the neutrinos are travelling through, n_e. The 3 terms on the right hand side of the equation are referred to as ΔP_1, ΔP_2 and ΔP_3.

The ΔP_1 and ΔP_2 terms of equation 2.10 are proportional to the matter effect factor, a, which itself is proportional to E_ν. The third, ΔP_3, is proportional to s_δ which arises from CP violation only. In the few GeV energy range the degeneracy between the asymmetries from the matter effect and CP violation can be resolved for a baseline over 1200 km [45]. To observe how an increased baseline and ability to use a wide energy band of energies helps disentangle ΔP_1, ΔP_2 and ΔP_3, the values of $\Delta P_1/L$, $\Delta P_2/L$ and ΔP_3 as a function of L/E_ν can be plotted as in figure 2.1. The asymmetry terms all oscillate with zeros at $L/E = 2\pi n/\Delta m_{31}$ for $n = 0, 1, ...$ but have an envelope which differs in magnitude as L/E increases. $\Delta P_1/L$ has an envelope which decreases whilst $\Delta P_2/L$ has an envelope of constant magnitude. In contrast, ΔP_3 has an envelope which increases, allowing the CP violating term to be differentiated from the matter terms using a single detector as long as a sufficient bandwidth of E_ν can be probed.

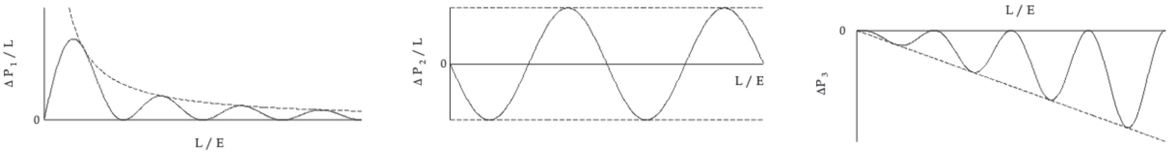


Figure 2.1: The matter effect terms ΔP_1 (left) and ΔP_2 (centre) both divided by the baseline, L of the neutrino oscillation experiment for the assumption that Δm^2_{21}, $c^2_{13}s^2_{13}s^2_{23}(1-2s_{13})^2 > 0$. The envelopes of these terms decrease and stay constant as a function of L/E. In contrast the CP violation term ΔP_3 is plotted on the right using the assumption that $s_\delta c^2_{13}s_{13}c_{23}s_{23}c_{12}s_{12} > 0$. It has an envelope which increases as a function of L/E. Reprinted with permission from [44][1].

2.1.3 Current Observed Oscillation Parameters

The data collected by the experiments listed in section 2.1.2 have been used by NuFIT to perform a global analysis [46]. The resulting best fit oscillation parameters are listed in table 2.1. Currently the normal mass hierarchy is favoured over the inverse hierarchy by only a 1.6σ significance.

The $\Delta\chi^2$ projections from NuFIT for the values of neutrino mixing parameters are shown in figure 2.2. for the normal (red) and inverted (blue) hierarchies. DUNE and other future neutrino experiments intend to further constrain the mixing angles, establish the neutrino mass hierarchy, constrain neutrino masses and mass differences further as well to establish whether CP violation occurs in the lepton sector.

2.1.4 Projected Sensitivity Of DUNE

DUNE will use a high purity muon neutrino beam which can be configured to produce neutrinos or antineutrinos over a wide energy band in the range of several GeV. This allows electron

[1] Copyright 1997 by the American Physical Society

[2] Image reproduced under Creative Commons 4.0 license `https://creativecommons.org/licenses/by/4.0/`

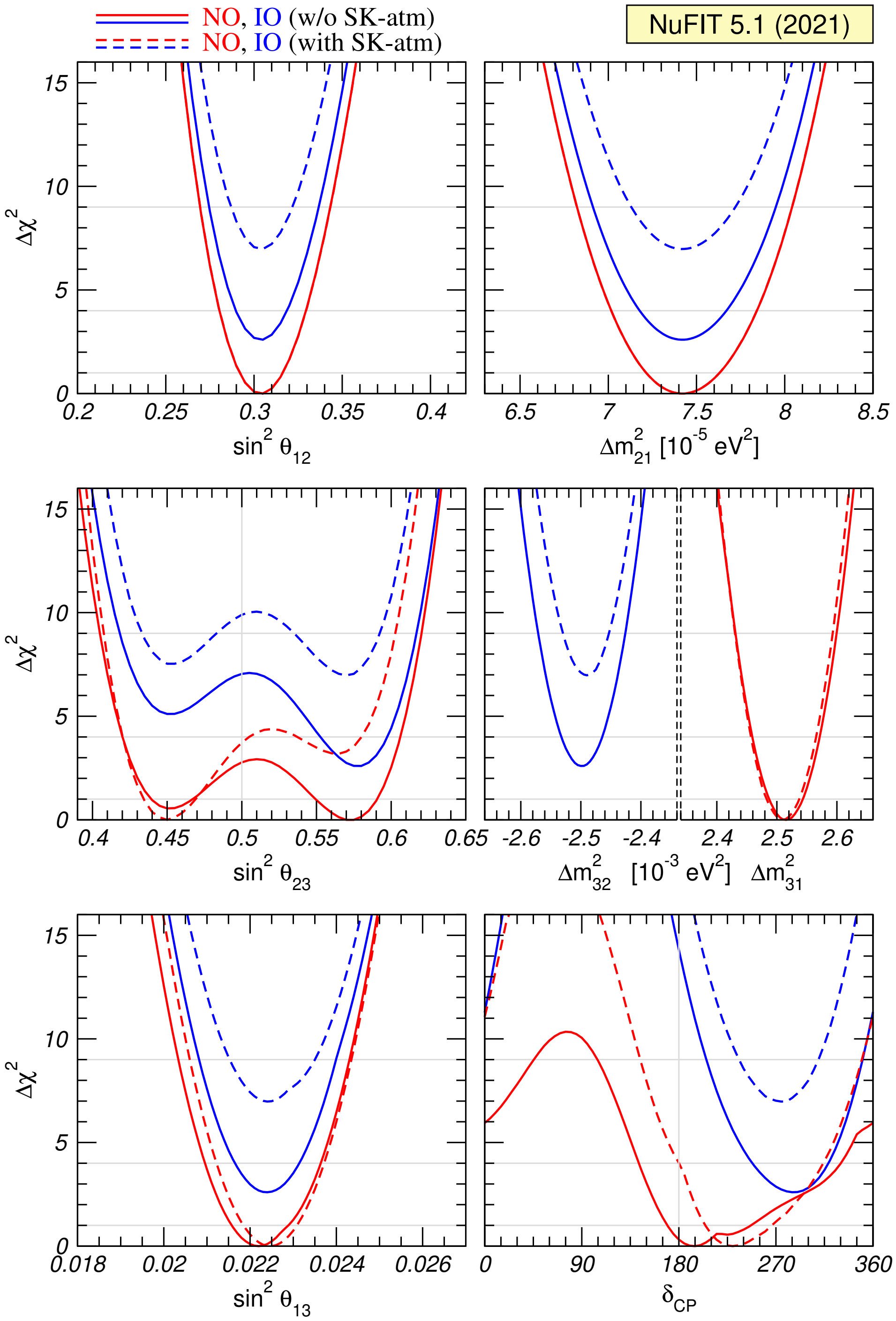


Figure 2.2: NuFIT 5.1 global 3ν oscillation analysis projections of $\Delta\chi^2$ for oscillation parameters from [46] $\Delta\chi^2$ is the confidence value corresponding to the chi-square quantile for the 6 degrees of freedom present in neutrino oscillations. The solid and dashed lines correspond to the projections when atmospheric neutrino observations from Super-Kamiokande are and are not taken into account respectively. The projections for the normal mass ordering (NO) and the inverted mass ordering (IO) are shown in red and blue respectively.[2]

Table 2.1: NuFIT 5.1 best fit for neutrino oscillation parameters [46] when including Super-Kamiokande atmospheric neutrino data. Values are quoted with 1σ error bars. $\Delta m^2_{3l} = \Delta m^2_{31} > 0$ for the normal mass hierarchy and $\Delta m^2_{3l} = \Delta m^2_{32} < 0$ for the inverted mass hierarchy.

Parameter	Normal hierarchy	Inverted hierarchy
$\sin^2\theta_{12}$	$0.304^{+0.012}_{-0.012}$	$0.304^{+0.013}_{-0.012}$
$\sin^2\theta_{23}$	$0.450^{+0.019}_{-0.016}$	$0.570^{+0.016}_{-0.022}$
$\sin^2\theta_{13}$	$0.02246^{+0.00062}_{-0.00062}$	$0.02241^{+0.00074}_{-0.00062}$
δ_{CP} (°)	230^{+36}_{-25}	278^{+22}_{-30}
$\frac{\Delta m^2_{21}}{10^{-5}\mathrm{eV}^2}$	$7.42^{+0.21}_{-0.20}$	$7.42^{+0.21}_{-0.20}$
$\frac{\Delta m^2_{3l}}{10^{-3}\mathrm{eV}^2}$	$+2.510^{+0.027}_{-0.027}$	$-2.490^{+0.026}_{-0.028}$

neutrino appearance and muon neutrino disappearance to be probed over a wide range of L/E_ν. It will have a baseline of 1300 km, which, in combination with the wide neutrino energy band, should allow mass and CP violation effects to be separated. Therefore DUNE should be capable of high sensitivity measurements of neutrino mixing parameters.

The sensitivity of DUNE to δ_{CP} is observed in figure 2.3, where the statistical significance expected to be gained from operation of DUNE for 7 years (purple) and 10 years (orange) are shown for different values of δ_{CP}. The smaller the CP violating phase is, the more difficult CP violation is to detect.

For the case of maximal CP violation sensitivity ($\delta_{CP} = -\pi/2$), less than a year of data collection will be required to achieve a 5σ measurement of the neutrino mass hierarchy. No matter what the value of δ_{CP} is, a 5σ measurement should take less than 3 years of collected data.

DUNE will also be sensitive to $\sin^2 2\theta_{23}$ using observations of $\nu_\mu \to \nu_\mu$ and to $\sin^2\theta_{23}$ through $\nu_\mu \to \nu_e$. The octant of θ_{23} is most difficult to detect near a value of $\sin^2\theta_{23} = 0.5$. After 10 years the octant should be detectable at 5σ if $\sin^2\theta_{23}$ is outside the range 0.46-0.58. An additional 5 years of data collection will improve the range of 5σ sensitivity by 0.01.

2.2 Neutrino Interactions In DUNE

DUNE will utilise LAr as a detector medium. The LAr will be held in a uniform electric field allowing any deposited ionisation to drift towards and be detected via instrumentation on the anode planes. This constitutes a liquid argon time projection chamber (LArTPC), first proposed

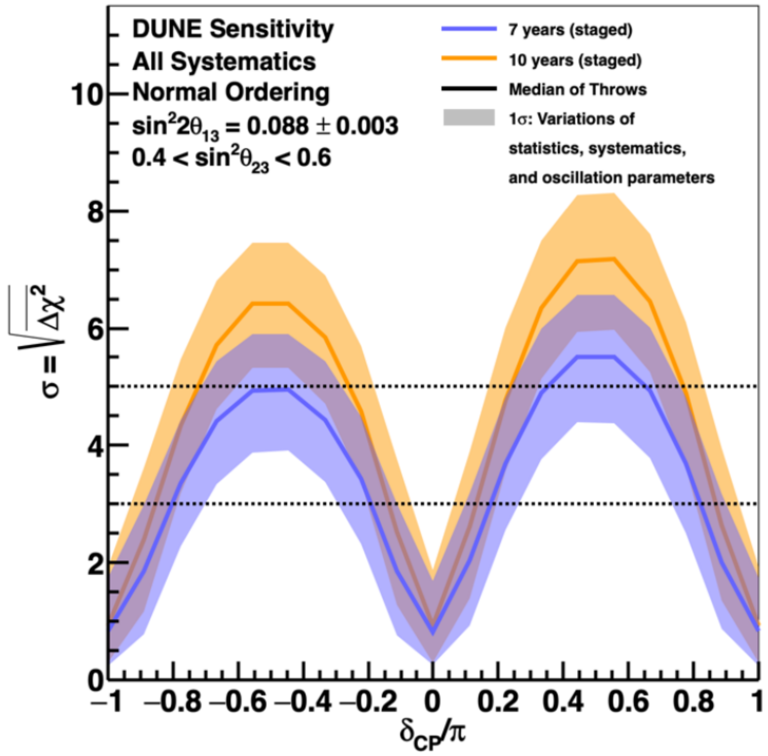


Figure 2.3: DUNE physics projections for sensitivity to CP violation as a function of the true value of δ_{CP}. Purple corresponds to 7 years of data collection and orange to 10 years. Uncertainties are quoted in 1σ error bands. Image reproduced from [47][3].

in 1977 [48] and is illustrated in figure 2.4. A full description of the detector readout on the anode planes can be found in section 3.2.

LAr constitutes an ideal detector medium for LArTPCs as a result of its properties. It has a high density, $\rho = 1.4\,\mathrm{g\,cm^{-3}}$, maximising the number of neutrino interactions, and has negligible electronegativity, so it does not attract or repel free electrons. It also has a high electron mobility, μ, which in typical LArTPC conditions of $E = 500\,\mathrm{V\,cm^{-1}}$ leads to a high drift velocity of $v_d = \mu E = 1.6\,\mathrm{mm\,\mu s^{-1}}$. Combined with a low transverse (and even lower longitudinal) diffusion constant of $D_T = 13.2\,\mathrm{cm^2\,s^{-1}}$ [49], a LArTPC is able to detect signals with a high resolution. By using multiple sets of sensing wires which are not parallel, measuring the signal on each over time, and detecting characteristic scintillation photons (which provide a t_0 for an interaction), the charge deposited in the 3D volume can be reconstructed. This can be converted to determine the energy loss of an ionising particle as it travels a distance x through the detector medium, dE/dx, which can be used for particle identification. In this section the production of signals in LArTPCs through ionisation and scintillation is described, followed by an explanation of the low energy neutrino interactions (<100 MeV) relevant to the work presented in this thesis.

[3] Image reproduced under Creative Commons 4.0 license `https://creativecommons.org/licenses/by/4.0/`

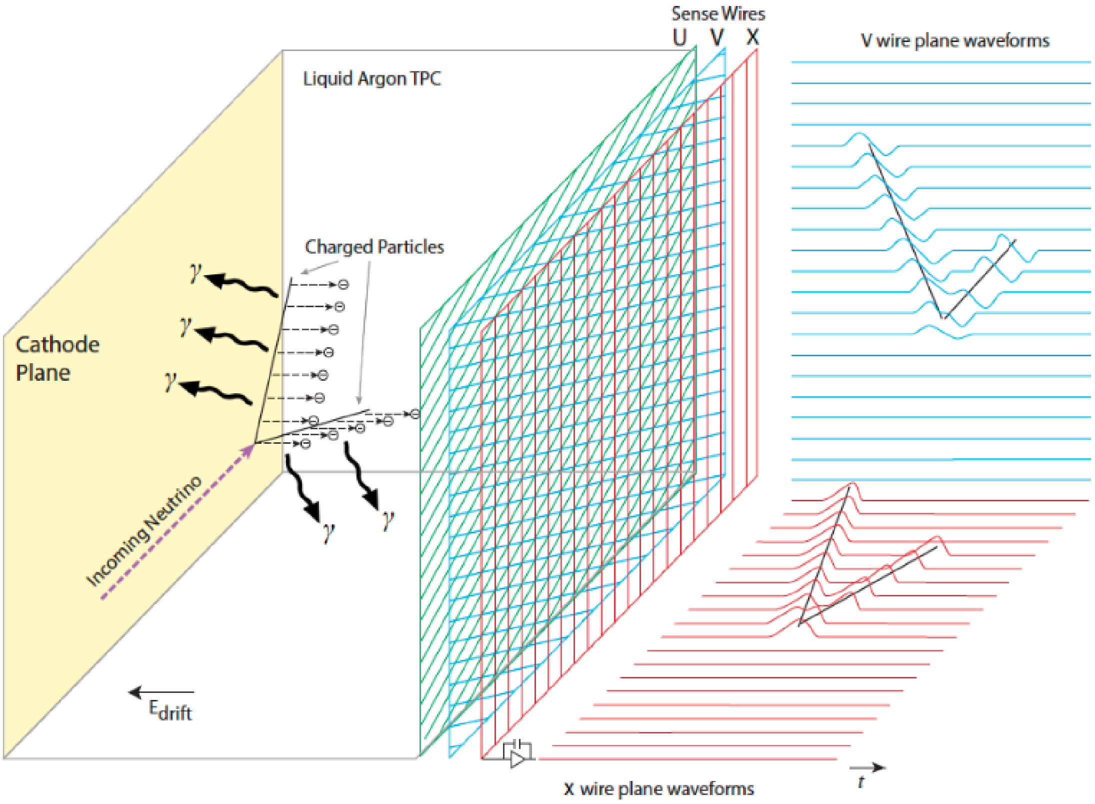


Figure 2.4: LArTPC volume with an applied electric field. Ionisation electrons drift towards the anode plane consisting of sensing wires, allowing the ionisation signals to be plotted as a function of time. Reproduced from [50][4].

2.2.1 Ionisation in a LArTPC

The mean rate of energy loss for moderately relativistic heavy charged particles due to interaction with the electrons bound to atoms in the absorbing medium is described by the Bethe-Bloch equation [51]:

$$\left\langle -\frac{dE}{dx}\right\rangle = Kz^2\frac{Z}{A}\frac{1}{\beta^2}\left[\frac{1}{2}\ln\frac{2m_ec^2\beta^2\gamma^2W_{max}}{I^2} - \beta^2 - \frac{\delta(\beta\gamma)}{2}\right], \tag{2.11}$$

where $K = 4\pi N_A r_e^2 m_e c^2$, a product of Avogadro's number, the classical electron radius, the electron mass and the speed of light. The charge of the interacting particle is z whilst Z and A are the atomic number and mass of the absorbing medium. β is the proportion of c that the particle is travelling at, γ is the Lorentz factor, W_{max} is the maximum energy transfer to an electron in a single collision and I is the mean excitation energy. The final term, $\delta(\beta\gamma)$ is a density correction term. A muon minimum-ionising particle (MIP) in LAr loses energy at an expected

[4] Image reproduced under Creative Commons 4.0 license https://creativecommons.org/licenses/by/4.0/

rate of $2.1\,\mathrm{MeV\,cm^{-1}}$ [49].

The deposited energy ionises atoms in the detector medium, resulting in the production of free electrons which drift if there is an applied electric field. In LAr, the average energy dissipation per electron-ion pair produced is $W_{ion} = 23.6\mathrm{eV}$ [52] and the number of produced ion pairs has root mean square error which is dependent on the Fano factor $F = 0.107$ [49][53]:

$$\text{r.m.s} = \sqrt{F \frac{E_0}{W_{ion}}} \tag{2.12}$$

where E_0 is the total energy of the ionising particle absorbed in the medium. The Fano factor takes into account the energy deposited via excitation but ignores the (negligible) nuclear scattering effects.

After production of ion pairs, a proportion immediately recombine. The efficiency of converting deposited energy into observable charge is known as the recombination factor, R_c and depends on the applied electric field and the density of the ionisation. It can be described by Birk's law [54]:

$$R_c = \frac{\frac{dQ}{dx}}{\frac{dE}{dx}} = \frac{A}{1 + \frac{k}{\epsilon}\frac{dE}{dx}} \tag{2.13}$$

where Q is the observable charge, $A = 0.8$, $k = 0.0486(\mathrm{g/MeVcm^2})(\mathrm{kV\,cm^{-1}})$ and $\epsilon = E_F\rho$, the product of the electric field and the density.

Whilst drifting to the anode plane, the electrons may be lost due to interactions with impurities, such as oxygen, water and nitrogen molecules, in the time projection chamber (TPC). The proportion of the ionisation charge which remains after drifting for a time t_d through the detector medium is:

$$Q(t_d) = Q_0 e^{-\frac{t_d}{\tau_e}} \tag{2.14}$$

where $\tau_e = 1/k_s N_s$ is the electron lifetime, a product of the rate of attachment of electrons to impurities, k_s, and the concentration of impurities in the detector, N_s.

The Bethe-Bloch equation describes energy loss for heavy charged particles but energy loss of electrons and photons in matter proceeds in a more complex manner. At energies lower than the 'critical energy', E_c, electrons and positrons lose energy primarily via ionisation [51]. Above E_c, bremstrahlung is the majority contribution to energy loss. For LAr, $E_c = 30.5\mathrm{MeV}$ [49]. The rate of energy loss via ionisation rises logarithmically with the electron energy whilst the rate of energy loss via Bremstrahlung rises linearly, causing it to be dominant at $> E_c$.

The main mechanism of energy loss for photons is also dependent on the particle energy. At low energies photons mainly interact via the photoelectric effect, which causes the emission of photoelectrons, and the Compton effect where the photon loses energy via inelastic scattering by an atomic electron. At higher energies, pair production ($\gamma \to e^- e^+$) becomes the primary mechanism. High energy electrons and photons can initiate electromagnetic cascades as pair production

[4] Image reproduced under Creative Commons 4.0 license `https://creativecommons.org/licenses/by/4.0/`

and bremstrahlung result in a shower of lower energy photons and electrons, eventually below the energies required for pair production.

2.2.2 Scintillation in a LArTPC

Scintillation in LAr is dependent on the formation and decay of diatomic excited molecules (excimers) of argon, Ar_2^*. This occurs via 2 processes, self-trapped exciton luminescence and recombination luminescence [55][56].

In the first of these processes, the excimer is generated through excitation of an argon atom allowing it to form an excimer state with another argon atom. This can then radiate a photon of characteristic wavelength and decay to the usual atomic argon state. Recombination luminescence requires an argon atom to be ionised and be in the presence of an electron cloud. This reaction depends on the applied electric field and the greater the recombination and light yield, the lower the ionisation yield. The processes are:

$$\text{Excitation}: \mathrm{Ar}^* + \mathrm{Ar} \rightarrow \mathrm{Ar}_2^* \rightarrow \mathrm{Ar} + \mathrm{Ar} + \gamma \tag{2.15}$$

for excitation and

$$\text{Recombination}: \mathrm{Ar}^+ + \mathrm{Ar} + e^- \rightarrow \mathrm{Ar}_2^* \rightarrow \mathrm{Ar} + \mathrm{Ar} + \gamma \tag{2.16}$$

for recombination.

The excimers are produced in singlet or a triplet states, with the weighting of the production of these states being different for the two reaction paths. The singlet state has a lifetime of 6 ns and the triplet state a lifetime of 1.6 µs [57], producing light with a wavelength of 128 nm. LAr is transparent to its own scintillation light, allowing it to be detected without attenuation. In the absence of an electric field the photon yield is $\approx 5 \times 10^4$ photons per MeV [56].

2.2.3 Neutrino Interactions in LAr

In the work presented in this thesis, interaction signals from low energy electron neutrinos were studied. At low energies (5-100MeV) nuclear interaction processes dominate, with neutrinos interacting with LAr via charged current interactions, elastic neutrino-electron scattering and neutral current excitation interactions [58]:

$$\nu_e + {}^{40}\mathrm{Ar} \rightarrow e^- + {}^{40}\mathrm{K}^*, \tag{2.17}$$

$$\overline{\nu_e} + {}^{40}\mathrm{Ar} \rightarrow e^+ + {}^{40}\mathrm{Cl}^*, \tag{2.18}$$

$$\nu_x + e^- \rightarrow \nu_x + e^- \tag{2.19}$$

$$\nu_x + \mathrm{Ar} \rightarrow \nu_x + \mathrm{Ar}^*. \tag{2.20}$$

Equations 2.17 and 2.18 possess Q-values of 1.5MeV and 7.48MeV respectively. The first excitation level of ^{40}Ar is 1.46MeV, ensuring that the neutral current excitation interaction also has a low threshold similar to that of $\nu_e{}^{40}$Ar CC [59].

The ν_e component in equation 2.19 is greater than the $\nu_{\mu,\tau}$ components due to the additional charged current component and the cross sections for ES increase linearly with the neutrino energy. If neutrinos are expected to come from a point source, such as a supernova, ES interactions may allow the location of the supernova in the sky to be determined [60].

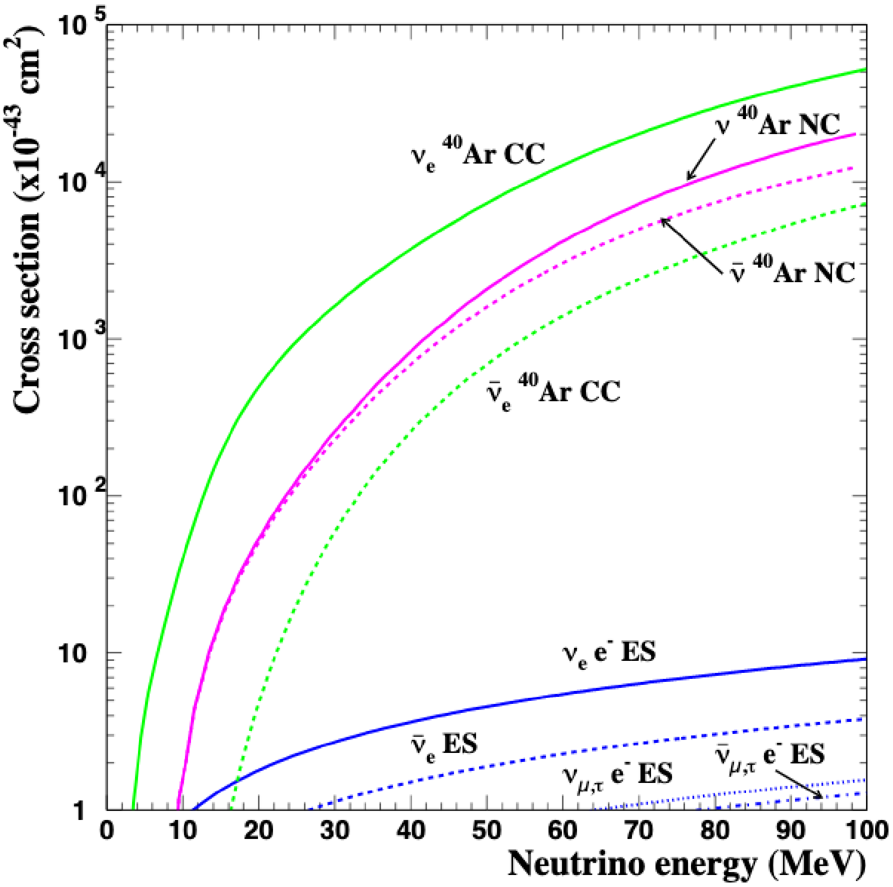


Figure 2.5: Neutrino cross sections for low energy interactions in LAr. The dominant cross section is ^{40}Ar charged current. Reproduced with permission from [59][5].

The cross section for each of these interactions is shown in figure 2.5, where it can be seen that the $\nu_e{}^{40}$Ar CC interaction is dominant, particularly at low energies. Calculated cross sections (with the exception of ES) are expected to have errors of 10-20% [58] but no direct measurements of the cross sections within this energy range yet exists.

[5] Copyright 2003 by IOP Publishing. Reproduced with permission. All rights reserved. https://doi.org/10.1088/1475-7516/2003/10/009.

2.3 Supernova Burst Physics

Supernovae are powerful explosions which result in the end of the life of some stars. There are a number of different types of SN, differentiated by spectroscopic characteristics and their light curves. For the purposes of studying the neutrino flux, the main subjects of interest are SN which proceed via an iron core collapse. This SN mechanism is the most common, applying to stars whose mass is at least 9 solar masses ($\gtrsim 9_m \odot$).

In a core collapse SN, the star loses 99% of its gravitational binding energy ($2-4\times10^{46}$ J) through emission of $\sim 10^{58}$ neutrinos [30]. Studies have estimated the expected rate of SNe in the Milky Way to range from 1.2 to 8.4 per 100 years, with most estimates being towards the lower value [61][62][63][64].

Since the advent of large neutrino detection experiments, a neutrino signal has only been detected from the SN1987A supernova event which occurred in the Large Magellanic Cloud (LMC) at a distance of 50 kpc. 25 neutrinos with energies consistent with a core collapse SN (5-50 MeV [65]) were detected by the Kamiokande-II [66], Irvine-Michigan-Brookhaven [67] and Baksan [68] experiments over a timescale consistent with a core collapse SN. These neutrinos have been used to verify the basic theoretical process of a core collapse SN and provide a number of constraints on physics properties including the energy loss and temperature of the SN [69]. Other studies have used them to set constraints on beyond the standard model physics signals [70], neutrino masses [71][72], the electron neutrino magnetic moment [73] and neutrino electric charge [74].

In this section the core collapse SN dynamics are described. The spectral form of the neutrino flux will be introduced along with a discussion of how it can change due to neutrino oscillations. This will be followed by a discussion of the parameters which detection of neutrinos from SNe can help to determine. Finally, the energy-time neutrino spectrum used in analyses in this thesis will be introduced and characterised.

2.3.1 Core Collapse Dynamics

Core collapse SNe can occur for stars whose masses are in the range of $9-40m_\odot$, and up to $60m_\odot$ if they have a high metallicity. SNe from $10-25m_\odot$ result in a neutron star, whilst those from $>25_\odot$ usually result in a black hole unless they have a high metallicity.

2.3.1.1 Capture Phase

The central region of a massive old star is composed of an iron core surrounded by concentric shells composed of lighter elements (primarily silicon, oxygen, helium and hydrogen, from inside to out) and supports its own weight through the pressure of degenerate relativistic electrons. Iron is the most tightly bound nucleus that can be achieved in stellar fusion and cannot be further burned. As the iron core contracts and the temperature of the core rises, iron can photodissociate into α particles and neutrons:

$$\gamma + {}^{56}\mathrm{Fe} \rightarrow 13\alpha + 4n, \tag{2.21}$$

which absorbs 124 MeV of energy and reduces the kinetic energy of the electrons. Electrons also capture on nuclei and free protons, producing neutrons and electron neutrinos which can leave the core and carry away much of the electrons' kinetic energy. The photodissociation and electron capture reduce the pressure provided by the electrons until the core collapses. As it does this, photodissociation and electron capture rates increase, acting to accelerate the collapse. Stars with a mass of $9-10m_\odot$ are not massive enough to burn oxygen but have neon and magnesium in their core which can undergo electron capture. As they do and begin a core collapse, these stars convert the oxygen, neon and magnesium to iron and also undergo an iron core collapse [30].

Over the course of this 'capture' or 'infall' phase, the density of the inner part of the core increases from 10^{10} g cm^{-3} to 10^{14} g cm^{-3}, the density of nuclear matter, over less than 10 ms. Due to the short time taken and high density, the capture phase only releases around 10^{44} J of energy. Above $\sim 3\times10^{11}$ g cm^{-3} the neutrinos can no longer escape from the inner part of the core as their mean free path becomes too short.

2.3.1.2 Neutronisation Burst

Upon reaching the density of nuclear matter, the inner part of the core is supported by nucleon degeneracy pressure and achieves hydrostatic equilibrium, forming a proto-neutron star. When the inner core stops collapsing and rebounds to become stable, it causes a supersonic shock wave which propagates through the outer iron core which is still contracting. The gas which is in a near free-fall state is decelerated significantly by the shock and begins to more slowly accrete onto the inner core. However, the rest of the outer core dissipates the energy of the shock via photodissociation of nuclei into free nucleons. The protons have a high electron capture rate, resulting in the production of electron neutrinos which pile up behind the shock wave until it reaches a region of density that is not opaque to them. This vast flux of $\sim 10^{46}$ J electron neutrinos occurs over a few ms and is known as the 'neutronisation burst' signal.

2.3.1.3 Accretion Phase

The shockwave loses about 10^{45} J for each $1m_\odot$ of material it photodissociates. If the shockwave is able to expel the entire envelope of a star, it is likely to do this on a timescale of $\sim$ 100 ms. However, even for stars weighing $10m_\odot$ the shockwave is likely to stall but will continue to photodissociate the matter which falls through it. A stalled shockwave can lead to the formation of a black hole without a full SN if the shock wave cannot recover from stalling.

During this 'accretion phase' neutrinos of all flavours are produced through heating of the shocked, hot material via energy released by accretion of matter to the proto-neutron star core. These are produced via electron-positron pair annihilation, electron-nucleon bremstrahlung, nucleon-nucleon bremstrahlung, plasmon decay and photoannihiliation which respectively take the forms [30]:

$$e^- + e^+ \to \gamma + \gamma \tag{2.22}$$

$$e^\pm + N \to e^\pm + N + \nu + \overline{\nu}, \tag{2.23}$$

$$N + N \to N + N + \nu + \overline{\nu}, \tag{2.24}$$

$$\gamma \to \nu\overline{\nu}, \tag{2.25}$$

$$\gamma + e^\pm \to e^\pm + \nu + \overline{\nu}. \tag{2.26}$$

Additional electron neutrinos are created via electron capture and antineutrinos via positron capture.

In the case of a stalled shock (considered to be the baseline model for a core collapse SN), a SN may still occur via thermally produced neutrinos in the core of the proto-neutron star [75] or through convection or oscillations [76]. This is still an area of active research, with few models having been able to successfully produce SNs from a stalled shock until recently [77][78]. The shockwave is expected to revive on the timescale of 0.5 s.

2.3.1.4 Neutrinosphere

The radius at which the density becomes low enough for the mean free path of the neutrino to allow it to stream out freely is called the neutrinosphere. The neutrinosphere is energy dependent, with lower energy neutrinos having a greater mean free path.

In fact there are actually 3 different types of neutrino which have their own energy-dependent neutrinosphere due to the impact of their interactions on their opacity. Charged current interactions with the protons and neutrons dominate for ν_e and $\overline{\nu_e}$ respectively. Combined with the region behind the shockwave being dominated by neutrons due to electron capture, the opacity is greater for ν_e, resulting in a larger neutrinosphere radius. The ν_μ and ν_τ and their antiparticles only interact via neutral currents and have lower neutrinosphere radii. This effect is generally modelled as producing a neutrino flux ϕ which, at a given time, is a function of the neutrino energy, E_ν, and a pinching parameter, α, which controls the suppression of the high energy tail [79]:

$$\phi(E_\nu) = N\left(\frac{E_\nu}{\langle E_\nu\rangle}\right)^\alpha \exp\left[-(\alpha+1)\frac{E_\nu}{\langle E_\nu\rangle}\right]. \tag{2.27}$$

The mean neutrino energy is $\langle E_\nu\rangle$ and N is a normalisation constant.

2.3.1.5 Cooling Phase

The accretion phase tends to produce neutrinos with a relatively low energy due to the increased neutrinosphere radius for higher energies. However, after the shockwave has revived, the matter density continues to decrease further and higher energy neutrinos stream out from the SN. Only $\sim 1\%$ of the energy of the SN goes into the ejection of matter in SNe, with the rest being carried away by neutrinos (in fact an order of magnitude more neutrinos than the initial number of leptons which were present [47]).

2.3.1.6 Summary Of Core Collapse

Initial collapse of the core produces an electron neutrino flux only as a result of electron capture. Once the inner core has achieved the density of nucleons, a shockwave occurs, allowing material which was already in an almost free-fall state to begin accreting on to the inner core and producing neutrinos of all flavours via equations 2.23-2.26. The shockwave itself results in a large neutronisation burst of high energy electron neutrinos until it is stalled. Whilst stalled, the neutrinosphere radii for the different types of neutrino stay relatively constant in size. When the shockwave resumes expansion, the neutrinosphere radii decrease in size, allowing a greater proportion of higher energy neutrinos to escape. The thermal emission of neutrinos is expected to last ~ 10s or more.

2.3.2 Searching For Physics Using Supernovae

Observation of a SN neutrino signal may allow a number of astrophysical and particle physics phenomena to be investigated. Simple examples include comparison of the neutrino time of flight to any gravitational wave signals and the potential to use the prompt neutrino signal to point telescopes in the right direction before the optical signal of the SN is observed. Additionally, if no SN is observed after a core collapse, then the formation of a black hole is expected to lead to a cutoff of the neutrino signal [80].

Other examples include oscillation of ν_e in the neutronisation burst and additional effects from neutrino-neutrino scattering within a halo of neutrinos [81]. The shockwave [82] and neutrino hydrodynamics [83][84] may also lead to oscillations which could have an observable impact on the neutrino flux. Understanding the neutrino energy spectrum will also help resolve models of SN dynamics.

As a SN is expected to be highly hermetic, measurements of the observed energy released by a SN in comparison to the expected energy converted through the loss of gravitational potential and the cooling rate of the SN can be used to search for potential new physics [85].

There are also a number of effects which the mass ordering is expected to have on the observed signal [86]. The most robust of these, both theoretically and observably, is the neutrino flux during the capture and neutronisation burst phases. The impact of the mass ordering on the expected

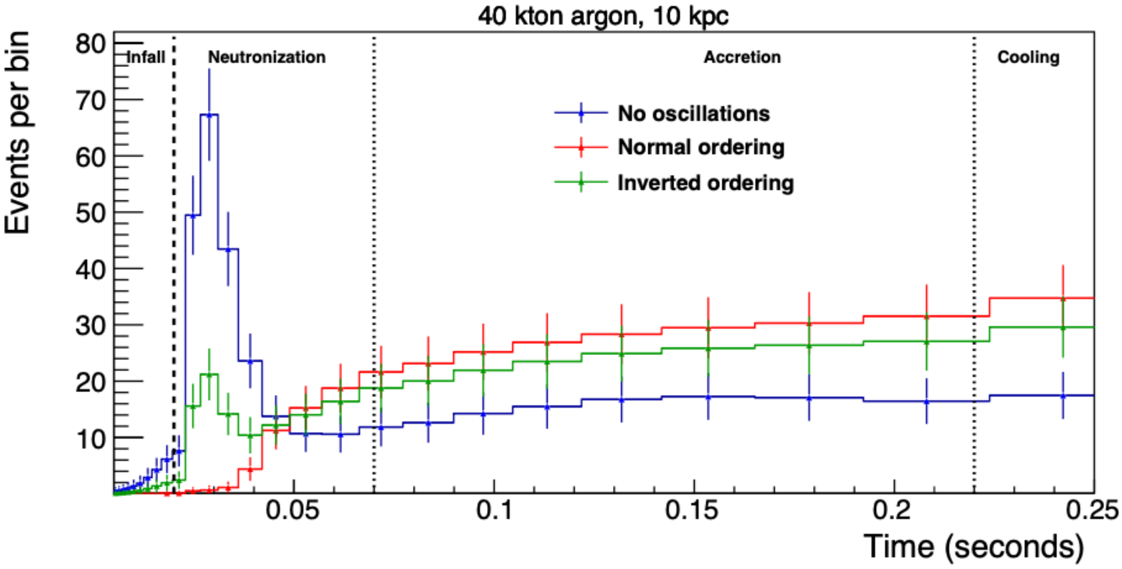


Figure 2.6: Expected event rates in a 40 kt LAr detector for a $9m_{\odot}$ SN at a distance of 10 kpc. The early unoscillated (blue) neutrino signal due to electron capture during core collapse and the subsequent neutronisation burst is suppressed due to neutrino oscillations in the normal mass ordering (red) and the inverted mass ordering (green). Image reproduced from [47][6].

signal in a LAr detector such as DUNE can be seen in figure 2.6, with neutrino oscillations very strongly suppressing the early flux in the case of the normal mass hierarchy and partially suppressing it in the case of the inverted hierarchy. This suppression is observed during the time period in the SN where the matter potential is dominant over the neutrino-neutrino potential and the flavour oscillations are more dependent on the matter effect [43].

2.3.3 Supernova Neutrinos In DUNE

The expected number of interactions observed in DUNE depends on the SN model used and the distance to the SN. Different SN models for different mass stars using different hydrodynamic equations of state and neutrino transport exist and are an active area of study [87].

The analyses of SNB trigger efficiency present in this thesis rely on a core collapse SN neutrino energy-time distribution [88]. This distribution is plotted in 2.7(a), with zero time taken to be the time of core bounce. The z axis represents the number of neutrino interactions for a SN at a distance of 10 kpc. This spectrum was produced using a program called SuperNova Observatories with GLoBES (SNOwGLoBES) [89] which convolved the expected neutrino flux, the neutrino cross sections and a simplified detector response. The neutrino flux was taken from the ‘Hudepohl-Cooling-Shen-s11.2co’ model [87] which corresponds to an $11.2m_{\odot}$ mass progenitor star. As mentioned in section 2.2.3, there are 10-20% uncertainties on CC and NC cross-sections

[6] Image reproduced under Creative Commons 4.0 license https://creativecommons.org/licenses/by/4.0/

in LAr, with the $\nu_e{}^{40}$Ar CC channel being dominant by an order of magnitude or more over the next most dominant cross section, ν^{40}Ar NC, for the energy range that the majority of SN neutrinos possess ($5-50$ MeV).

The energy spectrum provided was for the reconstructed energy, observed from a simplified detector response model. Ideally the energy spectrum would be quoted using the neutrino energy, but this information has not been provided. Despite this, the particular 'smearing matrix' (used by SNOwGLoBES to smear neutrino energy into a reconstructed neutrino energy with a given resolution) used in this case was from a model with a high resolution and should track the true neutrino energy reasonably well [88]. The energy spectrum for the expected number of neutrino energy interactions from a 10 kpc distant SN is plotted in figure 2.7(b). There is a hard cut-off at 5 MeV because the proportion of neutrino interactions producing enough ionisation charge to be detected in DUNE was expected to become negligible. The majority of neutrinos have an energy from $5-30$ MeV.

The spectrum provided for these studies was computed from a model which did not include any neutrino oscillations and as a result contains a large neutronisation burst feature, as can be seen in figure 2.7(c). As can be observed from figure 2.6, oscillations and the mass hierarchy have a large impact on the SN flux as a function of time. As a result of not having access to oscillated spectra, when studying triggering on this SNB signal it is best to look at the number of interactions over as long a time period as possible rather than to rely on any characteristic feature.

The expected number of neutrino interactions in 10 kt of LAr for SNe at different distances is shown in figure 2.7(d). The expected distance at which a SN is most likely to occur is around 10-15 kpc [47]. For a SN which occurred at a distance of 50 kpc (like SN1987A, although this had a much more massive progenitor), it is expected that there would have been 11 interactions in 10 kt of LAr. At a distance of 10 kpc, this model is expected to lead to ~ 267 interactions. SNe with larger masses would be expected to provide larger neutrino fluxes but $11.2m_\odot$ is a reasonable mass to establish a baseline efficiency for SN triggering.

Whilst the SN neutrino time-energy spectrum does not include any oscillations, has potentially large uncertainties in the ^{40}Ar cross sections and is for a single SN progenitor mass and model, it provides an opportunity measure the efficiency of a SNB trigger. Because the observed neutrino energy resulting from the model used should track the neutrino energy quite well, the observed neutrino energy will be used to approximate the true neutrino energy when drawing from this spectrum in studies presented in section 5.4 and 7.6.2. Due to the large number of uncertainties and the fact that only a single SN model was used, only the statistical errors computed for SNB trigger efficiencies are quoted in the results presented in this thesis.

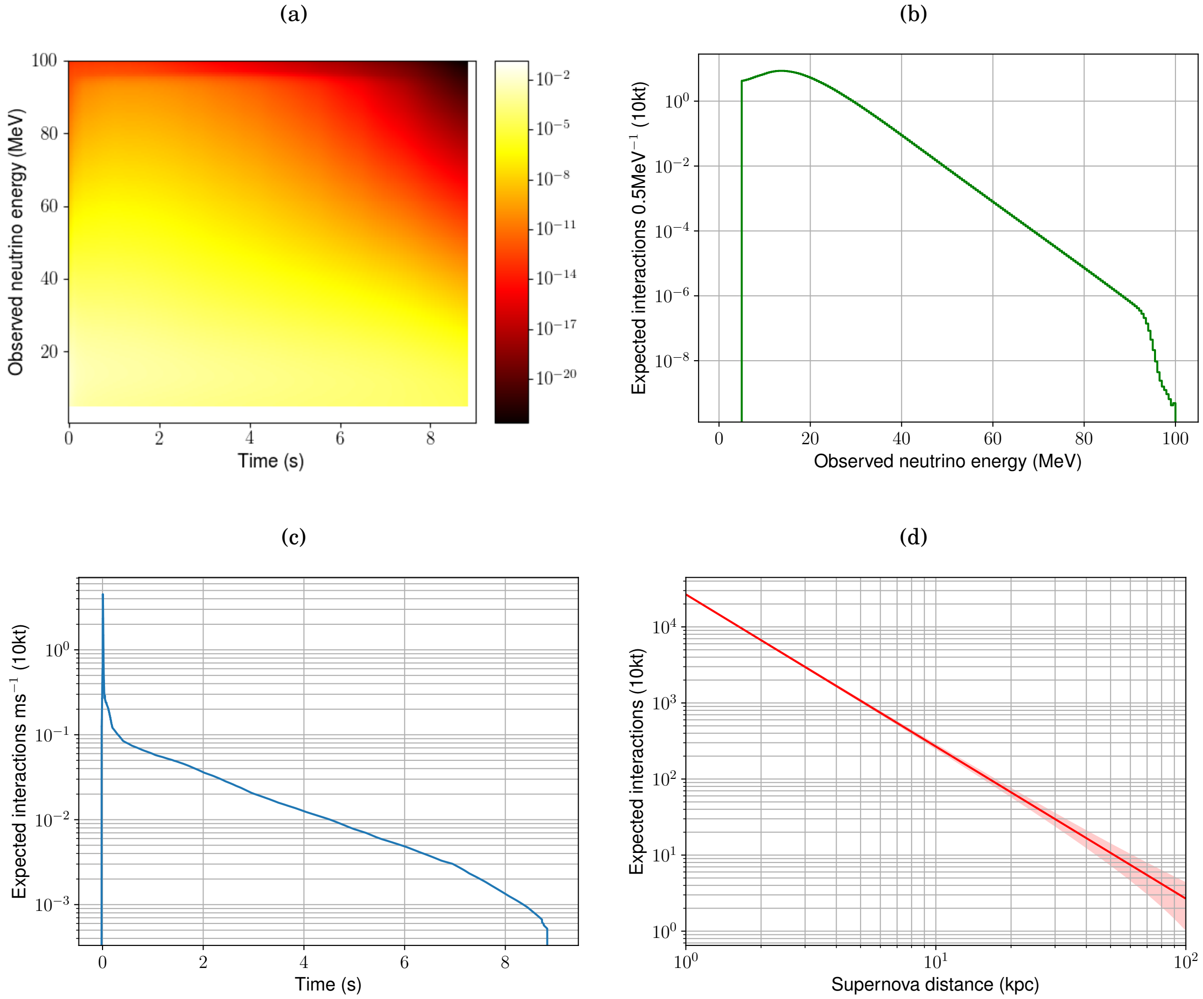


Figure 2.7: The unoscillated neutrino energy-time spectrum used as a benchmark for trigger studies, projections to the energy and time axes and the expected number of interactions as a function of SN distance, all for a detector volume of 10kt of LAr. The full energy-time spectra for a $11.2m_{\odot}$ SN progenitor at a distance of 10kpc is shown in (a), with projections to the energy and time axes given in (b) and (c) respectively with bin widths of 0.5 MeV^{-1} and ms^{-1}. The expected number of interactions in 10 kt of LAr, along with a band showing the 1σ statistical uncertainty from Poisson statistics is plotted in (d).

CHAPTER 3

DUNE Detector

The DUNE and the Long Baseline Neutrino Facility (LBNF) projects are hosted by FNAL and SURF. The LBNF project provides the conventional facilities for the DUNE detectors and the LBNF beamline. These will be located at a 'near' site at FNAL in Illinois and a 'far' site at SURF in South Dakota. As illustrated in figure 3.1, the near site will host the beamline and facilities for producing the high intensity neutrino beam as well as the DUNE near detector (ND), whilst the far site will host the DUNE FD.

The neutrino beam will be generated from a proton beam incident on a graphite target to produce secondary pions and kaons which will be focussed via a magnetic horn. The hadrons of a given charge can be selected to produce a muon neutrino via $\pi^+ \rightarrow \mu^+ + \nu_\mu$ or an antimuon neutrino via $\pi^- \rightarrow \mu^- + \overline{\nu}_\mu$ and similarly for kaons. The LBNF beamline will utilise the Fermilab Main Injector proton beam after the Proton Improvement Plan II (PIP-II) upgrade [90] to the linear accelerator (LINAC). This will provide a proton beam with a power of 1.0 MW to 1.2 MW and a wide energy band from 60 GeV to 120 GeV [91].

3.1 DUNE Near Detector

The resulting neutrino beam will pass through the ND 574 m downstream, which will serve as a control for the experiment by measuring the initial ν_μ and contaminant ν_e spectra and constraining the systematic errors. The ND will actually be composed of 3 detectors, which can be aligned along the beam axis. The upstream detector will be a 5 m (along the beam axis) by 7 m by 3 m LArTPC with a fiducial mass of 67 t called the ArgonCube Liquid Argon detector [92]. Immediately downstream from this will be a high pressure gaseous argon TPC surrounded by an electromagnetic calorimeter, referred to as the Multi-Purpose Detector (MPD). This calorimeter

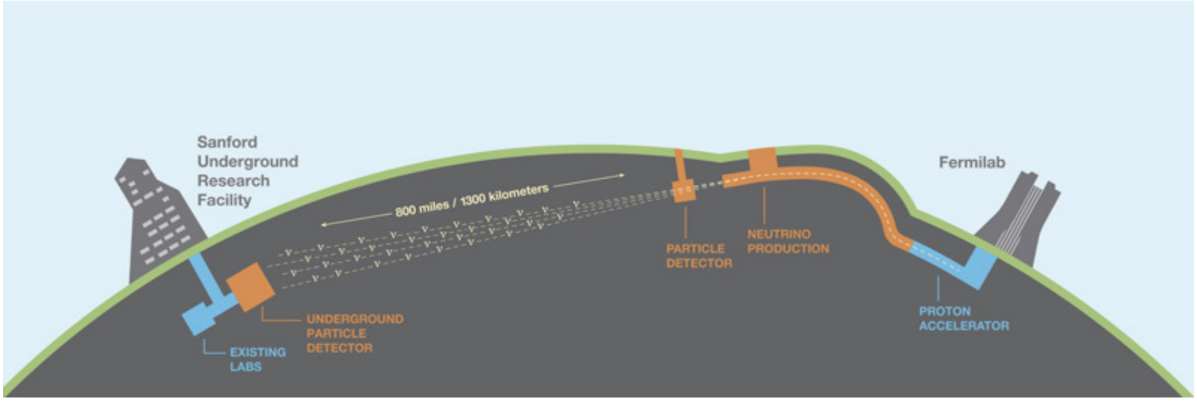


Figure 3.1: Cartoon of the path of the neutrino beam generated by the proton LINAC at FNAL through the ND, also located at FNAL to the FD hosted 1300 km away and 1.5km underground at the SURF from [50] [1].

will have an active region of 5.2 m in diameter and 5 m in length. It will be required to measure the momentum of muons with momenta greater than 0.7 GeV/c as these will not be contained by the LArTPC. From the measured muon momenta, the ν_μ energy can be inferred. The ArgonCube detector and the MPD will both be able to move off-axis by up to 33 m to measure the neutrino flux spectrum away from the the beam axis. These off-axis measurements will allow DUNE to avoid biases in measured oscillation parameters [92]. The downstream part of the ND, the System for on-Axis Neutrino Detection (SAND), will be permanently located on the beam axis to ensure that the beam remains stable during off-axis data collection and will reuse the $\mathrm{K_L}^0$ LOng Experiment (KLOE) detector and electronic calorimeter [93].

3.2 DUNE Far Detector

The neutrino beam will travel 1300 km through the Earth to the DUNE FD, which will be located 1.5 km underground. Its subterranean depth will reduce the background rate of cosmic ray interactions in the detector at the surface from 165 kHz, corresponding to a ratio of less than 1 neutrino interaction per million cosmic rays, to just over 1 neutrino interaction per cosmic ray [50]. The DUNE FD will be composed of 4 large cryostat modules each filled with 17.5 kt of LAr and possessing a fiducial mass of 10 kt contained within LArTPCs. Each cryostat will be encapsulated by an insulated double membrane based on technology commonly used in industry for storage of liquefied natural gas. There are multiple detector designs for installation in these cryostat modules, with at least 1 module using single phase (SP) LArTPC technology previously tried and tested on a smaller scale at ICARUS [94][95] , MicroBooNE [96] and ProtoDUNE single phase (ProtoDUNE-SP) [97] among others.

[1] Image reproduced under Creative Commons 4.0 license https://creativecommons.org/licenses/by/4.0/

At least one module will use a different technology called ‘vertical drift’ [98]. The technology for use in the remaining modules is yet to be decided, with ArgonCube LArTPC technology used in the ND being a candidate for use in one of them [99]. The studies presented in this thesis were performed on the SP LArTPC technology as presented in the following sections. First, the SP FD is introduced, followed by an explanation of the anode plane assembly (APA) instruments used to detect ionisation and scintillation in the detector volume. The detector readout is then explained, followed by the components of the DAQ system relevant to triggering the detector and storing the collected data.

3.2.1 DUNE Single Phase Far Detector

A DUNE SP FD module will have its fiducial volume of LAr held at 88 K and contained in a field cage. Within this field cage the detector will be split into LArTPCs between the 3 anode planes and the 2 cathode planes, as shown in figure 3.2. The planes extend for 58.2 m along the beam (z) axis of the detector and 12 m along the vertical (y) axis. The cathode planes will be held at a voltage of -180 kV and in combination with the field cage, provide a uniform electric field in the fiducial volume of the detector which act to drift ionisation electrons towards the anode plane. The anode planes will be composed of APAs which are each 2.3 m wide and 6 m tall. Each anode plane will have APAs stacked 25 wide and 2 high, resulting in 50 APAs per plane and 150 in a FD module.

Complementarily, the cathode planes will each be composed of 300 cathode plane assemblies (CPAs), each 1.2 m wide and 4 m tall, which will be bolted together to match the 58.2 m by 12 m size of the anode planes. The CPAs are composed of glass-reinforced epoxy laminate frames 6 cm thick and holding 3 mm thick panels coated in a thin layer of Kapton (a highly resistive material).

The drift length between the anode and cathode planes will be 3.59 m, resulting in an electric field of $500\,\mathrm{V\,cm^{-1}}$. The drift velocity will be $1.6\,\mathrm{mm\,\mu s^{-1}}$ [100]. The signal attenuation over the drift distance should be limited to less than 20% [101] which requires an electron lifetime of 11 ms.

In addition to the ionisation produced in the detector, the APAs will also use silicon photomultipliers (SiPMs) mounted on the APAs to detect the scintillation light produced. This will allow the time of interactions to be determined more accurately. These will follow the X-ARAPUCA design [102], where the SiPMs will be mounted inside casings composed of a dichroic filter and a wavelength shifter to select the characteristic LAr scintillation light and shift it into the visible spectrum for detection. In the work presented in this thesis, only ionisation information detected via the APAs has been used.

3.2.2 Anode Plane Assemblies

APAs of the same design have already been successfully built for and implemented in the ProtoDUNE experiment [97]. Each APA consists of an aluminium frame with 4 layers of 152 µm

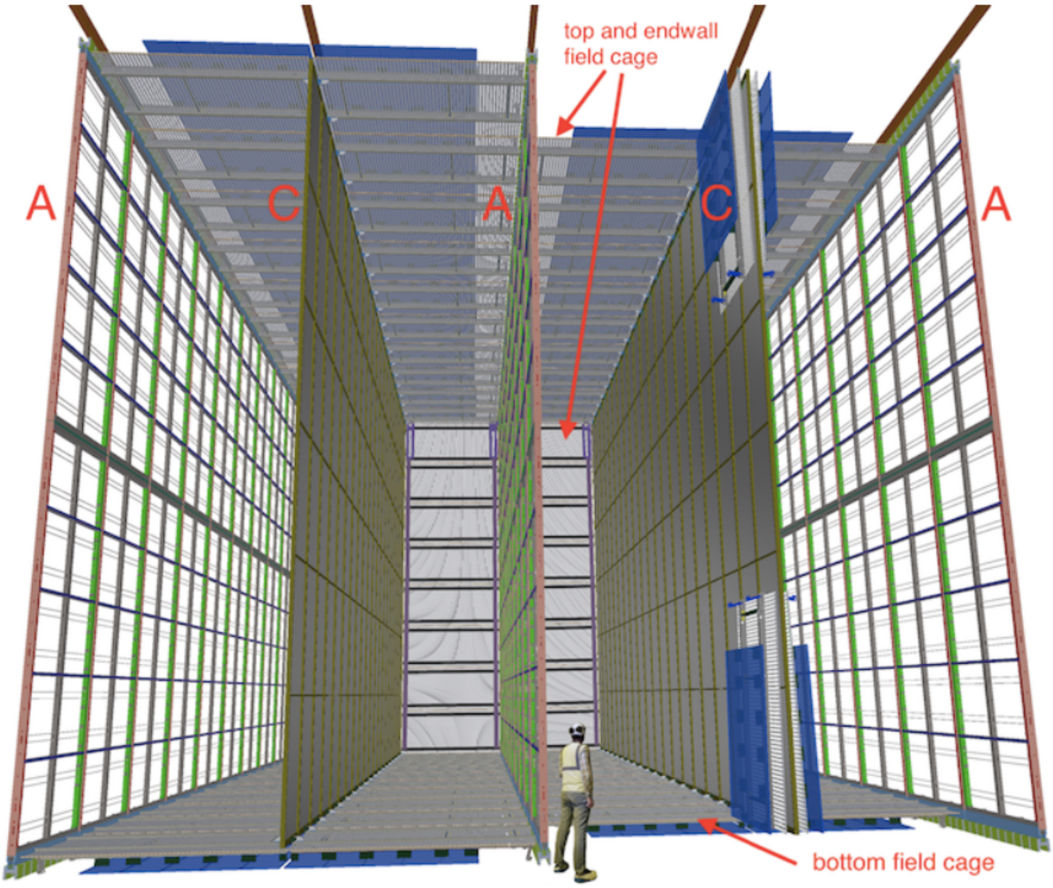


Figure 3.2: Cross section of a DUNE SP FD module through the x-y plane showing the arrangement of anode (A) and cathode (C) planes inside the surrounding field cage. Dimensions are 58.2 m along the z (beam) axis, 12 m along the y axis and 3.5 m for each drift length between an anode plane and a cathode plane. Image reproduced from [50][2].

diameter beryllium-copper wires held under a tension of 6 N. Three of these layers are connected to the cold electronics (CE) for readout whilst the outermost layer is not. It instead acts to shield and improve the signal shaping on the inner wires. This outermost layer of 960 (480 per APA face) wires is known as the grid (G) layer and the wires are aligned parallel to the y axis of the detector when installed.

The next two layers of 800 wires each are induction wires, denoted as the U and V layers. These receive an induced bipolar signal as charge drifts past them towards the innermost (collection) wires. This innermost layer of wires, also denoted as the X layer, consists of 960 collection wires (480 wires per APA face), which collect the charge and receive a unipolar signal. In between the X wires on each APA plane is a grounding mesh which acts to improve the uniformity of the electric field. As illustrated in figure 3.3, the X wires are arranged parallel to

[2] Image reproduced under Creative Commons 4.0 license `https://creativecommons.org/licenses/by/4.0/`

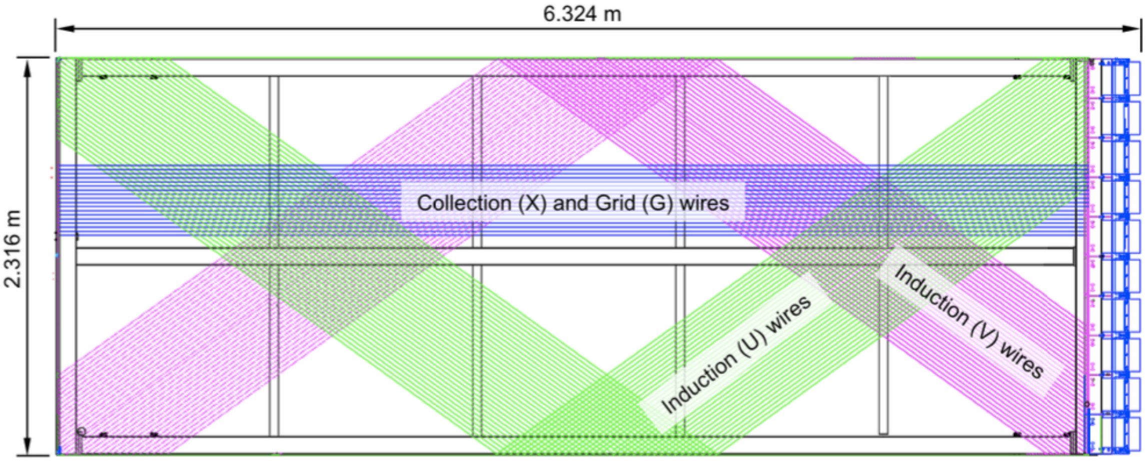


Figure 3.3: Diagram of the APA frame in black, with the cold electronics on the right side at one end. These read out the signals induced on the U (pink) and V (green) induction wires. The induction wires are wrapped at 35.7° relative to the collection/grid wires (blue), overlapping with each once. When installed, the collection wires are parallel to the y axis of the detector. Image reproduced from [100] [3].

the y axis of the detector whilst the U and V wires are wrapped around them at a relative angle of ±35.7°. This ensures that each X wires crosses each U and V wire once, which allows ionisation depositions to be mapped on the y-z plane (the face of the APA). The voltages each wire layer is held at ensure that the first two layers are transparent to the drifting electrons. These properties of the wire layers and the voltages they are held at during operation are summarised in table 3.1.

Table 3.1: Summary of APA wires from outermost (G) to the innermost (X), the bias voltages used in DUNE and the connection to the CE.

Layer	Number of wires	Bias voltages (V)	Connection to CE	Wire pitch (mm)
Grid (G)	960	-665	None	4.79
Induction (U)	800	-370	CR board	4.67
Induction (V)	800	-0	Direct	4.67
Collection (X)	960	820	CR board	4.79
Grounding mesh	n/a	0	n/a	n/a

The U and X wires are connected to the CE through capacitive resistance boards which act as a high pass filter but the V wires are directly connected to the CE.

[3] Image reproduced under Creative Commons 4.0 license `https://creativecommons.org/licenses/by/4.0/`

3.2.3 Cold Electronics

As seen in figure 3.3, the TPC readout CE is directly mounted to the head (right) end of the APA, minimising the capacitance of the connecting wires and therefore the electronics noise. The immersion of the CE in the 88 K LAr also helps to reduce the inherent electronic noise.

The 2560 wires on an APA are read out by 20 front-end motherboards (FEMBs), corresponding to 128 wires each. Each FEMB hosts 8 front-end (FE) application specific integrated circuits (ASICs) and 8 16-channel analogue to digital converted (ADC) ASICs. The FE ASICs provide amplification and signal shaping, whilst the ADC ASICs digitise these signals over a 12 bit range at a sampling rate of 2 MHz. Saturation occurs at the collection of 500,000 electrons, but the ADC conversion response otherwise linearly scales with the charge collected. For context, a MIP deposits 20,000 to 30,000 electrons depending on recombination effects [100]. The TPC electronics are required to have an 'equivalent noise charge' of 1000 or fewer electrons.

The layers of anode wires on the APA are separated by 4.75 mm and the drift velocity between the wire planes is between 1.4 mm μs^{-1} and 2.1 mm μs^{-1}. The time taken for a signal to transition from its baseline to its maximum value is known as the peaking time. The time taken for the drifting charge to travel between the wire layers typically corresponds to the duration of the signals observed. As a result, the peaking time used for signal shaping in the FE electronics is programmable in the range 1-3µs. The signal is then sampled at a rate of 2 MHz according to the Nyquist-Shannon sampling theorem [103], which allows the discretisation to capture all the information from the continuous time signal.

The FEMBs are connected to warm interface boards (WIBs) via a signal cable, with each WIB interfacing to 4 FEMBs (therefore there are 5 WIBs per APA). The WIBs pass TPC data downstream to the DAQ.

3.3 Data Acquisition

There will be a mezzanine floor above each FD module, referred to as the Cryo Mezzanine (CM). The DAQ system will be located physically at SURF, split between the CM and the main communications room (MCR) located on the surface. The raw data rate will be 1.2 TB s^{-1} per FD module, with the vast majority resulting from the ionisation information collected by the APAs. Only 30 PB per year of data storage will be allocated for all the data saved to disk from a single FD module, necessitating a reduction in the data volume by over 3 orders of magnitude. As a result, DUNE must make use of a trigger to limit data collection to important physics data. Triggering will be implemented as part of the DAQ system and the required processing must be able to keep up with the data rate from the FE without any dead-time.

The DAQ will be split into 5 subsystems: upstream data acquisition (UD), data selection (DS), the DAQ back-end (BE), the control, configuration and monitoring (CCM) and the timing and synchronisation system. As shown in figure 3.4, the data flow is from the UD through the DS and subsequently the DAQ BE. Data triggered on by the DAQ will be sent from the DAQ BE via a wide area network (WAN) to FNAL for permanent storage. The timing and synchronisation subsystem will distribute synchronous time signals to the DAQ and detector components, whilst the CCM subsystem orchestrates the data collection.

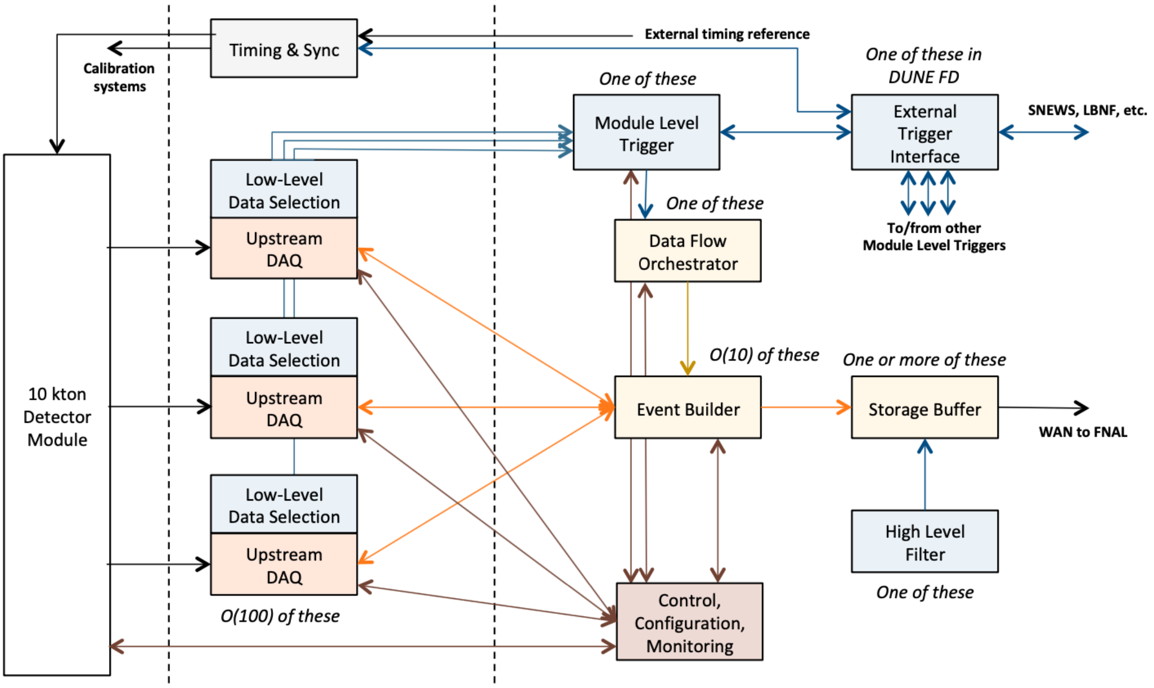


Figure 3.4: Diagram of the connections between the DAQ subsystems for a FD module. The split of DAQ subsystems between the cryostat, Cryo-Mezzanine and Main Control Room, from left to right, is shown by dashed lines. The UD subsystem is coloured orange, the DS subsystem is coloured blue, the BE subsystem is coloured in yellow, the CCM subsystem is coloured brown and the timing and synchronisation is coloured grey. Image reproduced from [100][4].

The UD subsystem will be located in the CM and be responsible for reception, buffering and preprocessing of the raw data. The DS subsystem will receive the preprocessed data from the UD and perform additional processing which may result in a trigger command being issued by the module level trigger (MLT). Data selection will be split between the CM and the MCR, with the data bandwidth for transmitting triggering information from the low level DS to the surface limited to 10 Gbit/s. If a trigger command is issued to the DAQ BE, then it will request the corresponding data from the UD buffers which will be read out over a 100 Gbit s^{-1} ethernet network and formed into an event record.

Multiple triggering algorithms may be applied in the MLT and potentially the high level filter (HLF), with the goal of each being to save event records relevant to signals from a given type of event. The DUNE DAQ design has been driven by the following 2 scenarios:

[4] Image reproduced under Creative Commons 4.0 license https://creativecommons.org/licenses/by/4.0/

1. a localised high energy (>100 MeV) interaction
2. an extended low energy trigger for SNBs which will result in multiple low energy (≈10 MeV visible energy) interactions over 10 s over the entire detector volume.

Each event record will consist of the full detector readout for a time period dependent on the event type. For a localised high energy interaction, 1 event is 5.4 ms of ionisation data, which will correspond to 6.5 GB. However, in the case of a SNB event, 100 s of data will be collected by the detector module, requiring 120 TB of data for the event record. The event records will be stored in a buffer in the DAQ BE before being sent to permanent storage. Event records in this buffer may be accessed by a HLF (a part of the DS subsystem) which will apply further processing and can discard records which do not meet the criteria for permanent storage.

Apart from the 100 Gbit s^{-1} connection between the DAQ BE and the UD data buffer, the UD, DS and DAQ BE subsystems will be connected via a 10 Gbit s^{-1} switched network which will allow data, processed trigger summary information, trigger decisions and data requests to be communicated. Further details about the UD and DS and DAQ BE subsystems relevant to triggering on SNB events will be given in the following sections.

3.3.1 Upstream Data Acquistion

The scope of the UD in the trigger chain will be from the reception of data from the WIBs to provision of trigger primitives (TPs) (detailed in section 3.4.1) to the DS subsystem. The UD will be comprised of 75 DAQ readout units (RUs), each with identical hardware and responsible for processing the ionisation data from 2 APAs. Each DAQ RU will be composed of a DAQ FE computer and a server which hosts custom hardware, firmware and software which implement the reception, buffering and preprocessing of the raw data, as well as receiving and fulfilling data extraction requests.

The server will host 256 GB of double data rate fourth generation (DDR4) random-access memory (RAM) and 2 TB of solid-state drive (SSD) storage. This will satisfy the requirements to be able to buffer 10 s of TPC information and to stream 100 s of TPC information to disk upon a SNB trigger command. Localised activity in the detector should require only 1 s of buffering but the early part of a SNB event signal may look similar to the signal from inherent radiological background in the detector and require a longer time, up to 10 s of buffering before a trigger will be issued.

The DAQ RU server will also host 2 front-end link exchange (FELIX) input/output cards [104]. The FELIX cards will each be responsible for processing all the TPC data from a single APA. Each FELIX card supports a bandwidth of 16 GB s^{-1} with its host computer over Peripheral Component Interconnect Express (PCIe) 3.0.

Raw TPC data from each APA will be conveyed from the WIBs via ten 10 Gbit s^{-1} optical links to the FELIX FPGA board via multi-gigabit transceivers (MGTs). This data will be temporarily stored in a circular RAM buffer and be interpreted and processed in the FPGA to extract summary information for use in triggering. Each instance of this low level summary information, is called a trigger primitive (TP). These will be passed downstream to the DS subsystem. The trigger primitive generation (TPG) functional block is explained in greater depth in section 3.5.2.

3.3.2 Data Selection

The DS subsystem will be hierarchical, with a low-level data selection stage implemented in the MLT, and a high-level stage optionally implemented in the HLF. In the low-level data DS, the TPs produced by the UD will be further processed on the server using CPU resources. TPs will only summarise signals on a single wire. The low-level DS aggregates TPs into one or more trigger cluster (TC), which represents the charge collected in a local region of the x-z plane (corresponding to time and wire extent) in the detector. As illustrated in figure 3.4, these TCs will be sent to the MLT, which uses them to form a module-level trigger decision.

The MLT will also be connected to the external trigger interface (ETI), through which it can be sent external triggers. These may be triggers from other MLTs, such as in the case of a SNB trigger being issued in another detector module. The MLT will also be able to receive external triggers from global coincidence trigger alerts, such as the supernova early warning system (SNEWS) [105], which looks for coincident signs in multiple detectors of the rising neutrino flux which occurs before core-collapse of a SN. It will also be able to receive trigger commands from another detector system in the module, such as the calibration system.

The last stage of the DS is the HLF. The low-level data selection should significantly reduce the amount of data which is read out by the DAQ BE. The HLF acts on event records, allowing it to implement DS algorithms which are more complex. The scope of the processing which may be applied in the HLF could be very simple or quite complex. For example, the low-level DS will not be able to apply algorithms which suppress correlated noise across wires in the detector and may result in a trigger command being issued. In such a case, the HLF could apply simple processing to discard any such event records (or retain them if physics signals remain). The hardware which will be used in the HLF is not yet decided but has the potential to be largely graphics processing unit (GPU) based.

3.3.3 Data Acquisition Back-end

The DAQ BE will have 3 parts: a data flow orchestrator (DFO), a set of event builder (EB) processes and a storage buffer. The DFO will receive the trigger commands from the MLT and distribute each to an EB process. The EB will query the UD buffers and then receive and process the data into event records which are saved as files in the output buffer. The output buffer will allow the DAQ to be resilient to fluctuations in trigger rates and potential delays in processing event records in the HLF. The output buffer will consist of several PB of storage, allowing several days of the triggered data to be stored. This will ensure that short breaks in the connection between the DAQ and offline storage do not affect the up-time of the detector.

3.3.4 DAQ Requirements

The DAQ is specified to require >95% efficiency for triggering on SNBs within 20 kpc. These are expected to produce at least 60 neutrino interactions in the active volume of a detector module during the first 10 s of the burst signal. Therefore the maximum latency of the SNB trigger is 10 s, although faster triggering would be beneficial as this would preserve interactions from any neutrinos which may precede the core collapse. The latency would also be affected by the energy-time spectrum of the SNB, which depends on the mass hierarchy, as shown in section 2.3.

The false positive rate (FPR) of the SNB trigger will be the main limiting factor on the achievable efficiency. As mentioned in section 3.3, only 30 PB per year per FD module can be sent to permanent storage. Of this, 2 PB will be allocated to store data from SNB triggers, leading to a maximum FPR of around one per month.

Fifteen PB of the data volume will be allocated to store all the TPs found by the UD as well as the data for detector performance studies. The intrinsic rate of radiological ^{39}Ar decays (present in the LAr) has been suggested for use as a calibration source, as $\sim 250,000$ decays can provide measurement of electron lifetime to 1% accuracy [106]. The data volume collected would depend on the desired pixelisation for the measurement of the electron lifetime. One measurement per APA per day to this accuracy would require 1.5 PB per year but finer pixelisation is desirable.

The quoted offline data storage volumes are before any data compression. DUNE will use lossless compression algorithms. From previous studies in the Micro Booster Neutrino Experiment (MicroBooNE), TPC information is expected to compress by a factor between 2-4 [107] but the TPs will not be compressible. Balancing the disk space required by detector performance studies, a conservative limit on the rate of TPs would be 2 PB per year per FD module.

Once the data from a SNB trigger has been stored offline it can be further investigated. If it became clear that the data did not result from a SNB, then the event record could be deleted, freeing up additional offline storage space and allowing the false SNB trigger rate to be relaxed. However, no studies of algorithms to reject SNB candidate event records have been done, so 2 PB per year and the resulting once per month FPR will be used as a conservative limit to determine the performance of SNB trigger algorithms.

3.4 Baseline SNB Trigger

The baseline SNB trigger design for DUNE is a simple trigger that counts the number of TCs which occur in a given time window and issues a SNB trigger command if a threshold (set using the 1 false positive SNB per month requirement) is met. As a result this trigger depends directly on the algorithms used to create TPs and TCs and their particular parameters. Its efficiency is limited by the radiological backgrounds present in the detector. In this section TPs and TCs will be defined, along with the algorithms which have been used to generate them in the work presented in this thesis.

The baseline design for the DUNE DAQ is to process TPC data using FPGA logic. A previous study has been completed by DUNE collaborators to determine the performance of a SNB trigger using a TPG algorithm designed to be implemented in CPU [85]. The algorithm which will be implemented in the FPGA-based SNB trigger will follow the same processing steps as that studied for CPU but using some different parameters. The differences between the two approaches exist only in the TPG stage and are elaborated on in chapter 5, where the TPs, TCs and SNB trigger performance resulting from FPGA-based TPG will be characterised.

In the CPU TPG study, the parameters used for clustering the TPs into TCs were chosen to be those which achieved the optimal SNB trigger efficiency for a FPR averaging one fake trigger per month. A 'brute force' method was used to study 40 combinations of clustering parameters and the resulting optimal clustering parameters are set out in section 3.4.2. These parameters were re-used to determine the performance of the baseline SNB trigger which uses FPGA-based TPs as will be reported in chapter 5.

3.4.1 Trigger Primitive Generation

TPs summarise the charge deposited on individual wires by ionisation signals to reduce the data volume passed downstream. The TPG is composed of 3 stages: pedestal subtraction, filtering and hit-finding. In FPGA-based TPG these algorithms will be implemented in a series of functional blocks which will be explained in section 3.5. The pedestal subtraction will act as a high pass filter and the filtering as a low pass filter. This will effectively result in a bandpass filter which futher shapes and smooths the signal on each wire. If the resulting signal (each digitised sample will be referred to as an ADC sample) exceeds a set hit-finder threshold, T_{HF}, then a TP will be formed which describes the signal for the duration it exceeds the threshold. This threshold is a parameter which can be adjusted to satisfy the limits on the number of TPs which can be saved to disk and should be kept as low as possible. TPs may also be referred to as 'hits' and the properties of each TP will be:

- Start tick, H_S, the tick of the first ADC sample which meets or exceeds T_{HF}
- End tick, H_E, the tick of the last ADC sample which meets or exceeds T_{HF}
- Ticks over threshold, $H_{TOT} = H_E - H_E$
- Peak tick, H_P, the tick of the ADC sample within the TP with the highest ADC value
- Peak ADC, H_{PADC}, the ADC value corresponding to H_P

- Summed ADC, H_{SADC}, the sum of all ADC values during the period the signal equals or exceeds T_{HF}.

Each TP will also contain a timestamp which will allow the tick information to be related to the detector clock and information allowing the individual wire to be identified in the detector module.

3.4.1.1 Pedestal Subtraction

When there is no ionisation input, the signal measured is called the pedestal. To remove fluctuations due to slow changes in the pedestal of the signal, a frugal streaming algorithm [108] was used. This simple algorithm made use of two quantities which were updated as each ADC sample on a wire was processed: a pedestal, P and an accumulator P_a. The pedestal was subtracted from the ADC value to ensure that in the absence of a signal the high frequency noise oscillated around a baseline of zero. The accumulator was used to update P before it was subtracted from the ADC value. If the next ADC sample was above P, then P_a would be increased by 1 (vice versa for below the pedestal). If P_a reached a threshold of $\pm X$, then P would be increased or decreased by 1. This algorithm allowed slow changes in the pedestal to be tracked and eliminated from the signal. A value of $X = 10$ was used for all studies presented in this work.

3.4.1.2 Filtering

After application of the pedestal subtraction (PS), to smooth fluctuations in the signal due to high frequency random noise on a wire, a finite impulse response (FIR) filter was used. This is essentially a weighted sum of the last N samples in the signal. The weights are referred to as ‘filter taps’, T and are configurable to shape the signal in a desired way. As unipolar signals were expected on collection wires, the values of these filter taps were chosen to be symmetrically distributed around the central tap, with the outside taps having the lowest values and the central ones the highest. For a digital signal with n samples, and a filter with N taps, the output for the n'th sample from the application of the FIR filter f depends only on samples $x_{n-N},...,x_n$:

$$f_n = \frac{\sum_{i=0}^{N} T_i x_{n-i}}{\sum_{i}^{N} T_i}. \tag{3.1}$$

where i is the index of the filter tap and the weighted sum has been divided by the sum of the filter taps to preserve the dynamic range. Using the symmetric and centrally peaked filter tap values causes the FIR filter to act to narrow and delay the peak of the signal by $N/2$ samples.

3.4.2 Trigger Cluster Generation

Whilst low energy neutrino interaction signals are quite small on an APA, to be detected with a low enough FPR they need to be differentiated from the ionisation which results from radiological backgrounds. This will be discussed further in chapter 5. To identify local regions of charge deposition which span one or more wires, the TPs are clustered together into TCs. TPs within a certain wire tolerance, t_W, are first grouped into protoclusters. The TPs in protoclusters are then grouped into TCs if they fall within a certain tick tolerance, t_T. The tolerance is defined as the number of wires or ticks without a TP which can be tolerated between adjacent TPs in the cluster. Clusters can be discarded using thresholds based on their properties. The full clustering algorithm is set out in appendix A.1.

The TCs possess the following properties:

- The number of TPs they are formed from, C_n
- The number of wires the TPs are on, C_{nW}
- The tick width, C_{rT}, which is the range of samples the TC covers
- The wire width, C_{rW}, which is the range of wires the TC covers
- The cluster ticks over threshold (TOT), C_{TOT}, which is the total number of ticks the TPs were above the hit-finder threshold for
- The summed ADC, C_{SADC}, which is the sum of the H_{SADC} values from the TPs.

From the study mentioned at the start of section 3.4, the combination of clustering parameters and applied thresholds which was found to result in the greatest SNB trigger efficiency is shown in table 3.2.

Table 3.2: Clustering parameters found to provide the greatest efficiency for a SNB TC counting trigger using CPU-based TPs [85].

Clustering Parameter	Clustering Parameter Value
Wire tolerance (t_W)	1
Tick tolerance (t_T)	20
Minimum C_n	6
Minimum C_{nW}	2

3.4.3 Supernova Burst Counting Trigger

The SNB counting trigger is a simple algorithm which counts the TCs which occur within a time window in the detector module and issues a SNB trigger command if the count reaches a threshold, T_{SNB}. To determine the performance, the rate of TCs from background sources in a given time window is determined and a threshold for the number of TCs which keep the false SNB rate to once per month or below is computed. This threshold is used as T_{SNB} to determine the efficiency for the SNB trigger, ϵ_{SNB}, as a function of the distance of a SNB.

3.5 Data Processing Firmware Architecture

The UD implements data processing to generate TPs in the FPGA mounted to the FELIX board using firmware implemented in a series of functional blocks. DUNE will use the same firmware architecture as has been developed by the DUNE DAQ collaboration and tested in ProtoDUNE-SP with small changes as explained in section 3.6. Each block in the firmware architecture fulfils a task in the processing pipeline which is run at 250 MHz to allow it to keep up with the online data rate.

The raw data is received from the MGTs in data packets, called WIB packets, each holding ADC data from two FEMBs (in total 256 channels) for a single detector tick. Ten instances of the data processing firmware, one per MGT are instantiated. As illustrated in figure 3.5, the ADC TPC data is received by the data router block which performs some data transformation and stores the data in a circular buffer in the dual ported random access memory (dpRAM) buffer. The unpacker then accesses this data when ready and demultiplexes it to be processed on 4 identical TPG blocks which each process the signal from a single wire at a given time. The TPs determined from the signal are packaged into 'hit packets' and pushed to the arbitrator block, which multiplexes them and forwards the data to the central router (CR) interface block which transforms the TPs into a format interpretable by the DS. The CR is able to prevent backpressure, filter out empty hit packets, and remove corrupted hit packets in addition to translating hit formats.

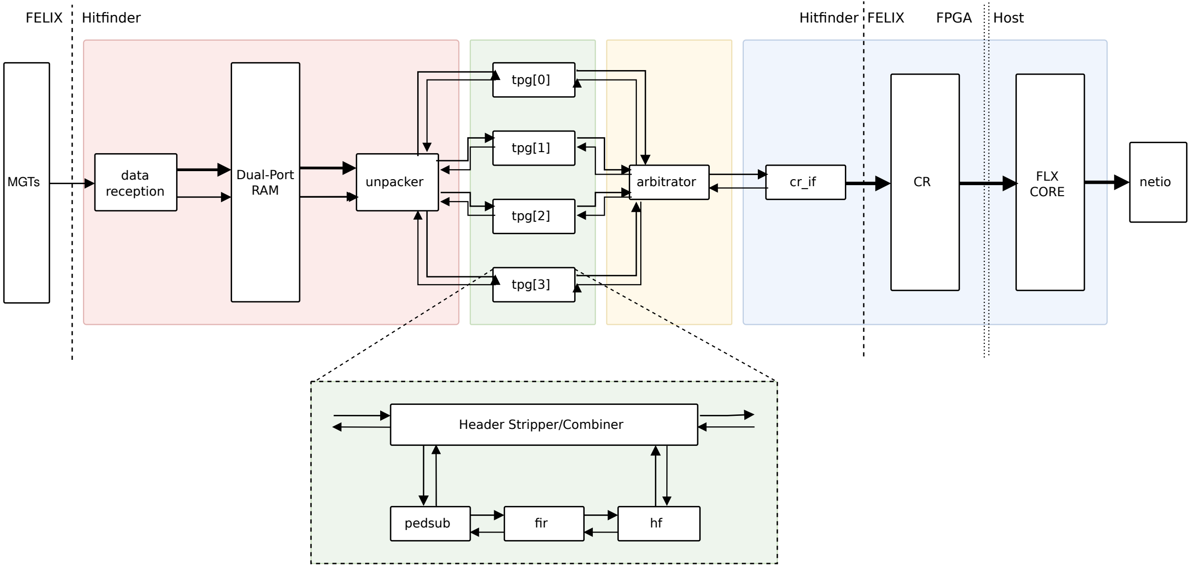


Figure 3.5: Firmware functional blocks in the FPGA and the connections between them. The data router block is highlighted in red, the TPG generation block is highlighted in green, the data arbitrator is highlighted in yellow and the CR interface is highlighted in blue. Image reproduced with permission from [109].

3.5.1 Data Router Block

The data router receives the ADC TPC data from an APA via an MGT and transforms multiple WIB packets of the data into ADC processing packets which carry the ADCs for a single wire for a given number of ticks. This number is configurable and limited by the block random access memory (BRAM) constraints of the FPGA. A value of 64 ticks was used in the work which will be presented in chapter 4. Any TPs which extend beyond the boundary of an ADC processing packet will be flagged and stitched back together in the downstream DS before clustering.

Once the ADC processing packets are stored in the dpRAM, these are accessed by the unpacker subblock which reads and demultiplexes the data for 4 channels at a time, providing each ADC processing packet to an independent TPG block instance. The ADC processing packets are transmitted downstream using the advanced extensible interface 4 stream (AXI4-Stream) protocol [110] which is used until the CR interface converts the format of the TPs and passes them to the FELIX CR.

3.5.2 TPG Block

The TPG block has several subblocks. The first strips header information and passes it downstream to become the header for the hit packet which results from the ADC processing packet. The ADC processing packet has PS and FIR filtering applied to it, both of which implement saving and restoring of state between consecutive ADC processing packets from a given channel. The ADC processing packet is then processed by the hit-finder (HF) subblock and the processing packets now hold the TPs which were found from the 64 ADC samples. Any resulting TPs are recombined with their header information by the header combiner. The arbitrator block multiplexes the outputs of the 4 different TPG instances back into a single AXI4-Stream output before the CR-interface translates these AXI4-Stream hit packets into the CR hit packet format, which is propagated forward.

3.6 ProtoDUNE Single Phase Detector

The DUNE collaboration has constructed and operated 2 large prototype detectors at European Council for Nuclear Research (CERN), one of which demonstrated the SP detector technology. Unlike DUNE, these are both located on the surface ensuring a high rate of cosmic ray interactions. They are also located at the end of a beamline able to provide beam particles with energies from 0.5 GeV to 7 GeV to perform detector, physics and calibration studies. ProtoDUNE-SP was completed in September 2018 and collected beam data until December 2018. During this period it was triggered externally, using beam spill signals or random triggers.

After this ProtoDUNE-SP was used as a test-stand to implement and test improvements to the detector, one of which was the testing of FPGA-based TPG as a step towards FPGA-based self-triggering. ProtoDUNE-SP demonstrated the use of the FELIX board as the detector readout technology. In this section, the ProtoDUNE-SP detector design will first be explained followed by the setup for testing FPGA-based TPG.

3.6.1 ProtoDUNE-SP Detector

ProtoDUNE-SP has a total mass of 770 t of LAr and is the largest monolithic LArTPC built to date but still only holds almost 1/20 of the total LAr mass which will be in a DUNE SP FD module. However, it allows the prototyping of detector components at a 1:1 scale in preparation for building the SP DUNE FD module.

The geometry of ProtoDUNE-SP leads to an active volume which is essentially a subvolume of the active volume of a DUNE SP module. It is 7 m deep along the z axis, 6 m high along the y axis and 3.5 m wide along the x axis for each drift volume (of which there are 2). There are 6 APAs, arranged in 2 planes of 3 which are on each side of the detector parallel to the z axis, as can be seen in figure 3.6. Between the anode planes is a cathode plane held at $-180\,\mathrm{kV}$ to achieve the same $500\,\mathrm{V\,cm^{-1}}$ electric field that will be present in DUNE.

3.6.2 FPGA-based TPG Testing Setup

To test and validate TPG, both the original ADC data from the WIBs and the TPs which were output to the FELIX host from the FPGA needed to be concurrently published and saved to disk. This was done for short test periods and used to validate the TPG using ProtoDUNE TPC data. These TPs were not stitched back together (as would occur in the FELIX host for DUNE) because this setup was designed to test the TPG firmware set out in section 3.5.

[5] Image reproduced under Creative Commons BY-SA 4.0 license https://creativecommons.org/licenses/by-sa/4.0/

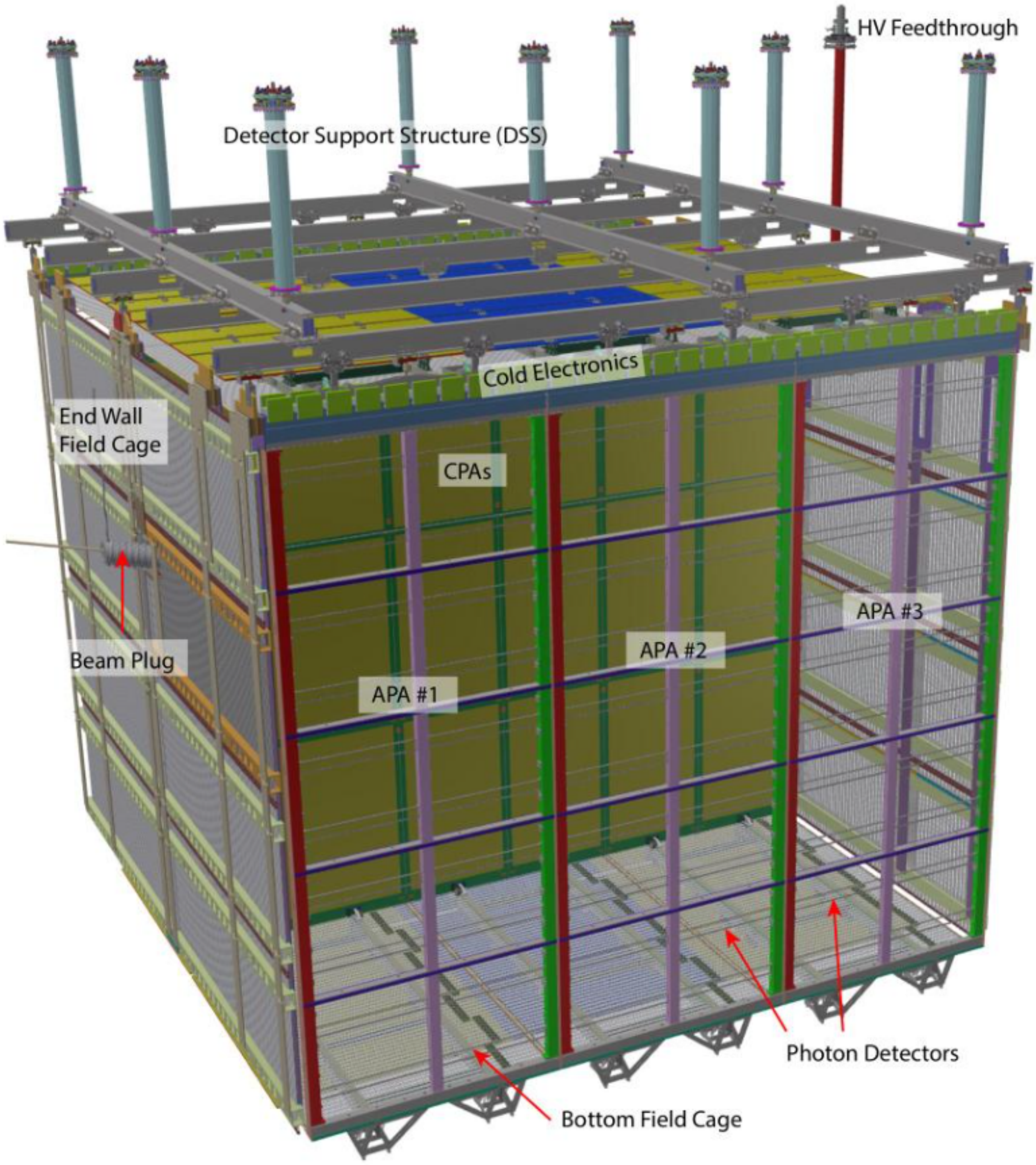


Figure 3.6: ProtoDUNE-SP detector which acts as a prototype for the DUNE SP FD. The test beam enters via the beam plug. The APAs are mounted to the detector support structure with a plane of 3 either side of the CPA plane in the centre of the active volume. Image reproduced from [111]^5.

Chapter 4

Validation Of Trigger Primitives

The firmware architecture, data flow of the data processing functional blocks and concepts of an 'ADC processing packet' and a 'hit packet' were introduced and described in section 3.5. The firmware was developed as a series of blocks, each implementing a stage of data processing and requiring validation. It was vital to be able to verify that the logic which was being applied to a test vector performed the intended task throughout the DAQ development cycle. Each of these blocks was validated through comparison with an emulation of the same algorithm implemented in stand-alone software. When using the same test vector, the outputs from the software emulation and the firmware were required to be identical to validate that the firmware performed correctly. The writing of firmware to program the FPGA to carry out these operations, including firmware testbenches and the integration of the firmware in ProtoDUNE was done by collaborators in the UD group. Collaborators also recorded the ProtoDUNE data which has been used for validation. A gold-standard implementation of the TPG algorithms in LArSoft was created by myself. A software suite for bit-wise validation of the whole firmware architecture was collaboratively contributed to by the UD group, including significant contributions from myself.

This chapter details work that was carried out during the development of the firmware to validate the correct implementation of the TPG algorithms. Firstly, validation of the PS and FIR filtering algorithms using simulation tools is presented. The data format used for validation of TP in ProtoDUNE is then introduced. An analysis which compares the TPs collected from implementing the hit-finder firmware architecture in ProtoDUNE to the TPs found through a bitwise emulation of the TPG firmware is then presented.

4.1 Validation In Simulation

The first subblocks to be developed in the data processing functional block were those which applied the PS and FIR filtering algorithms to ADC data on a single wire. This section describes the initial validation of these algorithms using a software emulation (SWEM) of the algorithms and a simulation of the firmware (FWSIM). At this stage of development, no state save/restore functionality was present in the firmware. These subblocks do not perform the operation of creating ADC processing packets from the test vector and, as such, will operate on as many ADC samples as are provided in the test vector.

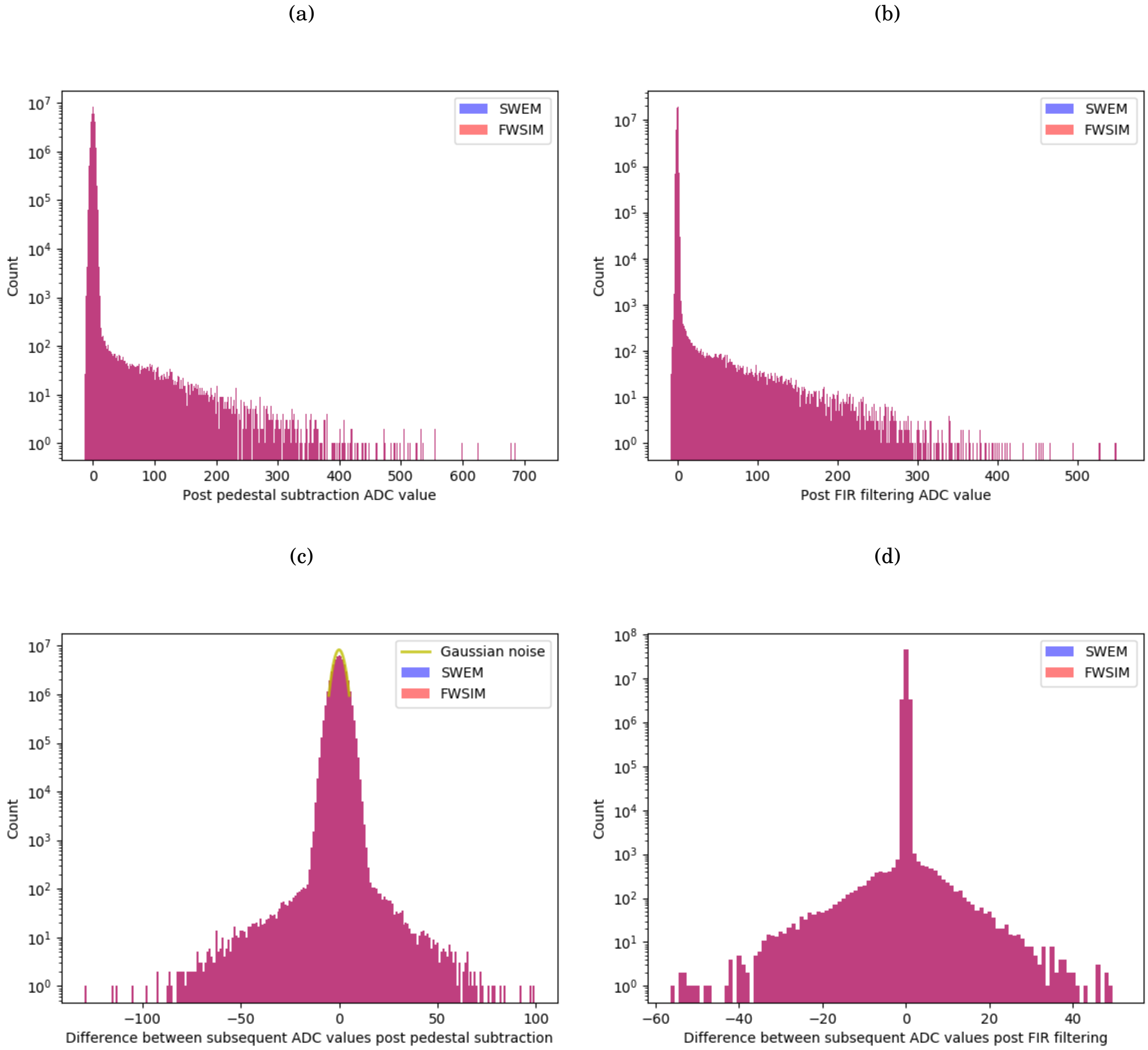
A testbench, which was run using the Xilinx Vivado Design Suite [112], simulated the Very High-Speed Integrated Circuits Hardware Description Language (VHDL) code which defined the logic for these processes. Software emulation was carried out using standalone software tools developed in DUNE's LArSoft based offline software framework. The test vector which was used for this validation test was a 1 GeV electron event generated with its momentum parallel to the z axis of the detector (along the beam axis). To allow a high statistics test on a single wire, the ADC samples from each collection wire were concatenated (for all 12 APAs in the detector simulation geometry) on to a single wire, resulting in a total of 51 million ADC samples which were used to validate the implementation of these algorithms. Noise was added to the simulated ADC samples using a Gaussian distribution with a standard deviation of 2.5 ADC counts.

4.1.1 Pedestal Subtraction

The ADC values returned by the firmware simulation and from software emulation of the PS were compared value by value and found to be identical, verifying that the outputs from software emulation and firmware in simulation agreed. This can be observed in figure 4.1(a), where the distribution of ADC values for samples is plotted for both the SWEM (blue) and FWSIM (red). The perfect overlap which results from the identical SWEM and FWSIM outputs causes the distribution to be coloured purple. From the distribution of the post PS ADCs, it is observed that the application of PS reduces the post-PS ADC value to near zero for the vast proportion of samples. The PS algorithm acts as a high pass filter which eliminates the effect of low frequency noise or pedestal drift. Low frequency noise was not present in this simulation and the high frequency noise is not filtered out. This was reflected by figure 4.1(c), in which the difference between consecutive ADC samples after PS was plotted. The subsequent samples can be seen to differ with a close relationship to the spectrum of the Gaussian noise, which has been plotted over the central peak in yellow after scaling the magnitude by the number of ADC samples.

4.1.2 FIR Filtering

The output from the PS was used as the input test vector for the validation of the FIR filtering algorithm. An exact match between the outputs from firmware simulation and software emulation



(a)

(b)

(c)

(d)

FIGURE 4.1. Validation of PS and FIR filtering algorithms on a single wire using a test vector of 51 million ADC samples from a 1 GeV electron event and added Gaussian noise with a standard deviation of 2.5 ADC counts. (a) ADC values after PS has been applied to test vector in firmware simulation (FWSIM) and in software emulation (SWEM). (b) ADC values after PS and subsequent FIR filtering have been applied to test vector in firmware simulation (FWSIM) and in software emulation (SWEM). (c) Difference between consecutive ADC values after PS has been applied to test vector in firmware simulation (FWSIM) and in software emulation (SWEM). (d) Difference between consecutive ADC values after PS and subsequent FIR filtering have been applied to test vector in firmware simulation (FWSIM) and in software emulation (SWEM).

were also found for the two implementations of the FIR filter. The distribution of the post FIR ADC values is shown in figure 4.1(b), which is seen to be similar in shape to the post PS values. The application of FIR filtering on the reduction of high frequency noise is shown in figure 4.1(d), where the difference between subsequent ADC samples after the application of FIR filtering is plotted. The distribution of ADC values is seen to narrow, demonstrating the use of the FIR filter as a low pass filter.

4.1.3 Summary

The PS and FIR algorithms were successfully validated against a gold-standard SWEM using FWSIM testbenches. At the time the PS and FIR algorithms were validated against SWEM in LArSoft, the hit-finder subblock was not yet ready. Test vectors for the validation of the hit-finding block were provided to the firmware developers who made use of these during its development. However, it was unfeasible to support the validation of anything more than data processing algorithms applied to ADC data using the LArSoft framework. Therefore, for the validation of the firmware comprising the data router, arbitrator and CR interface blocks, as well as the header stripper/combiner in the TPG block, a more formal bitwise validation technique was required.

A Python-based software suite was created for bit accurate verification of the firmware. It was designed to replicate the firmware components and the interfaces between them. Python classes were used to represent data at each stage of the hit finder firmware architecture and facilitate the conversion between different data formats throughout the processing pipeline. This was used to create test vectors for each firmware block for signals with a range of complexities. These were distributed to the firmware developers.

4.2 Hit Packet Structure

The TPs which resulted from the firmware TPG chain were stored in ‘hit packets’. The hit packet contains the header information for the ADC processing packet and contains the TPs which resulted from it. This data was stored in a ‘header frame’ and one or more ‘hit frames’, whose formats are shown in words 2-7 and 8-13 respectively in figure 4.2. For validation of the TPG an additional ‘pedestal frame’, as defined in words 14-19 in figure 4.2, was required. References to hit packet in the remainder of this chapter will include the pedestal frame implicitly.

Hit packets were the output of the TPG block. Multiple hit packets were output to a FELIX network buffer to form a ‘network packet’. The buffer was parsed and published by an onhost-BoardReader (BR) process which added two BR words to allow comprehension of the hit raw binary capture files. The first was a ‘magic word’ to allow the start of the hit packet to be identified. The second specified the size of the network packet which was read by the onhost-BR.

The pedestal frame contained ‘magic words’ which allowed the individual hit packets to be differentiated from each other and subsequently interpreted. They also contained the value of the pedestal and the accumulator which was retrieved from the saved state for the given wire. These pedestal and accumulator terms were introduced in section 3.4.1.1. The pedestal and accumulator values were vital to retrieve to enable perfect software emulation of the TPG. The frugal streaming [108] algorithm used for pedestal subtraction depends on all previous ADC samples processed. It was not possible to capture ADC samples from the moment that the hit finding firmware was flashed, which would be required to retrieve every ADC sample processed. Instead, by having access to the pedestal and accumulator it was possible to set the initial conditions to emulate TPG on each wire as long as at least 2 TPs were found on that wire during with the period processed. No hit packets were written if no TPs were found, depriving any wires without TPs from being validated.

Word	Bit																Data Frame
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	Boardreader Word 0: Magic Word (0xABCD)																BR words
1	Boardreader Word 1: Size (FELIX Frame size in bytes)																
2	crate No.					fibre No.			wireIndex No.								Header Frame
3	slot No.			Header Frame: Word 1: Reserved for flags													
4	Header Frame: Word 2: Time Stamp																
5	Header Frame: Word 3: Time Stamp																
6	Header Frame: Word 4: Time Stamp																
7	Header Frame: Word 5: Time Stamp																
8	Hit Frame: Word 0: Hit Start time (relative to 64 sample packet)																Hit Frame
9	Hit Frame: Word 1: Hit End time (relative to 64 sample packet)																
10	Hit Frame: Word 2: Hit Peak or Amplitude																
11	Hit Frame: Word 3: Hit Peak time (relative to 64 sample packet)																
12	Hit Frame: Word 4: Hit Summed ADC																
13	Hit Frame: Word 5: Hit Continue and other reserved flags																
14	0xF				Pedestal Frame: Word 0: median												Pedestal Frame
15	0xB				Pedestal Frame: Word 1: accumulator (twos complement)												
16	Pedestal Frame: Word 2: Magic Word (0xFEED)																
17	Pedestal Frame: Word 3: Magic Word (0xF00D)																
18	Pedestal Frame: Word 4: Magic Word (0xDEAD)																
19	Pedestal Frame: Word 5: Magic Word (0xBEEF)																
20	Boardreader Word 0: Magic Word (0xABCD)																BR words
21	Boardreader Word 1: Size (FELIX Frame size in bytes)																

FIGURE 4.2. Network packet structure. The BR words include information about the number of words published in a network packet. The network packet encloses 1 or more hit packet, each of which contain a header frame and a pedestal frame as well as 1 or more hit frames. The second set of BR words shows the start of a subsequent network packet. The pedestal term in the pedestal frame is labelled as the ‘median’.

4.3 Validation of Hits Captured In ProtoDUNE

The UD group integrated the TPG firmware with the ProtoDUNE DAQ and collected data to enable validation studies to be performed on the TPs recorded in ProtoDUNE. To validate the TPs, the raw ADCs were also saved to disk.

Beyond the hit-finding firmware architecture, some additional data processing was required to save the hits and ADCs to raw binary files. The FELIX card has two logic regions (super-logic regions (SLRs)) which each performed hit-finding on five of the ten ADC data streams. The hits computed from each of these SLRs were published to the network in an additional single data stream. A configurable BR [113] software process was executed on the host which serialised all these data streams from a direct memory access (DMA) buffer so that the TPs, ADCs and hits could be captured.

4.3.1 Study Outline

The validation study on ProtoDUNE TPs was carried out using the TPs collected with a single ADC data stream and a single TP data stream. This allowed the ADC samples from a single MGT (256 wires) to be used to validate the TPs which were produced by the hit finder firmware and the software surrounding it. To validate the TPs captured in ProtoDUNE, both the raw binary files containing the TPs and ADCs needed to be interpreted. The interpreted TPs provided the first opportunity to look for bugs by investigating whether each of the words in the hit packet were within the set of possible values defined by the TPG algorithm and firmware.

The ADC and TP captures did not contain data for exactly the same detector timestamp range, but did overlap each other. Therefore the timestamps at which they started and stopped overlapping were determined and these were used to extract the WIB packets and hit packets which had timestamps in this region.

To allow software emulation of the TPG algorithm, format conversion tools were used to convert the WIB packets into ADC processing packets. In order to correctly emulate TPG on these ADCs, the initial conditions for pedestal subtraction needed to be known. To allow the pedestal and accumulator values to be correctly set for TPG emulation, the TPs found in ProtoDUNE were interpreted and stored using a Python class. The first TP on each wire was determined and its pedestal frame was used to access the pedestal and accumulator values which were loaded from the saved state in the firmware prior to it applying PS to the ADC processing packet which produced that hit.

With the initial pedestal and accumulator values known, emulation of PS could be applied to the ADC processing packet. However, the FIR filtering causes an additional complication for emulation. Due to its symmetric tap coefficients, it effectively delays the signal by $N/2$ detector ticks, where N is the number of filter taps. This means that any hits found in software emulation in the first $N/2$ ticks will not be equivalent to those in the TP capture file. To avoid this affecting the accuracy of the validation, any TPs from this first ADC processing packet, whether emulated or from firmware, were dropped before comparison.

After TPG emulation, the emulated TPs were compared to the firmware TPs to identify potential bugs from discrepancies.

4.3.2 Investigating Raw TP Captures

A number of quantities in the header and hit frames were checked against the range of values which it was possible for them to possess. Within the header frame, the crate number was checked (used to denote the APA in ProtoDUNE that this data was collected on). The slot number was checked to be within the range $0-4$ and the fibre number was checked to be 1 or 2. These two quantities were used to track the path of the ADCs through the interface between the CE and the DAQ hardware.

The TP quantities were checked in a similar manner. Any hit start or end ticks which were > 61 or > 63 respectively were searched for in addition to any end ticks occurring in the first 2 ticks in the ADC processing packet. These values were chosen because the firmware hit finding was designed to require hits to be above the threshold for 2 or more ticks. Checks were also made that the peak ADC value was above the threshold used for hit finding and that the peak tick was within the range $0-63$.

Word 5 of the hit frame, as defined in figure 4.2, contains a flag which indicates if the TP continues up until the end of the ADC processing packet. This is stored in the least significant bit (LSB) of this word, while the remaining 15 bits are reserved for other flags. However, these were not actually defined in the firmware and should remain as zeros. Any occasion that this was not the case was checked for. Finally, the last check carried out was to identify any TPs which were flagged to be above threshold until the end of the ADC processing packet, but did not have a hit end tick of 63.

As a result of these checks, 3 different types of bugs were identified as reported in table 4.1. An example of each is shown. To verify that these were present in the raw binary files, they were hexdumped into 32 bit words and examples of TPs possessing these bugs were identified. These 32 bit words are presented in hexadecimal representation in figures 4.3(a)-4.3(c), which contain all words from the start of the header frame to the end of the pedestal frame. They can be interpreted with the hit frame structure in figure 4.2. The subsequent 16b words which make up a 32b word are arranged from left to right.

Out of a total of $\sim 11 \times 10^6$ TPs that were collected in the TP binary file, a total of $\sim 2 \times 10^5$ possessed one of these bugs (1.85%). The 'Continues Flag and End Tick' bug shows that there were occasions where the hit continue bit was toggled to 1 but the last tick with an ADC sample over the threshold was not the last tick in the ADC processing packet. The 'Hit Flags' bug indicates that the 15 most significant bits (MSBs) in the 5th 16 bit word of the hit frame were non-zero.

Lastly, the 'No Peak/Sum' bug is present where the peak ADC sample, peak ADC tick and the summed ADC (SADC) are all zero.

Any TPs which possessed any of these bugs were removed from the set of TPs carried forward for validation with software emulation.

TABLE 4.1. The 3 different bug types that were identified by interpreting all hits collected for 5 hit finder links. An example of each type of bug extracted from the raw hit capture file is presented.

Bug Type	Hit Start Tick	Hit End Tick	Hit Peak ADC	Hit Peak Tick	Hit Summed ADC	Hit Flags 15b	Hit Continues Flag
Continues Flag and End Tick < 63	51	62	337	57	3635	0	1
Hit Flags ! = 0	44	58	123	51	1748	30	0
No Peak/Sum	59	63	0	0	0	0	1

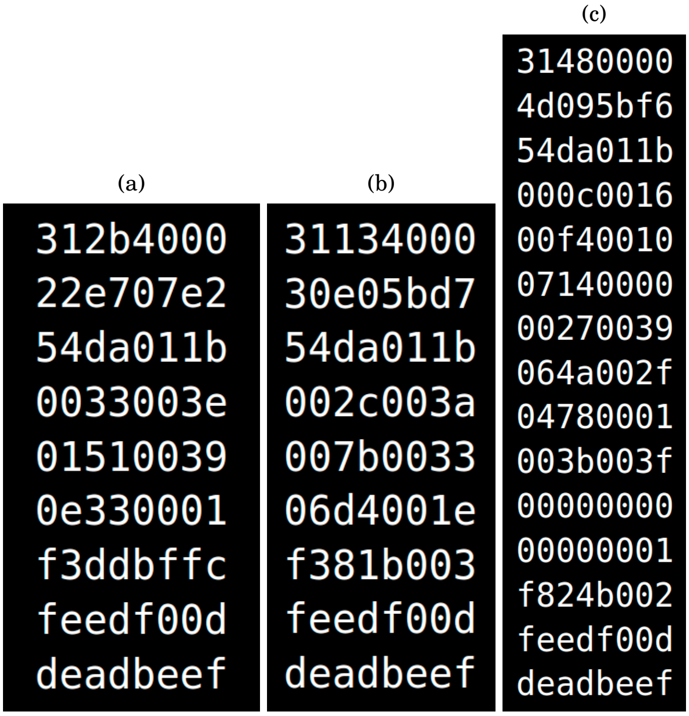
4.3.3 Comparison of Captured Firmware Hits and Software Emulated Hits

The procedure used to emulate the firmware TPs using the Python software suite was described in section 4.3.1. In this section, the selection of the firmware TPs used in this study is described. This is followed by a description of the process of pairing the firmware and emulated TPs. These paired TPs are compared and successive selections based on the most commonly observed dissimilarities are made. The possible causes of each dissimilarity are discussed before the paired TPs in that selection are removed and the next selection is made. Finally the proportion of pairs which perfectly match each other is reported.

4.3.3.1 FW Hits Used In Validation Study

There were $\sim 11 \times 10^6$ TPs which were present in the binary capture, which records TPs from five raw ADC data streams. A single ADC data stream and the TPs captured from it in ProtoDUNE were used in this study. TPs which did not originate from this ADC data stream were discarded along with any TPs which did not occur during the timestamp region where the captured ADC stream and captured TPs overlapped. TPs which contained bugs identified in section 4.3.2 were also discarded.

The number of remaining firmware TPs was 937, which occurred on 198 of the 256 wires being recorded in the data stream. As the first TP on each wire was used to set the pedestal and accumulator values, this left a remaining 739 TPs to use in the validation study. The total number



(a)

```
312b4000
22e707e2
54da011b
0033003e
01510039
0e330001
f3ddbffc
feedf00d
deadbeef
```

(b)

```
31134000
30e05bd7
54da011b
002c003a
007b0033
06d4001e
f381b003
feedf00d
deadbeef
```

(c)

```
31480000
4d095bf6
54da011b
000c0016
00f40010
07140000
00270039
064a002f
04780001
003b003f
00000000
00000001
f824b002
feedf00d
deadbeef
```

FIGURE 4.3. (a) 32 bit word representation of the example 'Continues Flag and End Tick' bug reported in table 4.1. (b) 32 bit word representation of the example 'Hit Flags' bug reported in table 4.1. (c) 32 bit word representation of the example 'No Peak/Sum' bug reported in table 4.1.

of TPs generated through software emulation was 868. Bugs in the firmware implementation could result in some TPs not being identified, which may account for why there were 129 fewer firmware TPs found than emulated TPs. The low statistics were primarily a result of the ADC data stream being captured for a shorter time than the TP data stream. The timestamp range of the raw ADC capture was 0.22% of the timestamp range of the captured TPs and equates to 5.48 milliseconds of recording.

4.3.3.2 Pairing Of Firmware and Emulated Hits

To make a comparison between the emulated TPs and the firmware TPs, it was necessary to assign emulated and firmware TPs to each other based on a TP property. The same emulated TP could be paired to multiple firmware TPs but each firmware TP was paired to only a single emulated TP. A pair of assigned TPs are referred to as 'paired' TPs. If all of the TP properties are identical, they are referred to as a 'perfect pair'. If any of their properties differ, they are referred to as a 'nonperfect pair'.

The approach taken was to assign the firmware and emulated TPs which occurred closest together in time to each other. Then the properties of these paired TPs were compared and used to identify a selection of paired TPs which were nonperfect pairs. The potential origins of these nonperfect pairs were investigated and this selection was then removed. The process was repeated until there were no more significant nonperfect pairs remaining.

The proportion of emulated TPs for which a pair remained, η_m, is reported after the TPs identified in each selection were removed:

$$\eta_m = \frac{N_m}{N_{SWEM}}, \tag{4.1}$$

where N_m is the number of paired TPs and N_{SWEM} is the number of emulated TPs used in the validation study.

The starting tick used for pairing TPs was calculated from the combination of the hit packet timestamp and the hit start tick within the hit packet. The difference between the starting ticks (S) for the software emulated and firmware TPs is $\Delta T = |S_{SWEM} - S_{FW}|$. For each firmware TP, the emulated TP on that wire with the smallest absolute value of ΔT was paired to it. As can be seen from a comparison of the starting ticks in figure 4.4(a), some of the pairs were very poor, occurring thousands of ticks apart. However, there is a central peak around zero, which indicates that many of the pairs may be close. Looking more closely at this central region in figure 4.4(b), the majority of close pairs were observed to occur within ±20 ticks. As a result, the first cut to be applied to the paired TPs was that pairs were required to have start ticks which were within 20 ticks of each other. It was assumed that paired TPs which were more distant than this were not directly related to one another and that this was probably a result of the firmware not finding a TP which it should have. After removing all pairs which did not meet this requirement, $\eta_m = 77.5\%$, with 673 remaining pairs.

There was a clear disparity between the summed ADC values of the paired software emulated TPs and the paired firmware TPs, as illustrated in figure 4.5. The summed ADC values from the firmware TPs were found not to exceed 4095. There was a strong linear correlation between the firmware and software emulated summed ADC values up to values of 4095. For software emulated TPs with summed ADC values larger than 4095, this linear correlation appears to continue as if the summed ADC value in the paired firmware TPs is 12 bit masked. The lack of any firmware TPs with summed ADC values larger than 4095 despite there being emulated TPs with larger ADC values and 16 bits allocated to store the summed ADC value in the firmware, appeared to indicate that there was a bug in the firmware implementation.

4.3.3.3 Selection 1

The TP properties after discarding pairs with $\Delta T > 20$ are shown in figure 4.6(a)-4.11(a). Each of these figures corresponds to a particular hit property and the other sub-figures present are the result of removing the paired TPs from each successive selection. These plots comparing the distributions of the properties of the remaining paired TPs were used to identify off-diagonal populations. The first of these selections, which will be referred to as selection 1, was identified

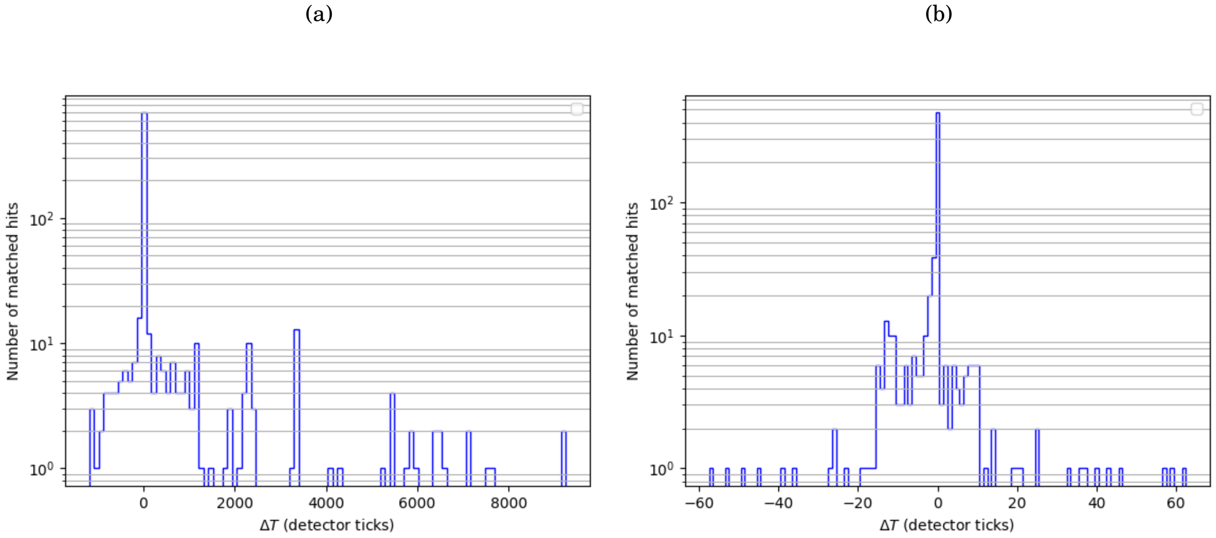


FIGURE 4.4. (a) The start tick of the paired firmware TPs subtracted from the start tick of the emulated TPs. (b) The same plot as figure 4.4(a) centred on the central 128 ticks, equivalent to the number of ticks in 2 ADC processing packets.

from figure 4.7(a) and contained 43 TPs, lowering η_m to 72.6%. Paired TPs which had firmware end ticks equal to 63 and software emulated end ticks lower than 25 were classified as members of selection 1. The effect of removing selection 1 can be seen from the distribution of each hit property in figure 4.6(b)-4.11(b). When selection 1 was removed, it became apparent from the paired TPs removed between figure 4.6(a) and figure 4.6(b) that the emulated TPs had start and end ticks which were within the first 25 ticks, whilst the firmware TPs had start and end ticks which were all later than tick 40. The start ticks of the emulated TPs were mostly zero and the end ticks of the firmware hits were mostly 63. So the mismatched TPs in selection 1 always occurred at different times in an ADC processing packet and this was evidence that the paired TPs were related to signals which stayed above threshold over ADC processing packet boundaries. It was found that the emulated hits in selection 1 all occurred one ADC processing packet after the firmware TP which they were paired to. This was consistent with the other strange phenomenon present in the paired hits, which was that no firmware hits had a start tick before the 10th tick in the ADC processing packet. The existence of a bug which causes the TPG to be unable to find TPs at the start of an ADC processing packet would lead to cases where an emulated TP which continues over a packet boundary for a short time can only be paired to the firmware hit at the end of the previous ADC processing packet. Indeed, the paired hits which have been removed between figure 4.8(a) and figure 4.8(b) were all 25 ticks or shorter.

By insisting that TPs had to come from the same ADC processing packet, the 43 TPs in selection 1 and 3 additional TPs not in selection 1 were removed, with these 3 additional discarded TPs reducing the η_m to 72.6%.

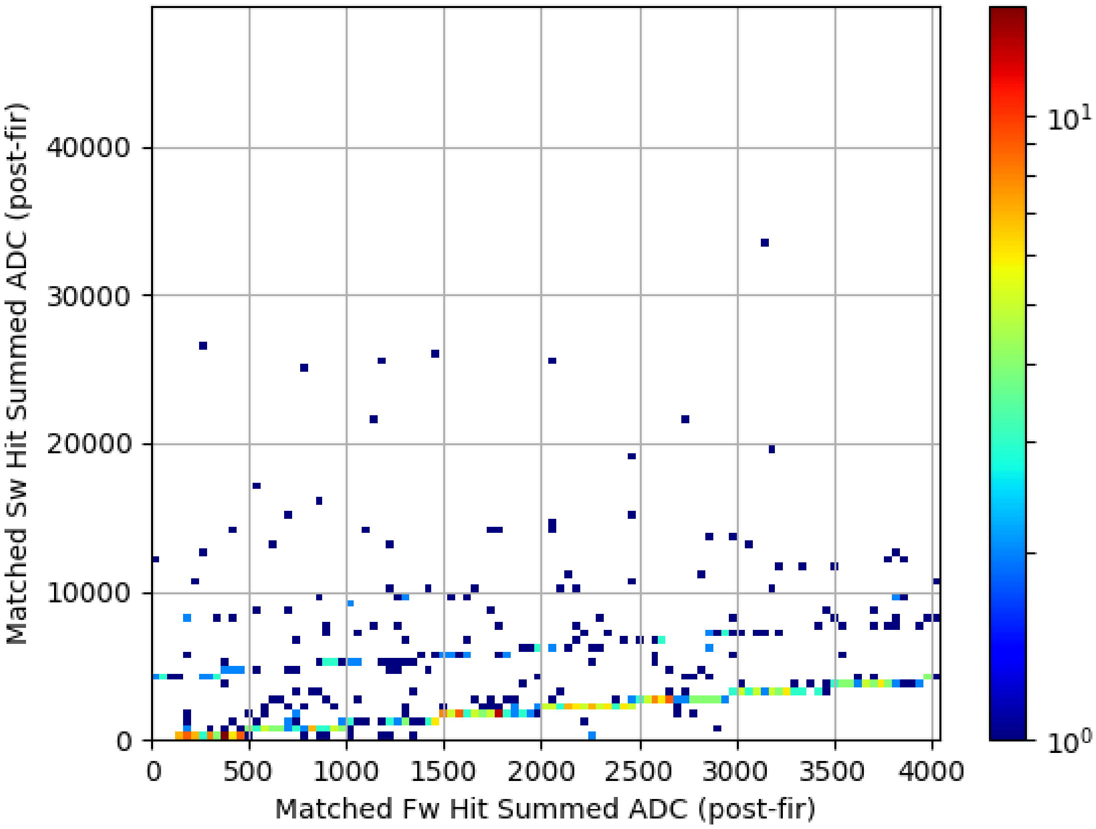


FIGURE 4.5. Comparison between the summed ADC of the paired TPs. Firmware hits had summed ADC values limited to the range 0-4095.

4.3.3.4 Selection 2

However, after the removal of selection 1, remaining paired TPs were not all perfect pairs. Therefore, a second population of nonperfect pairs, labelled selection 2, was identified. Selection 2 was made from any paired TPs whose emulated peak ADC value was greater than the peak ADC value of the firmware TP. This selection consisted of a further 105 paired TPs and reduced η_m to 60.1% . Removing paired TPs in selection 2 had the effect of removing paired TPs which were far closer to the central line of equivalency than the removal of selection 1 did. This effect was pronounced in figure 4.6(c), where many paired firmware TPs with start ticks below 20 which were paired to emulated TPs with earlier start ticks have been removed as a result. This is consistent with the paired hits which were removed between figure 4.9(b) and figure 4.9(c), which appear to indicate that the peak ADC tick of the paired emulated hits in selection 2 was earlier than the firmware TP, but not necessarily always earlier than the start tick of the firmware TP. Peak ticks are seen to be consistent between paired TPs only after the 20$^{\text{th}}$ tick in the packet. A similar story was true for the impact of the removal of selection 2 on the distribution of paired end ticks in figure 4.7(c). The vast majority of paired TPs in selection 2 had more ticks over threshold (TOT) in the emulated TP than the firmware TP, as can be seen from figure 4.8(b) and figure 4.8(c), again suggesting that the major reason for the discrepancy in peak ADC value was due to the firmware being unable to detect TPs occurring early in a packet.

4.3.3.5 Selection 3

A third and final population of nonperfect pairs was identified from figure 4.6(c) where the firmware start tick was later than the software emulated start tick and composed of a further 47 pairs, lowering η_m to 54.7%. Apart from looking at the start ticks of paired TPs, the effect of removing this 3rd selection could be seen most clearly in the difference between figure 4.8(c) and figure 4.8(d). It removed most of the remaining paired TPs where the emulated TP lasted longer than the firmware TP. Selections 1, 2 and 3 are all consistent with having resulted from a bug in the implementation of TPG in firmware which did not allow TPs in the first part of the ADC processing packet to be detected. Upon insisting that the emulated and firmware TPs pair exactly (except for the summed ADC due to the inconsistency identified earlier), then η_m became 51.8%.

4.3.4 Conclusions

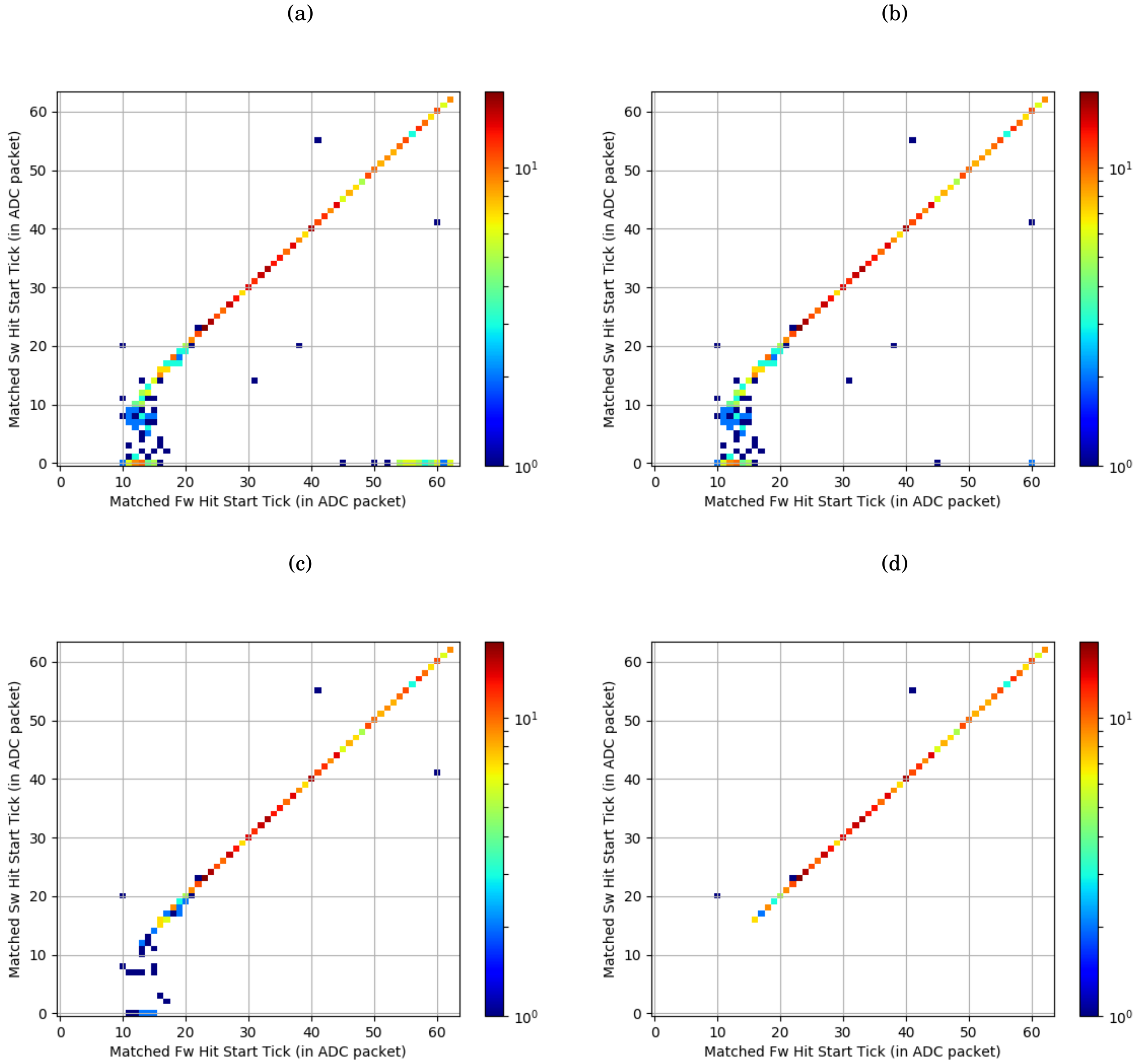
This chapter has explained the work done to validate the implementation of TPG in firmware and has set out the scope of this process throughout the development of this firmware. An example of an initial study was provided and the data collection process for capturing firmware generated TPs online using ProtoDUNE was explained, along with the resulting network packet format. Potential bugs in the firmware implementation were identified through both looking at the properties of a high number of captured TPs as well as running a bitwise emulation of the firmware TPG and comparing the resulting TPs to those found in ProtoDUNE for a known set of initial conditions and captured raw ADC data. The bugs identified from looking at TPs in bulk were listed in table 4.1 and are:

1. TPs with a hit continues flag but an end tick before the end of the ADC processing packet

2. TPs with a non-zero entry for their hit flags term

3. TPs with a peak ADC value, peak ADC tick and summed ADC value all of 0.

These bugs were present in 1.85% of TPs. An additional couple of bugs were identified through software emulation. The first was that the summed ADC values of firmware TPs were never larger than 4095 and the second was that there were no firmware TPs with start ticks in the first


(a)

(b)

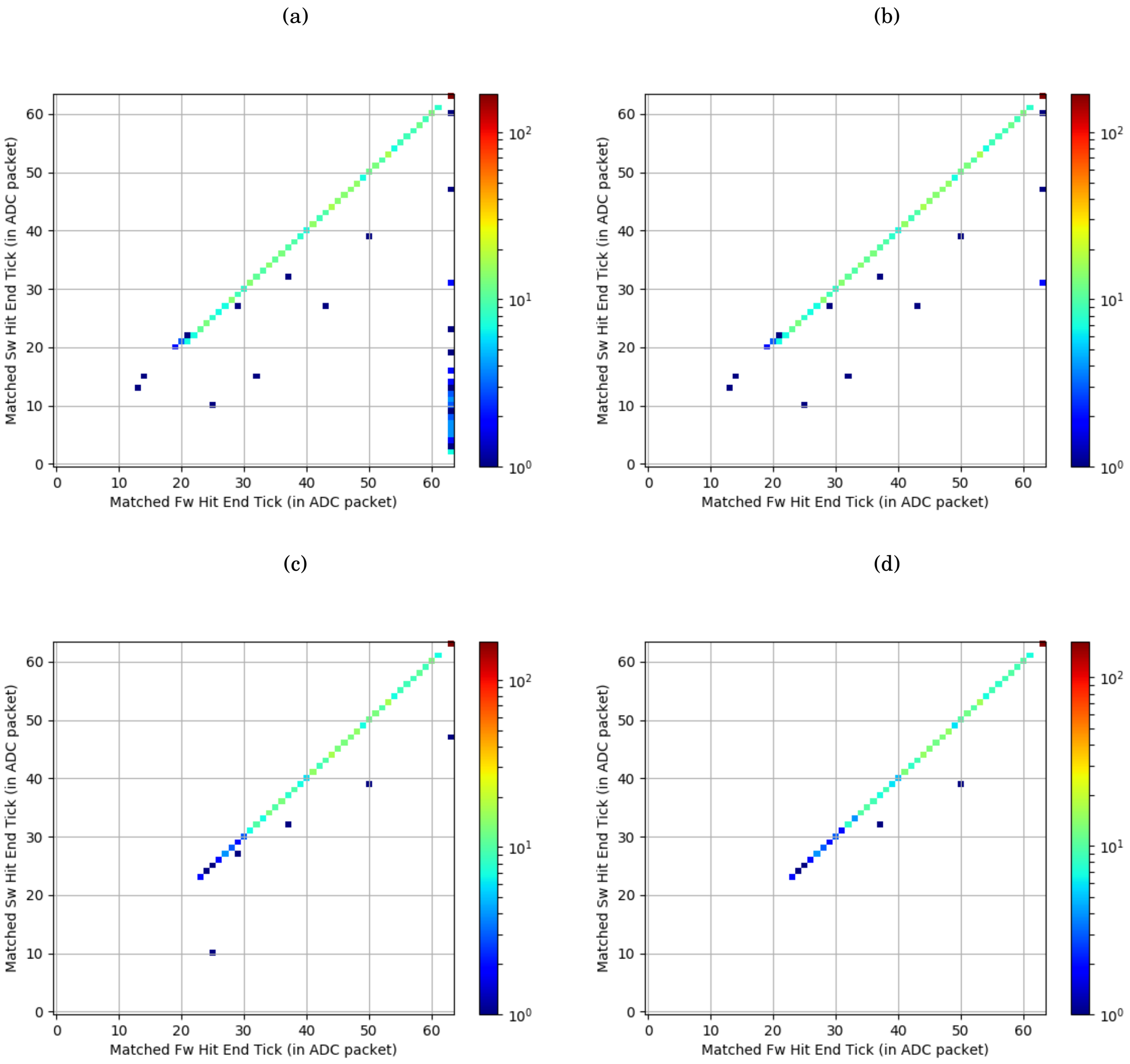
(c)

(d)

FIGURE 4.6. The start tick within the ADC processing packet of paired firmware and software emulated TPs for: (a) pairs with $\Delta T < 20$ ticks; (b) pairs remaining after also removing selection 1 ; (c) pairs which remain after additionally removing selection 2 ; (d) pairs remaining after further removing selection 3.

several ticks of any ADC processing packet. When paired TPs were required to be perfect pairs, η_m was determined to be 51.8%.

These bugs were reported to the firmware developers, who managed to identify probable causes for some of them and implement fixes. The lack of firmware TPs in the first few ticks of a packet was found to be a result of a local reset problem. The local reset was occuring a clock cycle too early, which interfered with the state save/restore mechanism for the FIR subblock. The bug where TPs had peak ADC values, peak ADC ticks and summed ADC values of 0 was identified to


(a)

(b)

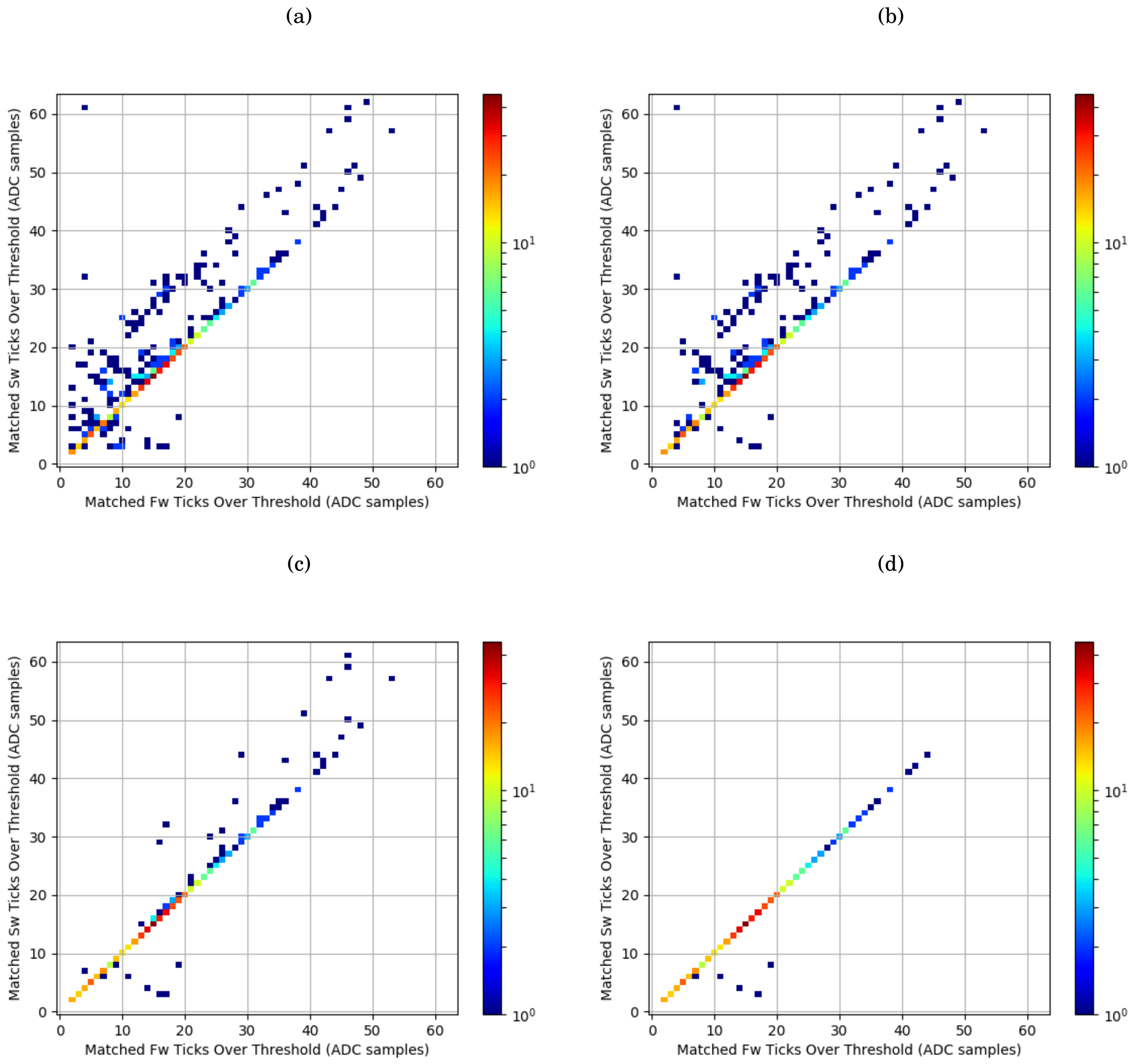
(c)

(d)

FIGURE 4.7. The end tick within the ADC processing packet of paired firmware and software emulated TPs for: (a) pairs with $\Delta T < 20$ ticks; (b) pairs remaining after also removing selection 1 ; (c) pairs which remain after additionally removing selection 2 ; (d) pairs remaining after further removing selection 3.

likely be a corner case due to multiple TPs in a packet. Since the HF subblock needed 6 clock ticks to output the TP properties, if a second TP occurred within this time frame the hit packet output could be a result of improper combination of two hit frames. This was fixed by buffering the TPs found in each packet before sending them to the output and requiring the minimum number of ticks to be above threshold for a TP to be increased to 4. This constrained the maximum number of TPs per packet so that memory resources could be guaranteed to be sufficient.

For the bug where the 15 MSB of the hit continues flag were not 0, no obvious cause was


(a)

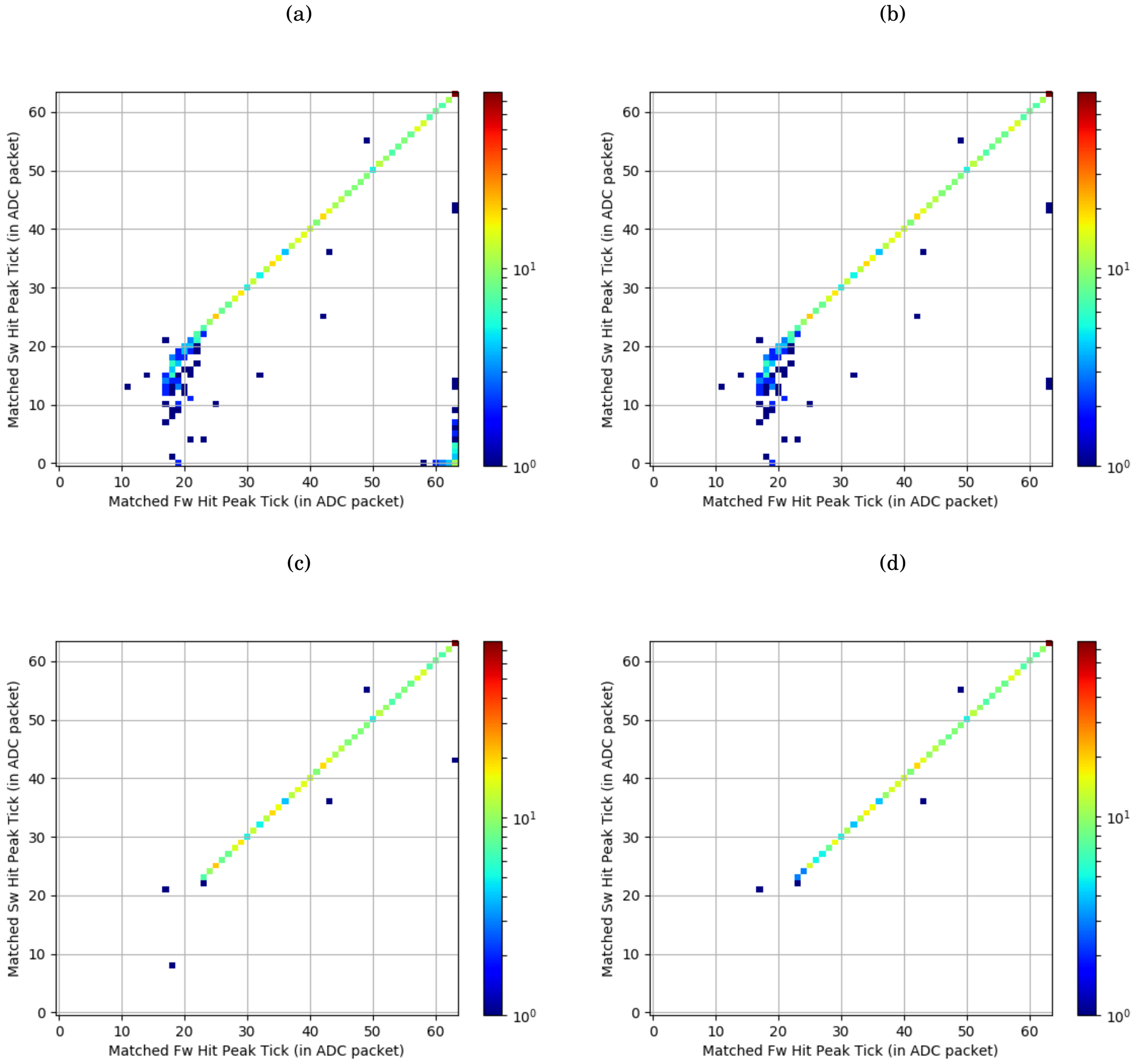
(b)

(c)

(d)

FIGURE 4.8. The number of TOT within the ADC processing packet of paired firmware and software emulated TPs for: (a) pairs with $\Delta T < 20$ ticks; (b) pairs remaining after also removing selection 1 ; (c) pairs which remain after additionally removing selection 2 ; (d) pairs remaining after further removing selection 3.

found. The same was true for the bug where the LSB of the hit continues flag was 1 but the TP did not last up to the end of the packet. These could not be reproduced using the simulation tools or feeding test data patterns through a ZCU102 board. There will be the opportunity to repeat validation tests similar to those performed on ProtoDUNE data and presented in this chapter when the second iteration of the ProtoDUNE-SP detector is commissioned in early 2022. These will be an essential step towards validating that the FPGA-based TPG performs correctly in experimental conditions to prepare for its use in the DUNE SP FD module.



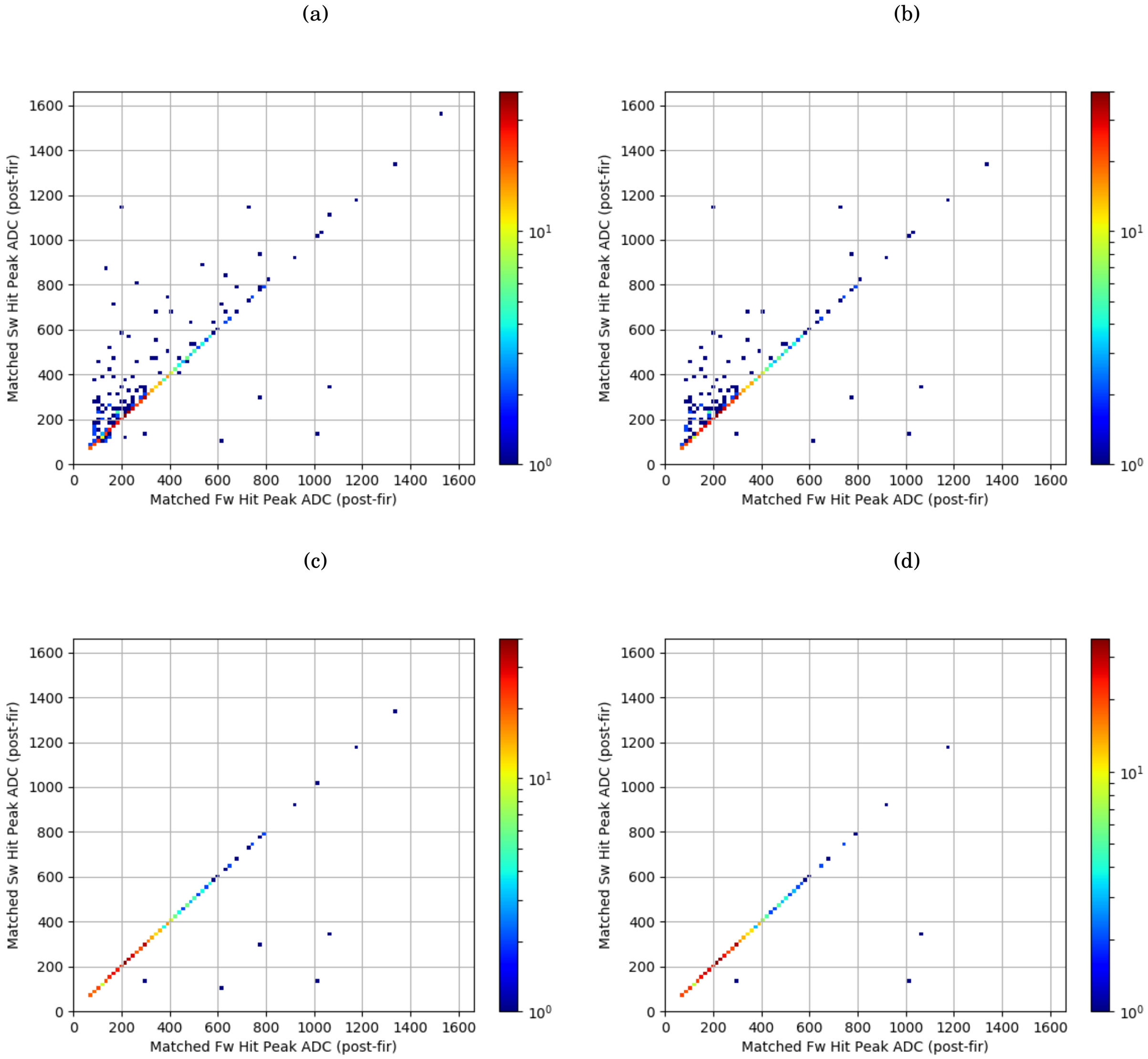
(a)

(b)

(c)

(d)

FIGURE 4.9. The peak tick within the ADC processing packet of paired firmware and software emulated TPs for: (a) pairs with $\Delta T < 20$ ticks; (b) pairs remaining after also removing selection 1 ; (c) pairs which remain after additionally removing selection 2 ; (d) pairs remaining after further removing selection 3.

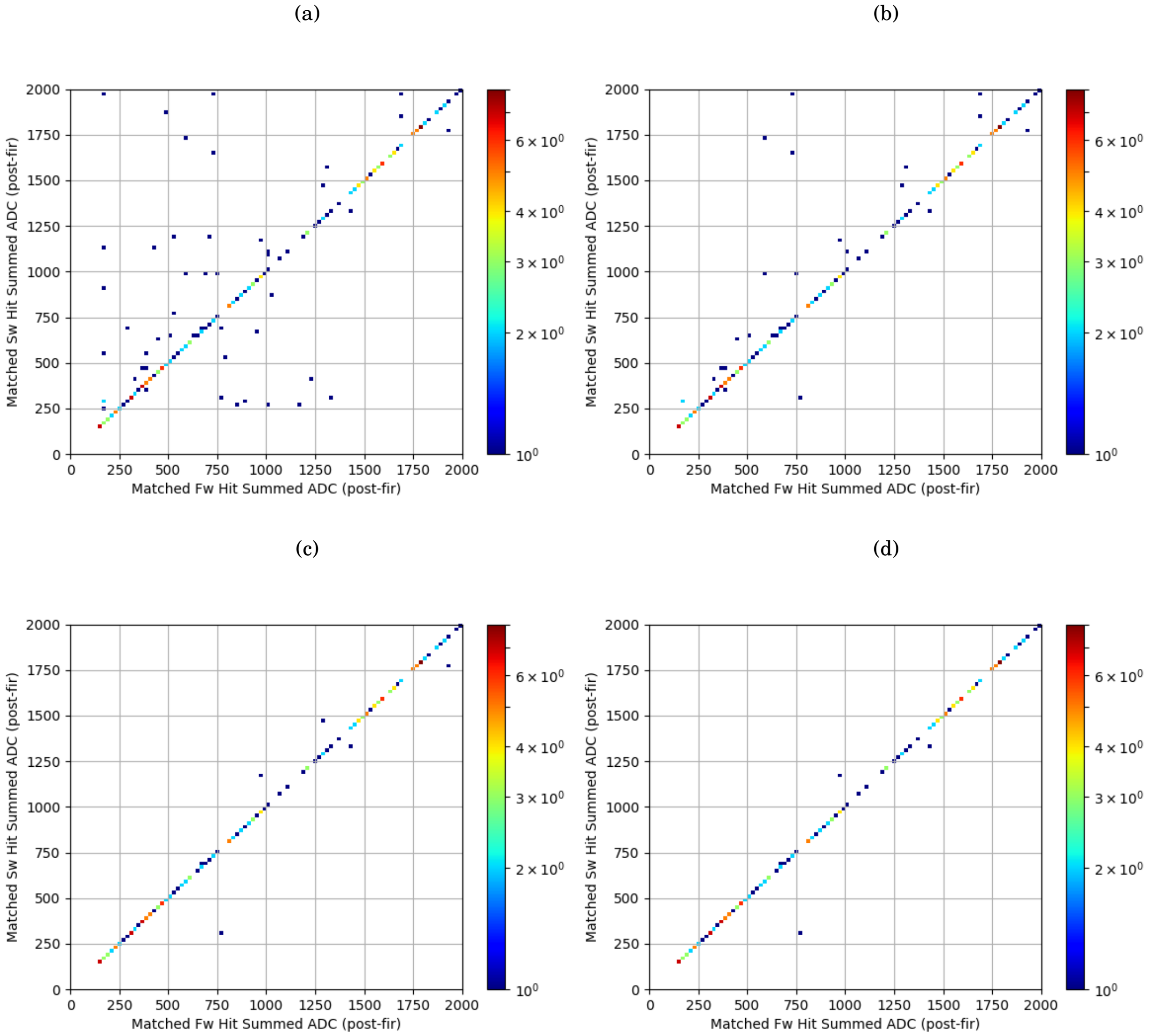

(a)

(b)

(c)

(d)

FIGURE 4.10. The peak ADC value of paired firmware and software emulated TPs for: (a) pairs with $\Delta T < 20$ ticks; (b) pairs remaining after also removing selection 1 ; (c) pairs which remain after additionally removing selection 2 ; (d) pairs remaining after further removing selection 3.


(a)

(b)

(c)

(d)

FIGURE 4.11. The summed ADC of paired firmware and software emulated TPs for: (a) pairs with $\Delta T < 20$ ticks; (b) pairs remaining after also removing selection 1 ; (c) pairs which remain after additionally removing selection 2 ; (d) pairs remaining after further removing selection 3.

Chapter 5

Baseline Supernova Burst Trigger Performance

As described in section 3.3.4 the DAQ needs to meet the following requirements:

- a data volume of 2PB per year of TPs to be saved to disk from a 10 kt detector module
- an average SNB false trigger rate of 1 per month
- a > 95% expected efficiency for a SNB with at least 60 interactions, each of a minimum 10 MeV in true neutrino energy.

A study on the performance of a similar SNB trigger has been completed by DUNE collaborators on TPs generated using a similar TPG algorithm designed to be implemented in CPU instead of a FPGA [85]. This algorithm was found to have a SNB trigger efficiency of 100% in the Milky Way, dropping to an efficiency of 8% at the LMC [114]. The solar system is 8.2 kpc from the centre of the Milky Way galaxy [115] and the distribution of expected supernovae in the Milky Way as a function of galactocentric distance falls to near zero beyond 18 kpc [116]. The LMC is located at a distance of 49.6 kpc [117].

The reported SNB trigger efficiencies from their study were the highest achieved through a grid search over clustering parameter values. The optimal clustering parameters found were described in section 3.4.2.

The efficiency of the SNB trigger in the CPU-based TPG study was determined by first finding the threshold on the number of TCs in a 10 second window required to achieve a false SNB trigger rate lower than a 1 per month. A Poisson distribution was assumed to describe the rate of background clusters. For a SNB expected to produce N neutrino interactions in the detector (ie: occurring at a given distance), this number of interactions was combined with the overall clustering efficiency to create an expected number of neutrino interactions to use as the mean rate of another Poisson distribution. This was then used to determine the probability that the number of interactions detected met the required SNB trigger threshold.

This chapter describes a study of the performance of the baseline SNB trigger using FPGA-based TPG. There are small differences between the TPG algorithm parameters used for the CPU-based TPG study and the FPGA-based algorithm. The CPU-based TPG used a 7 tap FIR filter whose coefficients summed to 100 [118] and a hit-finding threshold of 16 ADC counts. The FPGA-based TPG used a 16 tap FIR filter whose coefficients summed to 144. Its hit-finding threshold was configurable and was chosen to be that which limited the data volume of TP to 2 PB per year. It is possible to filter which TPs are saved to disk by their properties (such as a threshold on H_P) but the selection of a hit-finding threshold via this method allows a reasonable lower limit on the SNB trigger efficiency to be determined. The optimal clustering parameters found in the study on CPU-based TPs were re-used to cluster the FPGA-based TPs in the studies presented in this chapter. The false trigger rate and efficiency of the SNB trigger is evaluated using a Monte Carlo method.

The DUNE production team simulated a dataset of neutrino interactions and background decays in the DUNE FD. Code emulating the FPGA-based TPG has been written and run on this dataset to produce TPs. Firstly, the details of the dataset and its simulation are described. This is followed by characterisation of the TPs and the subsequent TCs. Finally, the Monte Carlo method is explained and the SNB trigger efficiency is reported.

5.1 Monte Carlo Simulation

The LArSoft toolkit [119] is designed to allow the accurate simulation of physics in LArTPC neutrino experiments. It was used to simulate supernova neutrinos, radiological backgrounds and detector noise in the DUNE FD. This simulation was performed in the following steps, which are each described in their own subsection. Firstly, lists of primary particles from neutrino interactions and radioactive decays within the detector geometry (and any modelled surroundings) were simulated. The interactions of these particles within the detector volume were then simulated using GEometry ANd Tracking 4 (GEANT4) software [120], followed by a simulation of the detector response.

5.1.1 Primary Interaction

The detector geometry used in simulations was the $1 \times 2 \times 6$ geometry, which corresponded to the number of APAs along the x, y and z detector axes respectively. This allowed the simulation of 12% of the volume of a single FD module. This allowed the simulation to be more computationally tractable as the number of objects and wires in the simulation was reduced but meant that there were limitations on the effectiveness of simulating external backgrounds. The APAs in the $1 \times 2 \times 6$ geometry corresponded to a sub-region of the central anode plane in the detector. As a result, neutrons from spontaneuous fission of ^{238}U in the concrete of the detector walls are not proportionally represented. Neutrons could travel for metres before being captured, so additional sources of this background would be present in a simulation of the full detector geometry. The configuration of backgrounds which was used is known as "Model 1". The full list of backgrounds, rates and their points of origin is tabulated in table 5.1.

Table 5.1: Types of radioactive decay present in the simulation for the "Model 1" background configuration and their origins in the detector.

Radioactive decay	Decay rate	Decay origin
^{39}Ar beta decay	1.41 mBq cm^{-3}	liquid argon
^{42}Ar beta decay	128 nBq cm^{-3}	liquid argon
^{85}Kr beta decay	0.16 mBq cm^{-3}	liquid argon
^{222}Rn decay chain	55.8 µBq cm^{-3}	liquid argon
^{222}Rn decay chain	5 µBq cm^{-3}	photon detector surface
^{60}Co beta decay	82 µBq cm^{-3}	anode plane assembly (APA) frame
^{40}K beta decay	2.72 mBq cm^{-3}	cathode plane assembly (CPA) frame
^{238}U spontaneous fission	7.60 µBq cm^{-3}	concrete

The drift time between the anode and cathode planes was 2.25 ms. Neutrino interactions were generated with a uniform distribution in the detector volume. The detector was simulated for a drift time either side of the time at which the neutrino interactions were generated, $t = 0$. This allowed the radiological decay signals to travel an entire drift length before the simulation began to record signals at $t = 0$. Each of these simulated time periods is termed an ‘event’ in LArSoft.

The primary particle lists corresponding to neutrino interactions were generated using the Model of Argon Reaction Low Energy Yields (MARLEY) event generator [121] which is specifically designed for the simulation of low energy (several MeV) neutrino interactions and expected to be accurate for neutrinos with energies up to ~ 50 MeV. In this dataset only the products from the charged current interaction in equation 2.17, the dominant cross section in DUNE, were simulated. The energies of the interacting neutrinos were drawn from the energy spectrum of the ‘Hudepohl-Cooling-Shen-s11.2co’ model [87] which corresponds to an $11.2m_{\odot}$ mass progenitor star. This is the same model which produced the spectra shown in figure 2.7.

The relationship between the energy spectrum for the incoming neutrinos and the electrons from their interaction with the argon nuclei is shown in figure 5.1. The neutrino energies range from 3 MeV to 60 MeV, with the energies of the primary electrons being linearly correlated in a series of bands. Each LArSoft event simulated 3 neutrino interactions to save on storage space.

5.1.2 Particle Propagation

The primary particles were stepped through the detector volume using GEANT4 which determined the ionisation and scintillation signal in the LAr. The individual energy depositions resulting from each propagated particle, and its daughters, were recorded so that the signals

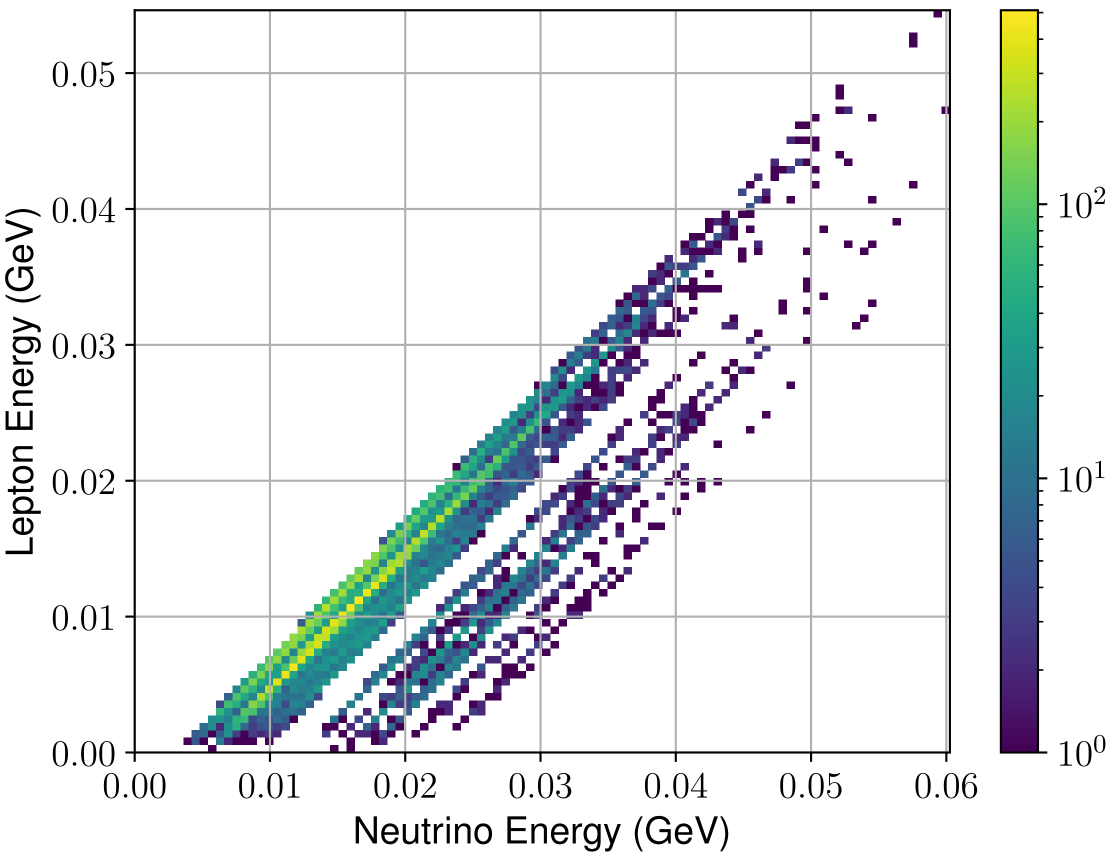


Figure 5.1: Relationship between the incoming neutrino energies and the energy of the primary electron produced from the charged current interaction, equation 2.17, when simulated with MARLEY.

could be tracked back to the primary particle. GEANT4 split charge deposits into clusters of electrons and propagated the ionisation through the LAr to the APAs, including the application of attenuation, diffusion and distortions of the electric field. It then determined the wire on the APA that each of these electron clusters was collected on. In this study only the charge deposited on collection wires was used.

5.1.3 Detector Response Simulation

The response of the detector cold electronics to these charge deposits was then simulated to determine the raw signal in ADC counts. No deconvolution or further processing was applied to this signal. It should also be noted that no detector noise was added up to this point in the simulation. The dataset of neutrino and background interactions which was simulated by the DUNE production team was provided in this state.

The standard DUNE noise model written by David Adams was then used to add noise to the raw detector response ADC counts on each wire. No coherent noise across wires was added. The details of this implementation are given in appendix A.2. After this, the default pedestal used for collection wires in DUNE, 500 ADC counts, was added.

The emulation of FPGA-based TPG was then applied to the ADC counts from $t = 0$ to $t = 2.246$ ms in the simulation. To avoid extra computation required to stitch hit packets together, instead of using a packet size of 64 ticks, a packet size including all 4492 ticks in the drift window was used. This resulted in TPs which were identical to those produced by stitching TPs with a smaller packet size together without needing to apply hit-stitching.

The TPs were tracked back to primary particles using a backtracking algorithm in LArSoft which finds the energy deposits which were deposited on the same channel as the TP over the same time window that the TP covered. The time window that was searched for these energy deposits was widened by 20 ticks at the start and end of the TP. The window was also shifted to account for the delay in the start time of a TP caused by the 16 tap FIR filter. The TP was labelled as being of an unknown origin if no ionisation was deposited on the wire during this time window and these were a result of detector noise. Otherwise, the origin of the TP was set by finding the GEANT4 track which deposited the largest proportion of the energy in the specified time window. This track ID was used to track it back to a primary particle from either a MARLEY interaction or one of the radiological backgrounds.

5.2 Characterisation of Trigger Primitives

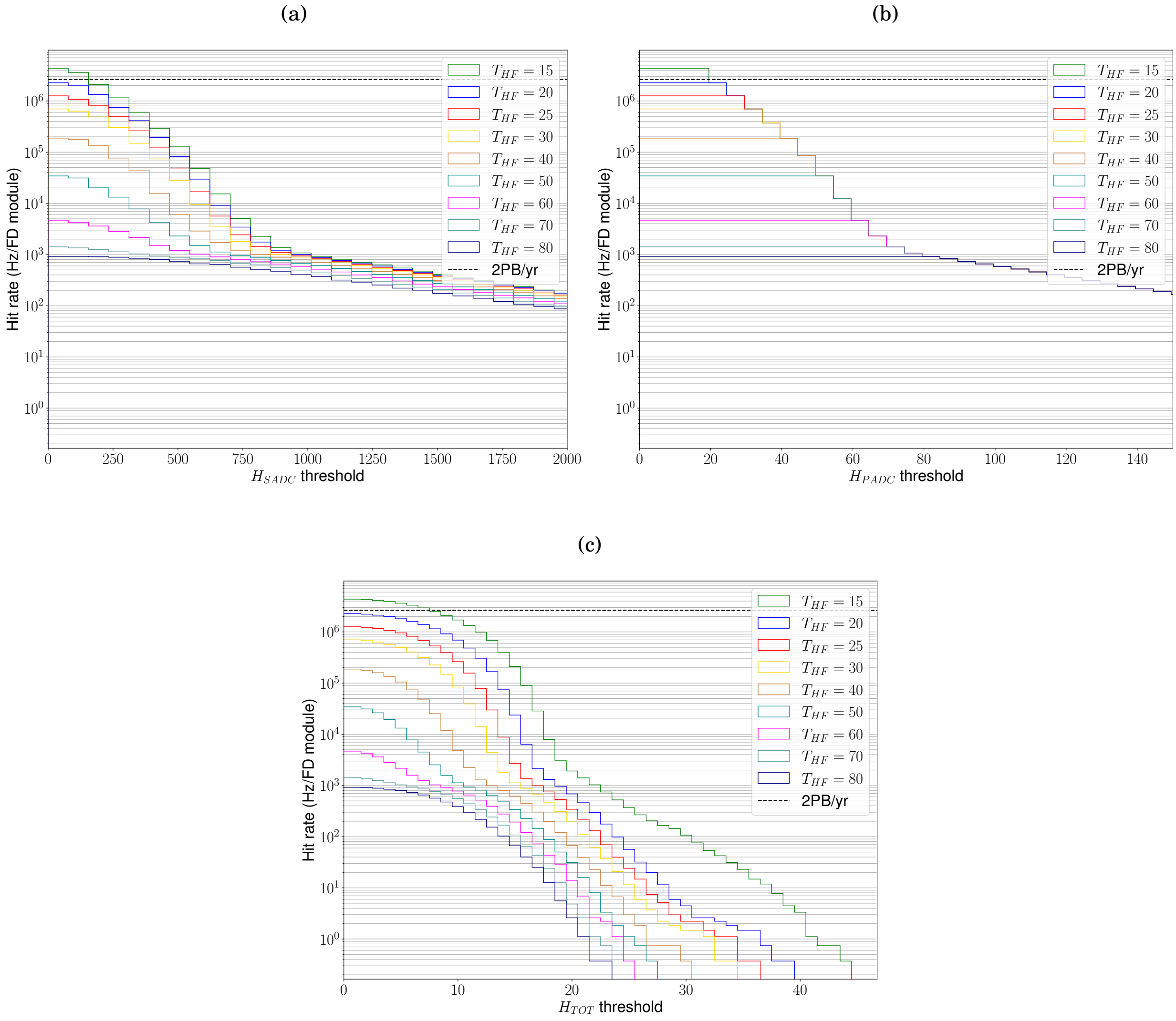
Several hit-finder thresholds were studied to determine a threshold which produced TPs at a low enough rate to satisfy the 2 PB per year requirement. Hit-finder thresholds from 15 to 80 counts were used. In these studies 10,000 events were used to determine the distributions of TP properties. This dataset is labelled ‘10k events containing SN neutrinos’.

In order to determine the rate of TPs from the detector noise and radiological backgrounds, the TPs which were backtracked to MARLEY neutrino interactions were filtered out. The rate of TPs from each of the T_{HF} values is displayed in figure 5.2, where the rate of TPs which survive the application of a threshold on the H_{SADC}, H_{PADC} and H_{TOT} are shown in sub-figures a, b and c respectively. Each TP is 24 bytes including metadata, so limiting the data volume to 2 PB per year leads to a required rate of 2.64×10^6Hz of TPs. A T_{HF} of 20 ADC counts can be seen to satisfy the 2PB per year requirement.

It was observed that the choice of T_{HF} applied in TPG affected the hit quantities. The most obvious effect can be observed in figure 5.2(b), where the minimum H_{PADC} is limited to be that of T_{HF} or above. It can also be observed from figure 5.2(c) that as T_{HF} is increased, the TPs have fewer ticks over threshold and therefore are only probing the peak of a region of signal which is shown to extend further when a lower T_{HF} is applied. This is supported by figure 5.2(a) where the H_{SADC} for lower T_{HF} values is consistently observed to be greater than that for higher T_{HF}.

These observations are not surprising but demonstrate clearly that increasing the threshold will result in the loss of some signals.

The difference between the TP rates for different T_{HF} values is within an order of magnitude above H_{SADC} = 1000 ADC counts. However, whilst the maximum rate of TPs for T_{HF} = 80 is below 1 kHz, for T_{HF} = 20 the maximum rate is just above 2 MHz. It is clear that the use of a high T_{HF} causes the TPG to discard the majority of TPs from background sources. These background signals can be used in detector calibrations [122], so it is advantageous to record as many as possible. The lowest T_{HF} which satisfied the 2PB per year requirement, 20 ADC counts, was chosen to produce the TPs used henceforth in this work.


(a)

(b)

(c)

FIGURE 5.2. Comparison of TP rates for different hit-finder thresholds, T_{HF}, used in TPG as a function of thresholds on the TP properties: (a) Hit summed ADC, H_{SADC}; (b) Hit peak ADC, H_{PADC} ; (c) Sum of ticks over threshold in hit, H_{TOT}.

Whilst the rate of TPs has been chosen to be as high as possible, it is useful to know the source of origin of these TPs. It was found that there were no TPs which were produced only as a result of detector noise as they could all be tracked back to originate from a radioactive decay source. From figure 5.3(a) it can be seen that the vast majority of TPs were a result of ^{39}Ar and ^{85}Kr decays. This is expected from the comparatively high decay rates of these isotopes in the detector simulation (see table 5.1) and the fact that they were present throughout the liquid argon in the detector volume.

However, compared to the other backgrounds, these sources produce a H_{SADC} distribution which does not extend beyond $H_{SADC} = 1500$. This may be explained by the maximum possible energies of the electron produced in their beta decays which are 566 keV and 687 keV respectively. These are low energies in comparison to the other beta decay sources tabulated in table 5.2. The only other radioactive decay which has no TPs with a H_{SADC} value above 1500 is ^{40}K. This undergoes beta decay producing electrons with energies up to 1.31 MeV and has a high decay rate but is present only in the CPA frame. It is possible that the lack of a strong ionisation signal is a result of recombination or diffusion effects which may be more pronounced for particles produced in these decays as the ionization has to drift an entire drift length to the collection wires before it is collected and digitised.

^{60}Co beta decays produce TPs with a larger maximum H_{SADC} value. These are the lowest energy beta decays in this set of radiological backgrounds but produce an excited nucleus, ^{60}Ni* which decays to two photons with energies of 1.17 MeV and 1.33 MeV. These ^{60}Co TPs with higher H_{SADC} values may result from resulting Compton scattering of electrons in the LAr.

Spontaneous fission of ^{238}U originating in concrete around the detector results in neutrons which may be captured by the LAr nuclei. The TPs which result have a distribution whose rate drops to zero at $H_{SADC} = 4000$. The only producers of TPs with greater H_{SADC} values are ^{42}Ar and the ^{222}Rn decay chain. Whilst ^{42}Ar produces electrons with an energy of 599.4 keV from its own beta decay, the resulting ^{42}K nucleus has a half-life of 12.36 hours and beta decays to produce an electron with an energy of 3.52 MeV. This may explain how the H_{SADC} distribution for TPs resulting from ^{42}Ar extends to higher values than other sub-MeV beta decays. The ^{222}Rn decay chain has two sources in the detector simulation: the LAr and the photon detector surface. No TPs were found which originated from ^{222}Rn in the photon detectors but TPs which originated from ^{222}Rn in the LAr were found to occur at a rate of 13 kHz per FD module.

The distribution of TP rates follows a very similar trend for the H_{PADC} value as can be seen in figure 5.3(b). This is expected since the H_{PADC} value is expected to be quite closely coupled to the H_{SADC} value. As illustrated in figure 5.3(c), there is not a clear relationship between the decay source and the TP TOT which may be due to the fact that the radioactive decay products are emitted in random directions whilst the collection wires are aligned with the y-axis of the detector.

The MARLEY-generated neutrino interactions produced primary electrons whose energy

Table 5.2: Maximum beta decay energies for decays present in the simulation for the "Model 1" background configuration

Decay Type	Maximum Decay Energy
^{39}Ar beta decay	565.5 keV
^{42}Ar beta decay	599.4 keV followed by beta decay of ^{42}K with 3.52 MeV
^{85}Kr beta decay	687 keV
^{60}Co beta decay	317 keV followed by $^{60}Ni^* \rightarrow 2\gamma$ with energies of 1.17 MeV and 1.33 MeV
^{40}K beta decay	1.31 MeV

spectrum peaks between 5 and 20 MeV as shown in figure 5.1. Only a small proportion of primary electrons have energies as low as the primary electrons produced from the radiological backgrounds. Therefore it makes sense that the H_{SADC} distribution of the TPs which originated from MARLEY neutrinos extends to much larger values than those of the background TPs, which is shown in figure 5.4(a). The full range of the MARLEY H_{SADC} distribution extends up to 30,000. However, it is observed that fewer than 5% of them have $H_{SADC} > 5000$, which is the value at which the rate of background TPs falls below 1 Hz. Once again the H_{PADC} distribution in figure 5.4(b) is very similar to the TP H_{SADC} distributions in figure 5.4(a). The TOT appears to discriminate more weakly between TPs from MARLEY neutrinos and backgrounds than the H_{SADC} or H_{PADC} as the difference in the initial gradients in figure 5.4(c) is shallower.

5.3 Characterisation of Trigger Clusters

In this section the TCs which resulted from using the optimal clustering parameters set out in table 3.2 during the application clustering on this dataset are investigated. Firstly clusters composed only of TPs from background sources are characterised, followed by the characterisation of the clusters which result when including the TPs from neutrino interactions.

5.3.1 Cluster Background

In order to determine the rate of clusters which resulted from background sources only, the TPs which were labelled as originating from neutrino interaction were filtered out before clustering. When the background TPs from this dataset were clustered, it was discovered that no TCs occurred which were solely a result of backgrounds.

Investigation revealed that using these tick and wire tolerances for clustering, clusters remained whose extent in wires on an APA, C_{rW}, was up to 4. However, the maximum number of TPs in a background cluster, C_n, was found to be only 4. As a result, to characterise the clustering of backgrounds, the clustering was re-run after changing to use $C_n \geq 2$, which is simply requiring that a cluster has at least 2 TPs.

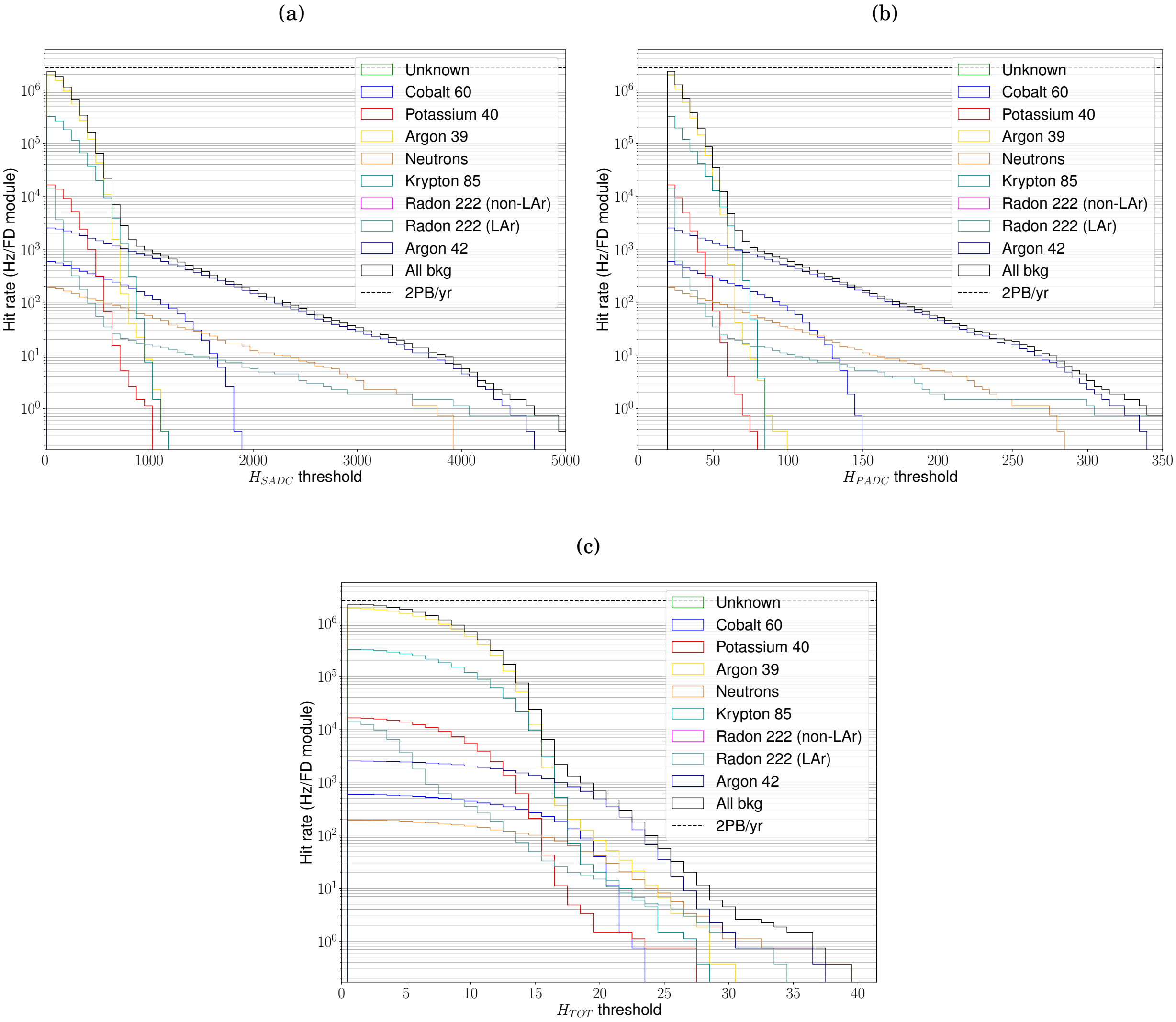


FIGURE 5.3. Comparison of TP rates for different detector background sources as a function of thresholds on the TP properties: (a) Hit summed ADC, H_{SADC}; (b) Hit peak ADC, H_{PADC} ; (c) Total ticks over threshold in hit, H_{TOT}.

The clusters were assigned to a particular background source based on which TPs in the cluster deposited the greatest proportion of the cluster summed ADC, C_{SADC}.

As can be seen by contrasting the distribution of C_{SADC} in figure 5.5(a) with the distribution of H_{SADC} in figure 5.3(a), it is apparent that the same background sources lead to the clusters and hits with the highest and lowest summed ADC values. This is unsurprising, as the concurrence of TPs in the detector has been found to be low. The similarity between the hits and clusters from the same radiological backgrounds extends to the comparison of their total time length. This is parameterised by the tick range of the clusters, C_{rT}, and the H_{TOT} of the hits. Both of these distributions show that TPs and TCs do not extend beyond 40 ticks in length. These can

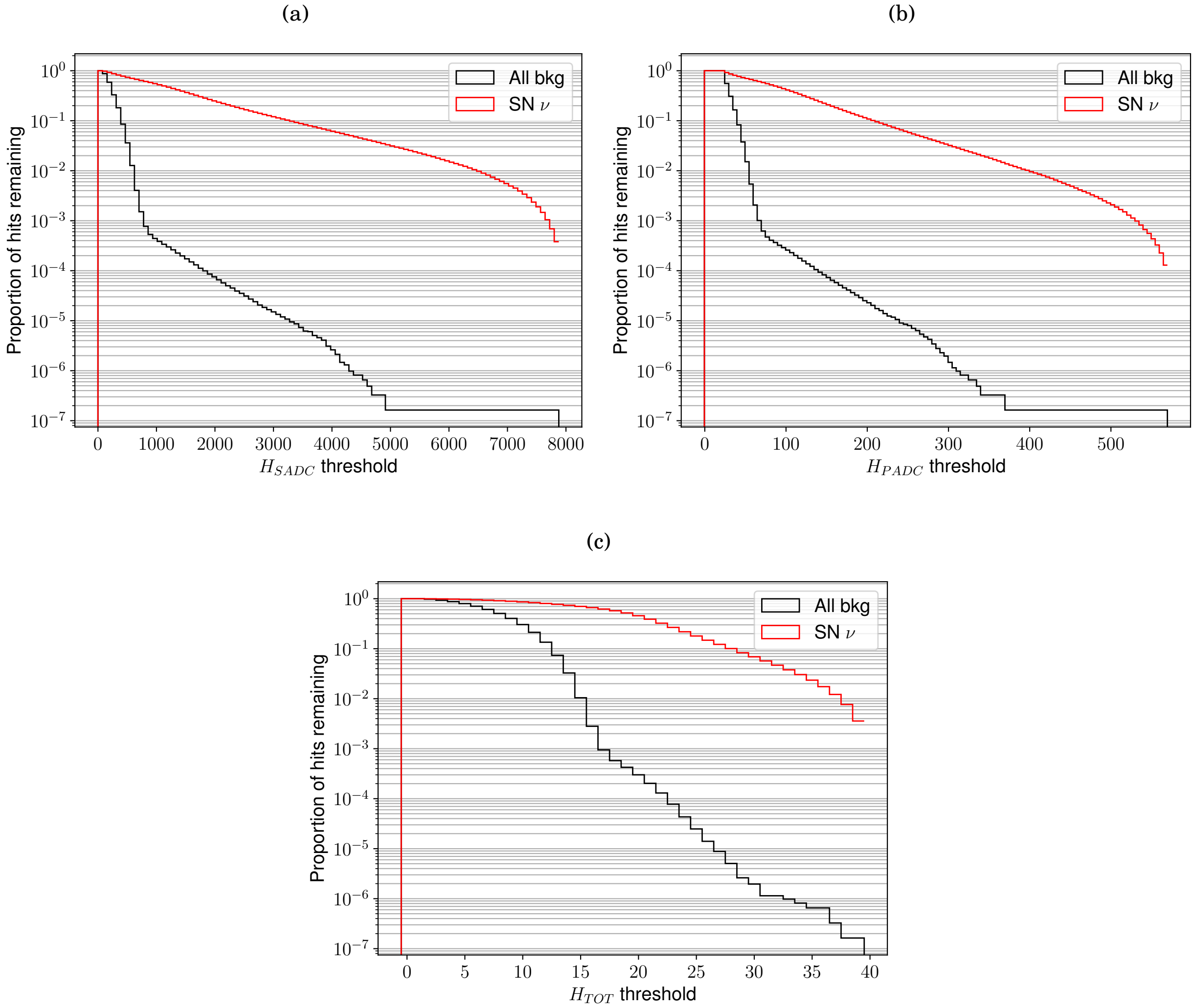


FIGURE 5.4. Comparison of the proportion of TPs from detector backgrounds and MARLEY neutrino interactions which remain after applying thresholds on: (a) Hit summed ADC, H_{SADC}; (b) Hit peak ADC, H_{PADC}; (c) Total ticks over threshold in hit, H_{TOT}.

both be contrasted with the distribution of the total number of ticks over threshold of the cluster, C_{TOT}, in figure 5.5(c), where it can be seen that ^{222}Rn, ^{42}Ar and neutrons are the only sources of backgrounds which result in clusters made up from TPs which have a non-negligible H_{TOT} in more than 1 wire.

5.3.2 Clustering With Neutrinos

The optimal clustering algorithm was then applied to the TPs whilst including the TPs from neutrino interactions. The dataset had 3 neutrinos simulated in the $1 \times 2 \times 6$ geometry per event,

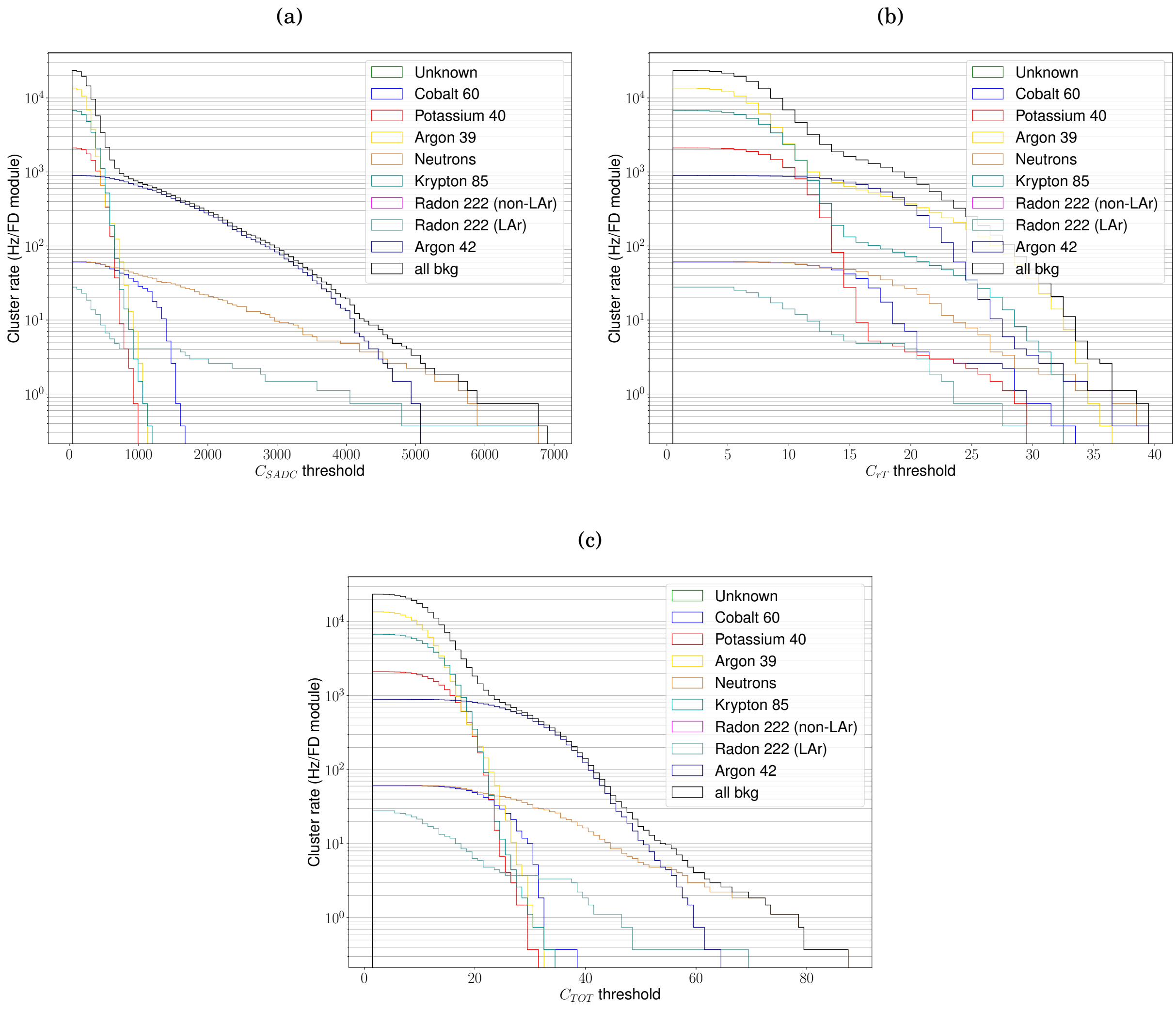


FIGURE 5.5. Comparison of TC rates for different thresholds on TC properties for each radiological background: (a) Cluster summed ADC C_{SADC}; (b) Tick width of cluster, C_rT; (c) Total ticks over threshold in the cluster, C_{TOT}.

so to avoid any pileup of neutrino signals, the TPs from only 1 neutrino at a time were included when applying the clustering. If a TC possessed 2 or more TPs which were labelled as being from the neutrino interaction, it was classified as a successfully detected neutrino signal (also referred to as a neutrino cluster).

The resulting properties of the neutrino clusters are plotted in figure 5.6, where it can be seen that the distributions of C_{SADC}, C_{rT}, C_{TOT}, C_c and C_{rW} all follow an exponential drop-off as the threshold on the cluster property is increased. The distribution of C_n, figure 5.6(d) shows that more than 50% of the detected clusters contain 8 or more TPs, though these do not all necessarily correspond to TPs from a neutrino interaction. The background clusters which were found using the relaxed clustering algorithm in figure 5.5(a) only extend to having C_{SADC} up to 7000. The neutrino clusters are observed in figure 5.6(a) to almost all have C_{SADC} values larger than this and extend up to 60,000. The maximum C_{rT} is also observed to be four times larger and both the C_{TOT} and C_{rW} are seen to be 6 times larger than for the background clusters. Taken together, these properties indicate that the neutrino interactions deposit a larger ionisation signal and do so over a greater region of the detector.

This shows that this clustering algorithm can indeed be used to detect neutrino signals with a low background. However, it was not the case that all neutrinos were successfully detected. Figure 5.7 shows the efficiency of finding at least one neutrino cluster as a function of the neutrino energy. The neutrino energy spectrum for the dataset is peaked between 5 MeV and 30 MeV and in this region it is observed that the clustering efficiency increases from 20% to 40% as the neutrino energy increases. It does not appear that the efficiency increases significantly past 40% though the statistics for these outlying regions are low so this cannot be said with certainty. The clustering parameters t_W and t_T seek to limit the range over which TPs are collected into a TC whilst C_n ensures that clusters with too few TPs are filtered out. As figure 5.6(d) shows that > 70% of TCs have $C_n \geq 6$, it is clear that the combination of the C_n with t_W and t_T are responsible for the reduction in clustering efficiency.

5.4 SNB Trigger

The SNB trigger is a simple algorithm which counts the TCs which occur within a time window in the detector module. To determine the performance, the rate of TCs from background sources in a given time window is determined and a threshold for the number of TCs which keeps the false SNB rate to once per month or below, T_{SNB} is computed. This threshold is used to determine the efficiency for the SNB trigger, ϵ_{SNB} as a function of the distance of a SNB.

As explained in the introduction to this chapter, previous studies done by collaborators first determined T_{SNB} by modelling the rate of background clusters as a Poisson distribution. They then combined the overall neutrino clustering efficiency with the expected number of neutrino interactions in the detector for a SN at a given distance. The resulting expected number of observed interactions in the detector was used as the mean of a Poisson distribution to determine ϵ_{SNB} using the T_{SNB}.

In the work presented in this section a different approach was taken. A Monte Carlo (MC) method was used determine ϵ_{SNB}. One identical assumption which was made was to assume the rate of TCs which result from backgrounds followed a Poisson distribution. As no TCs were found in this sample of 10000 events, an upper limit on the rate of TCs due to backgrounds was used as the mean. This was 0.367 Hz, equivalent to the precision of the dataset used. This provides an upper limit for the rate of clusters due to backgrounds and has been used to determine a lower limit on the rules-based SNB trigger efficiency. In addition, the average number of neutrino clusters (rather than efficiency) as a function of neutrino energy has been determined and is used

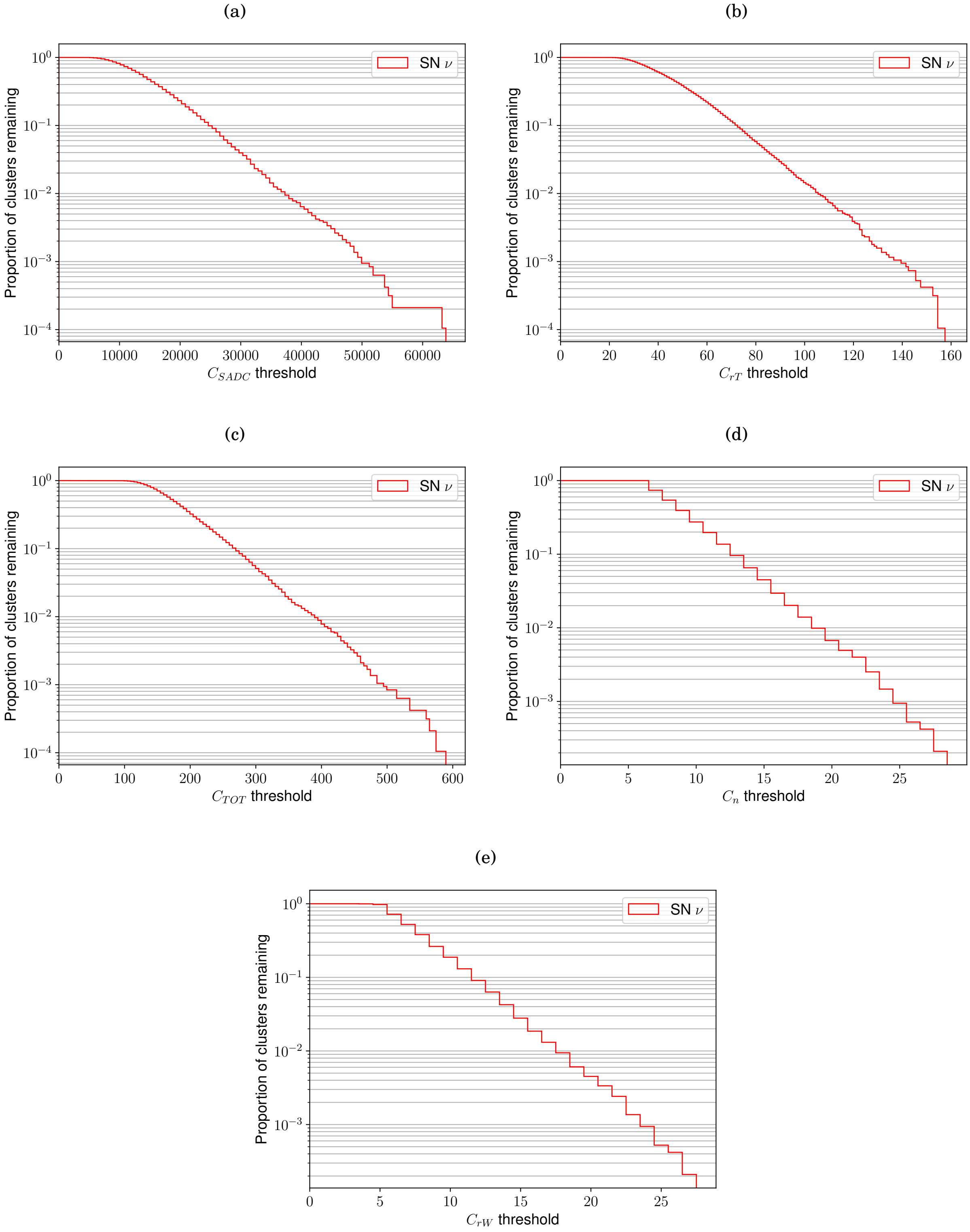


FIGURE 5.6. Proportion of TCs remaining for different thresholds on TC properties for TCs including ≥ 2 TPs from neutrino interactions: (a) Cluster summed adc, C_{SADC}; (b) Tick length of cluster, C_{rT}; (c) Total ticks over threshold of cluster, C_{TOT}; (d) Number of hits in a cluster, N_c; (e) Wire range of cluster, C_{rW}.

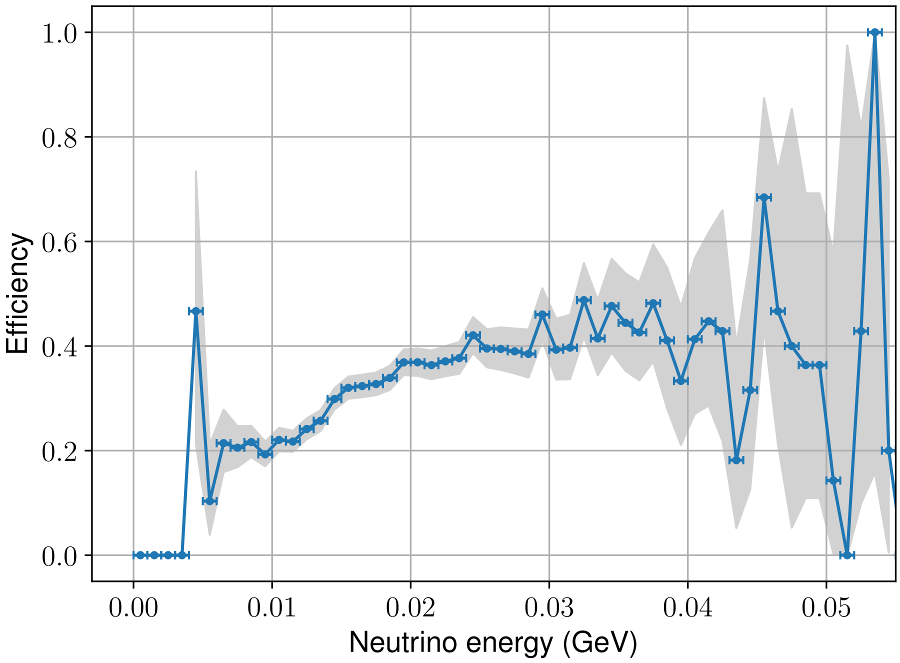


Figure 5.7: Efficiency of finding at least 1 TC with ≥ 2 neutrino hits by clustering the TPs from both backgrounds and neutrino interactions as a function of the neutrino energy. The horizontal error bars denote the width of the energy bins and the grey band denotes the a 95% confidence bound for clustering efficiency.

in combination with the SN energy-time spectrum reported in figure 2.7 to allow a more accurate determination of ϵ_{SNB}.

In this section the Monte Carlo method is described and the performance is reported.

5.4.1 Monte Carlo Model

Each trial simulated the number of TCs in a single FD module for 10 s - the maximum SNB trigger latency as a result of the 10 sec buffer. Trials were run separately for the background only and the combined background and SNB cases. The background-only trials were used to determine the false positive SNB trigger rate and the combined trials were used to determine ϵ_{SNB}. All SNB events started at $t = 0$ seconds in the trials and one set of trials was run for each SNB distance studied.

The simulation was split into 1 ms time bins. For each bin the number of background TCs was drawn from a Poisson distribution whose mean was the expected number of TCs from background sources - in this case 0.367 Hz. For trials which included a SNB event, the number of TCs in each signal additionally included a number of TCs from the SNB added according to the following procedure.

The 2D histogram, figure 2.7(a), containing the neutrino energy-time spectrum for a $11.2m_{\odot}$ SN at a distance of 10 kpc was used. The expected number of neutrino interactions in this distribution is given as a function of the time since the neutrino horizon reached the detector. These interactions could be simply scaled for different SNB distances using the inverse square law. The number of neutrino interactions was sampled for each time tick by projecting this histogram on to the time axis. This value was then used as the mean to sample a random number of neutrino interactions from a Poisson distribution.

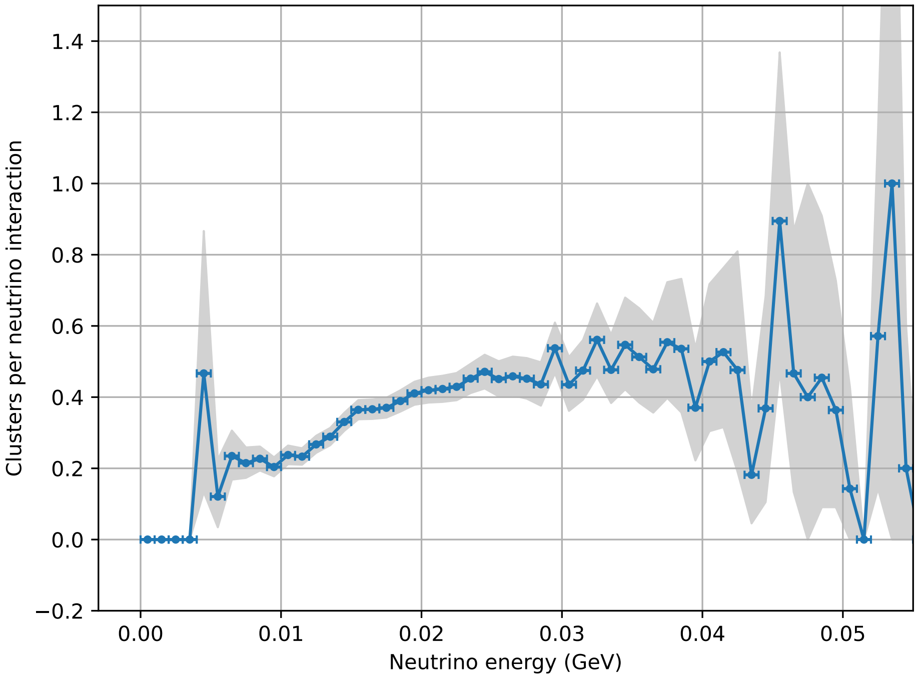


Figure 5.8: The average number of TCs as a function of the neutrino energy when using the baseline clustering algorithm with the optimal parameters. The horizontal error bars denote the width of the energy bins and the grey band denotes the a 95% confidence bound for clustering efficiency.

For each interaction, the neutrino energy was sampled by taking the distribution of energies for the given time bin and sampling from this histogram via the rejection method [123]. Since each neutrino interaction has been assigned an energy, the expected number of clusters was determined from figure 5.8, which plots the average number of TCs as a function of the neutrino energy for this clustering algorithm. The average number of TCs from the appropriate energy bin was then used as the mean of another Poisson distribution which was used to sample the number of TCs which resulted from this neutrino interaction.

After applying the same logic to determine the number of neutrino interactions in each bin in each trial, the total number of TCs in each trial was summed. Two months of background was simulated. Sets of 10,000 trials each were run to simulate SNB events at distances from 5-60 kpc.

5.4.2 Supernova Burst Trigger Performance

In order to determine ϵ_{SNB}, the T_{SNB} which was required to limit the false positive rate of SNB triggers to fewer than one per month was determined. The effect of the threshold on the number of TCs on the SNB false trigger rate can be seen in figure 5.9. A threshold of 15 TCs decreased the false SNB trigger rate to less than one per month.

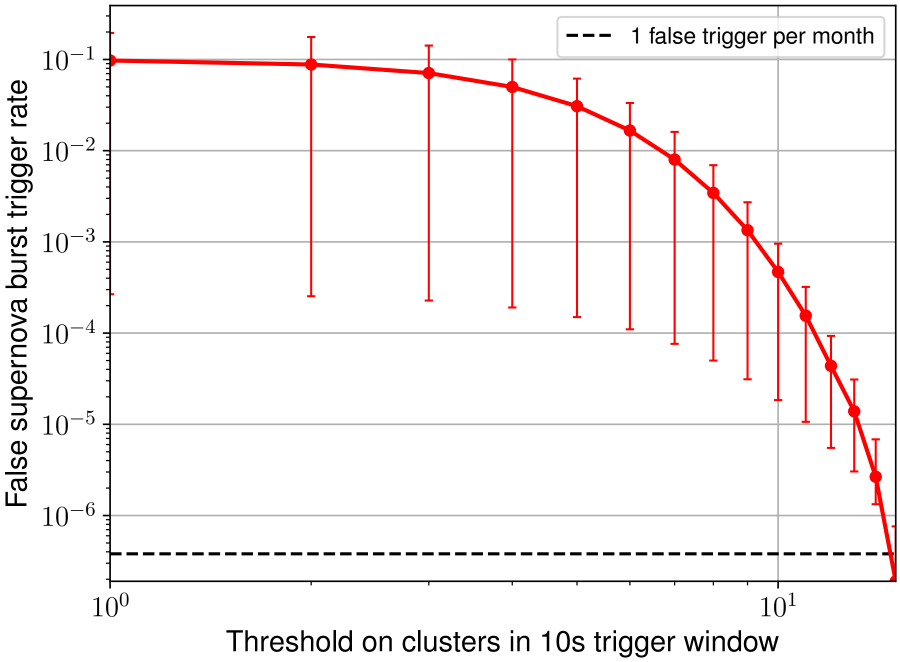


Figure 5.9: The false SNB trigger rate in Hz as a function of the threshold on the number of TCs collected within a 10 sec window. A threshold of 15 TCs was found to limit the false SNB trigger rate to less than one per month. Error bars show the 95% confidence region.

The efficiency resulting from this T_{SNB} as a function of the SNB distance is shown in figure 5.10. A SNB efficiency of 100% was achieved up to 15kpc which dropped to $97.7^{+0.2}_{-0.3}$% at 20kpc and $71.3^{+0.9}_{-0.9}$% at 25kpc. The efficiency at the LMC (50kpc) was found to be $1.4^{+0.1}_{-0.1}$%, lower than the 8% reported in the study on CPU-based TPG.

The DUNE SNB trigger is required to have an expected efficiency of > 95% for a SNB with at least 60 interactions, each of a minimum 10 MeV in true neutrino energy. As can be seen in figure 2.7(d), the distance at which 60 neutrino interactions are expected in a 10 kt detector module is 20kpc. Therefore, the rules-based SNB trigger satisfies the DUNE requirement for SNB sensitivity.

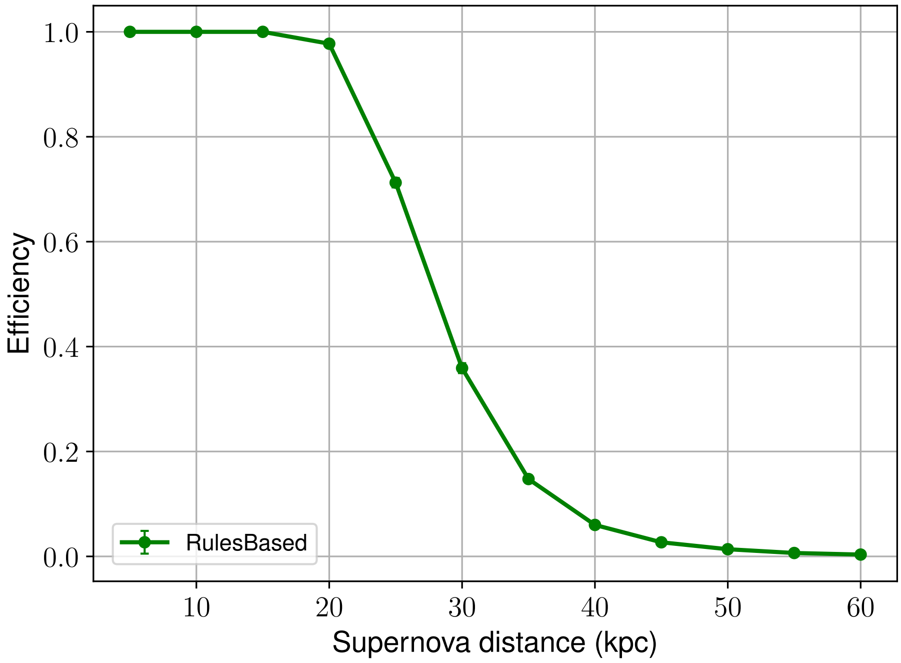


Figure 5.10: The efficiency of the SNB trigger using a threshold of 15 TCs within 10 sec to limit the false SNB trigger rate to less than one per month. Error bars were negligible.

5.5 Conclusions

In this chapter the baseline SNB trigger algorithm utilising FPGA-based processing has been simulated to determine its efficiency for a $11.2m_\odot$ progenitor SN. It was determined that a threshold of 20 ADC counts was required after filtering the signals in order to achieve a data rate of < 2PB per year of TPs. The properties of these TPs were characterised and they were clustered to form TCs. If a TC was formed from at least 2 TPs which had been back-tracked to a neutrino interaction, it was categorised as originating from a SN neutrino. The energy spectrum for the average number of TCs per neutrino interaction was determined along with an upper limit for the expected rate of TC due to backgrounds. These were used as inputs to determine the baseline SNB trigger efficiency using a toy MC method. The required threshold of TCs required to limit the SNB trigger to a false trigger rate of < 1 per month was determined to be 15. This resulted in an efficiency of $97.7^{+0.2}_{-0.3}\%$ at 20kpc. As can be observed from figure 2.7(d), this is the distance at which 60 neutrino interactions are expected in a FD module. Therefore, the baseline triggering algorithm fulfils the technical requirements for DUNE to be able to trigger on a SNB which produces at least 60 neutrino interactions in the detector with an efficiency of > 95%. However, for SNBs beyond 20kpc, the efficiency for detection rapidly falls of, dropping to 6% at 40kpc. If a SNB were to occur in the LMC, it would only have a 1.4% chance of being detected via this baseline SNB trigger. Whilst at this distance only 10 neutrino interactions are expected, similarly small numbers of neutrino interactions from SN1987A have led to hundreds of publications on constraints to neutrino properties and core collapse SN models. Being able to study the electron neutrino signal in DUNE would also complement the detection of electron antineutrinos in currently active water cerenkov detectors to allow additional insights into neutrino formation in SNe. Due to the low expected rate of occurrence of SNe in the galaxy, there is a strong motivation to increase the efficiency of the SNB trigger to be as high as possible, at least up to the LMC, which is the furthest source of SNe that would likely lead to several neutrino interactions in a FD module.

As stated in the introduction to this chapter, the clustering parameters used in this work were the optimal parameters found via a grid search of permutations in a previous study on similar but CPU-based TPs by DUNE collaborators. A more sophisticated Monte Carlo technique was also used to determine the SNB efficiency in the study which has been presented. The performance of FPGA-derived TPs was found to be only slightly poorer, with ϵ_{SNB} dropping from 100% to 97.7% at 20kpc and dropping from 8% to 1.4% at the LMC. This may be a result of their use of a lower peak-finding threshold in addition to the other differences. In the SNB triggering study presented in this chapter, the peak-finding threshold used was that which limited the total rate of TPs to less than 2 PB per year. If a lower threshold was found to increase the SNB trigger efficiency further, it would be possible to use it for SNB triggering and limit the rate of TPs saved to disk by filtering TPs based on their properties.

The optimal clustering algorithm was also determined from the exploration of quite a limited hypothesis space. Rules-based clustering depended on low energy neutrino signals leaving localised charge deposits which could be differentiated from backgrounds using only information about their spatial extent and constituent TP. It may be advantageous to explore a less limited hypothesis space and, rather than use a grid search to determine the optimal parameters, to converge upon them iteratively.

Chapter 6

Machine Learning

An algorithm can be used to parameterise a process and express a mathematical model of a system. Models are often used to perform classification of data or use it to make predictions through regression. All data can be considered to have an underlying distribution or function which generates it. Sampling this distribution generates a dataset. The optimal values of the parameters in a model can be iteratively converged towards by providing data to a ML algorithm, which can be said to have learned from some data if its performance at a task improves when measured by a given performance measure [124]. However, the no free-lunch theorem [125] states that if all data-generating distributions are averaged over, then every classification algorithm has the same error rate when classifying previously unobserved data points. This means that no ML algorithm will universally perform better than any other algorithm when applied to model all functions. Therefore, when applying machine-learning to a given real problem, assumptions have to be made about the kind of functions which can be used to model it. By making these assumptions ML algorithms can be designed which can allow a model to improve its performance at a given task on data sampled from a given distribution.

A parametric function, g, can be used to make predictions of some distribution or vector of parameters (vectors are represented with bold typeface), $\boldsymbol{\theta}$, from a set of m independent and identically distributed data points, $\boldsymbol{x}^{(1)},...,\boldsymbol{x}^{(m)}$, each consisting of a vector describing the observation such as physical quantities or pixels in an image. Point estimators, $\hat{\boldsymbol{\theta}}$, are an attempt to make a prediction of the $\boldsymbol{\theta}$ parameters only from the set of m observed data points [126]:

$$\hat{\boldsymbol{\theta}} = g(\boldsymbol{x}^{(1)},...,\boldsymbol{x}^{(m)}). \tag{6.1}$$

The hat represents the fact that the point estimator is determined from observed data. Since the data is drawn randomly from the unknown data-generating distribution, $p_{data}(\boldsymbol{x})$, $\hat{\boldsymbol{\theta}}$ is a random variable.

In order to determine the best $\hat{\boldsymbol{\theta}}$ for a given model given some observed data points, maximum likelihood estimation (MLE) can be used. The maximum likelihood point estimator, $\hat{\boldsymbol{\theta}}_{\boldsymbol{ML}}$, for the parameter space indexed by $\boldsymbol{\theta}$ when using a model composed of a set of parametric probability distributions, $p_{model}(\boldsymbol{x};\boldsymbol{\theta})$ is

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{ML}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \prod_{i=1}^{m} p_{model}(\boldsymbol{x}^{(\boldsymbol{i})};\boldsymbol{\theta}). \tag{6.2}$$

When the likelihood function is maximised, the observed data is most probable as a result of the statistical model.

As the MLE is difficult to compute, the logarithm can be taken to instead obtain $\hat{\boldsymbol{\theta}}_{\boldsymbol{ML}}$ as a sum. After taking the logarithm and dividing through by m, $\hat{\boldsymbol{\theta}}_{\boldsymbol{ML}}$ can instead be expressed as an expectation value, $\mathbb{E}$, with respect to the distribution defined by the observed data points, $\hat{p}_{data}$:

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{ML}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \mathbb{E}_{\boldsymbol{x}\sim\hat{p}_{data}} \log(p_{model}(\boldsymbol{x};\boldsymbol{\theta})). \tag{6.3}$$

Therefore the MLE is actually a minimisation of the dissimilarity of the empirical distribution $\hat{p}_{data}$ - which is represented by the observed data points - and the distribution represented by the model $p_{model}(\boldsymbol{x};\boldsymbol{\theta})$. By using a measure of the difference between probability distributions, the Kullback-Leibler (KL) divergence [127] D_{KL}:

$$D_{KL} = (\hat{p}_{data}||p_{model}) = \mathbb{E}_{\boldsymbol{x}\sim\hat{p}_{data}}[\log(\hat{p}_{data}(\boldsymbol{x})) - \log(p_{model}(\boldsymbol{x})], \tag{6.4}$$

it can be seen that to minimise D_{KL}, only $-\mathbb{E}_{\boldsymbol{x}\sim\hat{p}_{data}} \log(p_{model}(\boldsymbol{x})$ is a function of the model. Hence when training a ML model on some observed data, only this negative log-likelihood needs to be minimised. The MLE is particularly useful in ML because as $m \to \infty$, the MLE converges to the actual $\boldsymbol{\theta}$ value. When using a parametric model, a way to measure how close the value of a parameter is to the true parameter is the mean squared error (MSE). The MSE decreases as m increases and when m becomes large, the Cramer-Rao lower bound shows that no estimator has a lower MSE than the MLE [128] [129].

The concept of a MLE has been introduced. It has been shown that a MLE can be used in conjunction with observed data to identify the optimum parameters for a given model and can increasingly closely approximate the truth parameters as the sampled data set increases in size. Deep learning algorithms are composed of a model, a dataset, an optimisation procedure and a cost function which is minimised through the negative log-likelihood described above, leading to the maximum likelihood estimation.

In this chapter the concepts and theory of artificial neural networks (ANNs) and convolutional neural networks (CNNs) will be explained, including the optimisation procedure and its relation to the cost function. Finally the YOLOv3 network architecture and cost function will be introduced along with the main metrics which can be used to evaluate its performance.

6.1 CNN Theory

CNNs are a class of ANN which make use of additional regularisation and are often applied to data with a grid-like topology. In this section, the mathematics of ANNs and the training of their parameters is introduced. These concepts are general to CNNs, which are then introduced along with common techniques employed to improve their performance.

6.1.1 Artifical Neural Networks

ANNs can be used to approximate a complex function through the combination of simpler ones. A feedforward ANN is described by a directed acyclic graph (DAG), made up of 'nodes' connected by 'edges'. An example of an ANN is shown in figure 6.1, with a direction of propagation of the edges from left to right. As the DAG is directional, any path through it is finite and will always go from the input nodes to the output nodes. The input nodes are those which do not have any input edges and are provided with values from the observed data set. Any computation only depends on the previous nodes in the network. The computation at each node of the network, f_i, is parametrised by a vector of inputs, $\boldsymbol{x}$ and the weights in the network, $\boldsymbol{w}$:

$$f_i(\boldsymbol{x},\boldsymbol{w}) = g\big(\sum_j x_j w_{ij} A_{ij} - b_i\big) \tag{6.5}$$

where j is the index of the input edges to node i, x_j is the input data carried by each edge, w_{ij} are the weights applied to the input data from the j'th edge at this node, A_{ij} is the adjacency matrix which describes the connections between nodes in the DAG and b_i is a bias term for this node. $x_j w_{ij} A_{ij} - b_i$ is a linear function which describes an affine transformation from an input vector to an output scalar. To allow the network to model a nonlinear function, a non-linear activation function g is applied.

The nodes in an ANN are often arranged into layers where the nodes of each layer are connected to all the nodes from the previous and next layers, forming a fully connected network. The layers between the input and output layers are referred to as hidden layers. In most cases, the network architecture is kept constant and the weights and biases are optimised iteratively. From hereon the biases will be implicitly included whenever the weights are discussed. As a whole, the network accepts an input vector $\boldsymbol{x}$ and produces an output vector $\hat{\boldsymbol{y}}$. Through forward propagation its action on $\boldsymbol{x}$ as a whole can be denoted as $f(\boldsymbol{x},\boldsymbol{w})$, where $\boldsymbol{w}$ are the weights in the network.

For an input dataset, $\boldsymbol{x}$, the goal is to determine a set of weights, which map the input to a set of targets, $\boldsymbol{t}$, which represent the desired output of the network via the application of $f(\boldsymbol{x},\boldsymbol{w})$. For a given input-target pair, a 'loss function', L which evaluates the dissimilarity of $\hat{\boldsymbol{y}}$ and $\boldsymbol{t}$ can be computed. The weights can be optimised to minimise L through gradient descent, as explained in the following section. The goal is to determine an optimal set of weights, $\boldsymbol{w_{ML}}$ which optimises the loss function and can be parametrised as:

$$\boldsymbol{w_{ML}} = \underset{\boldsymbol{w}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} L(f((x_i, \boldsymbol{w}), t_i)), \tag{6.6}$$

where there are N input-target pairs.

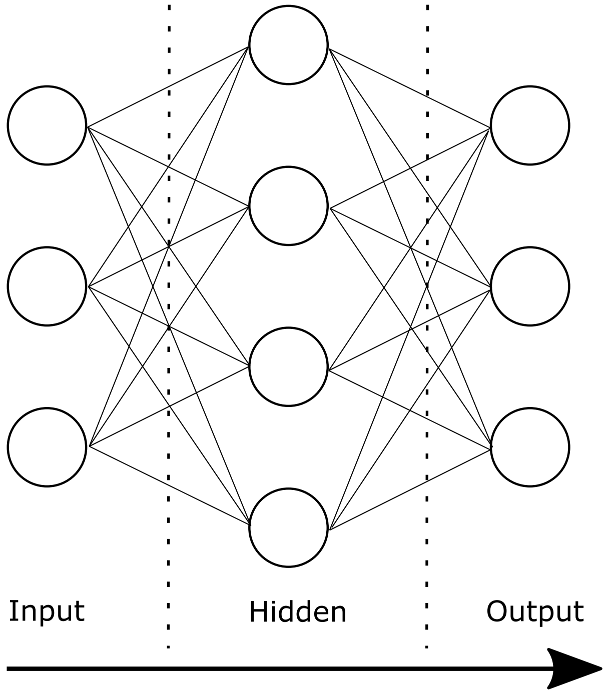


Figure 6.1: A fully connected artificial neural network with an input layer composed of 3 nodes, a hidden layer with 4 nodes and an output layer with 3 nodes. Each node has a bias value. The layers are fully connected by edges, each with a weight parameter. The computations are directed from left to right.

6.1.2 Loss Functions and Backpropagation

The loss function is a measure of the difference between $\boldsymbol{\hat{y}}$ and $\boldsymbol{t}$. When a network is used as the model for a regression task, a common metric used to compute the loss function is the mean squared error (MSE). The hidden layers of the network can be thought of as building up a transformed representation of $\boldsymbol{x}$ and the last layer can be thought of as a linear regression under the assumption that $\boldsymbol{t}$ follows a linear model in the transformed space representation. Assuming that the errors are randomly distributed according to a Gaussian distribution, the probability of observing the targets given the input data and network weights, $p(\boldsymbol{t}|\boldsymbol{x},\boldsymbol{w})$, is given by

$$p(\boldsymbol{t}|\boldsymbol{x},\boldsymbol{w}) = \prod_{i}^{N} \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{(f(x_i,\boldsymbol{w})-t_i)^2}{2\sigma}}. \tag{6.7}$$

Taking the logarithm this becomes

$$\ln\big(p(\boldsymbol{t}|\boldsymbol{x},\boldsymbol{w})\big) = \frac{N}{2}\ln(\sigma) + \frac{N}{2}\ln(2\pi) + \frac{1}{2\sigma}\sum_{i}^{N}(f(x_i,\boldsymbol{w}) - t_i)^2 \tag{6.8}$$

from which it can be seen that optimising a MSE loss is the MLE for a regression task with an assumed Gaussian error distribution and a fixed data set size as $p(\boldsymbol{t}|\boldsymbol{x},\boldsymbol{w})$ is then dependent only on the MSE term.

Unlike linear regression, neural networks with non-linear activation functions are not convex functions and therefore have the possibility of getting stuck in local minima rather than always finding a global minimum (in reality they will iterate towards a minimum asymptotically rather than find it exactly). Despite this they are useful for approximating complex functions and their behaviour during training can be controlled via optimization procedures.

As long as the feedforward network is composed of only differentiable functions and the network architecture is acyclic, backpropagation [130] can be used to train the network. Backpropagation makes use of the loss function to update the network weights in each layer. The loss function can be expressed as

$$L(\boldsymbol{x},\boldsymbol{t}) = \sum_{i}^{N} l(f(x_i,\boldsymbol{w}),t_i) \tag{6.9}$$

where l is the loss for the i'th input-target pair and $f(x_i,\boldsymbol{w})$ represents the action of the network on the inputs with respect to $\boldsymbol{w}$. In order to find the optimal weights $\nabla_{\boldsymbol{w}}L(\boldsymbol{x},\boldsymbol{t})$, the gradient of L, can be taken to find the direction in which L is fastest minimised with respect to $\boldsymbol{w}$:

$$\nabla_{\boldsymbol{w}}L(\boldsymbol{x},\boldsymbol{t}) = \sum_{i}^{N} \nabla_{\boldsymbol{w}} l(f(x_i,\boldsymbol{w}),t_i) \tag{6.10}$$

where

$$\nabla_{\boldsymbol{w}} = \Big(\frac{\partial f}{\partial \boldsymbol{w}}\Big)^T \Big(\frac{\partial}{\partial f}\Big). \tag{6.11}$$

This can be used to determine an updated value for the weights $\boldsymbol{w}'$, where the scale of the update is controlled via a positive scalar, ϵ, known as the learning rate

$$\boldsymbol{w}' = \boldsymbol{w} - \epsilon\frac{1}{N}\nabla_{\boldsymbol{w}}L(\boldsymbol{x},\boldsymbol{t}). \tag{6.12}$$

As seen from equation 6.11, updating the weights in each node requires the computation of the Jacobian of the function which represents the neural network. A gradient can be computed for the output nodes of the network from the loss function. As the network is composed of simple and differentiable functions and the connections between nodes represent a DAG where each node depends only on the nodes in the previous layer, the chain rule can be applied recursively to determine the partial derivative of the loss with respect to each of the weights in the network. With this information the weights can be updated according to equation 6.12.

6.1.3 Gradient Descent and Optimisation

In practice it is often unfeasible to use the ANN to process all N input-target pairs in the training data set before updating the network weights. Optimisation algorithms which use the entire training set to update weights are known as batch gradient methods, whilst those which only use a single sample from the training set are known as stochastic gradient methods. As a result of using only a subset of the training data, stochastic gradient methods have some inherent noise when they are used in the optimisation of the weights but as long as the samples are independent and identically distributed, stochastic methods will not bias the estimate of the gradient.

Rather than using batch or stochastic gradient descent it is usual in practice to use minibatch stochastic gradient descent, where the weights are incremented after processing a subset of several examples from the training data set, making up a minibatch. Minibatch stochastic gradient descent (SGD) is set out in full in appendix A.3 and is the application of 6.12 to a minibatch of data. Its use can result in the speedup of training even though larger minibatch sizes give a more accurate estimate of the gradient.

When there is a shallow gradient, SGD converges slowly towards a minimum. Recent implementations of gradient descent employ additional techniques to try to ensure a minimum is reached efficiently. A widely used SGD algorithm known as Adam [131] incorporates momentum by including a velocity term, v, which helps to accelerate the minimisation by keeping track of an exponentially decaying average of the negative gradient over the previous iterations. This is particularly helpful in situations where the stochastic gradient has high variance (which can be a result of small minibatch sizes) and where the loss has a shallow gradient. The other main conceptual differences from SGD are that Adam uses an adaptive learning rate for each parameter instead of a fixed one and gradient rescaling is implemented. The details of the Adam optimiser can be found in appendix A.4.

6.1.4 Activation Functions

All functions of which the network is composed must be differentiable. As a nonlinear activation function is required to allow the network to approximate nonlinear functions, these activation functions must also be differentiable. Some activation functions are commonly applied in the hidden layers of the network, some only ever applied to the output layers and some are applicable to both cases.

6.1.4.1 Hidden Layers

A recommended activation function to use in the hidden layers of an ANNs is the rectified linear unit (ReLU) [132]:

$$g_{relu}(z) = \max(0, z), \tag{6.13}$$

which is linear if $z > 0$ but zero if $z \leq 0$. This has the effect of ensuring that any computed gradient is large compared to the value which would be computed using other activation functions, such as the sigmoid, $g_{sig}(z) = \frac{1}{1+e^{-z}}$ or the tanh function $g_{tanh}(z) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$. ReLU has been shown to achieve a faster convergence during training in many cases than tanh or sigmoid activation functions [133], which may be a result of vanishing gradients occurring in tanh and sigmoid functions as the weights are trained.

The initialisation of weights is an important consideration when training a network. The interactions between the techniques used to initialise weights and the activation functions as the network trains are important to consider. In the situation that the weights in a node are initialised to a negative value, the output from the applied ReLU on that node will be zero, meaning that it does not contribute to the gradient. Additionally, no matter what activation function is used, if all weights are initialised to the same value in a given layer of the network, then all the nodes will be identical and, as they are fully connected in an ANN, will therefore be learning exactly the same thing. This behaviour creates a network which is equivalent to a single node at each layer which had its weights uniformly initialised. A conventional way to initialise weights in nodes which use a ReLU activation function was set out by He et al [134] where for a network layer with N nodes, the weights are initialised from a Gaussian probability distribution with a mean of 0 and a standard deviation of $\sqrt{\frac{2}{N}}$.

A variation of the ReLU is the leaky ReLU, g_{leak}, which is defined as [135]:

$$g_{leak}(z) = \begin{cases} z, & z > 0 \\ 0.01z, & else. \end{cases} \tag{6.14}$$

This increases the proportion of non-zero weights but allows a small, non-zero gradient when the activation returns a negative value, allowing the weights to continue being optimized for this node.

6.1.4.2 Output Layers

When applied to a regression problem, the output of a network will often simply use a linear activation function. If instead a binary classification score is required, in most cases the sigmoid activation function is used.

If there are K classes in the dataset and a vector holding the probabilities that the prediction is a member of each class, $\boldsymbol{c}$, is required, the softmax function is used:

$$\sigma_{softmax}(\boldsymbol{c}) = \frac{e^{c_i}}{\sum_{j=1}^{K} e^{c_j}} \tag{6.15}$$

6.1.5 Convolutional Neural Networks

ANNs are not well suited to use in deep learning on image data. This is because for any image with a reasonable number of pixels, a fully connected network would have to learn many millions of weights. This is impractical when training the weights and requires long computation times to classify images from a trained model due to the large number of computations between layers. Additionally, if an image is at all translated or rotated, this would cause completely different activations in each layer of the network which makes generalisation increasingly problematic as the number of network parameters increases.

Instead CNNs are commonly used to process data which has a known grid-like topology. They learn the weights of 'convolutional kernels' rather than for connections between pixels. The cross correlation, which is usually used in ML software libraries to perform convolutions, is described by:

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n) \tag{6.16}$$

where i,j are indices of the pixels in an input multidimensional array I, and m,n are the indices of non-zero elements in the kernel K. The output $S(i,j)$ is known as a feature map. The multidimensional array may be an image with one or more colour channels or could be a vector of feature maps. The kernel array is usually composed of weights which are zero except in a small local region, as shown in figure 6.2. If the data is arranged in a 2D grid-like topology, such as an image, then the kernel will possess learnable weights over two dimensions but will be applied to all channels in the image.

Usually the kernel is much smaller than the array it is convolved with. This sparse connectivity between the inputs and the weights allows a decrease in the required number of computing operations in addition to decreasing the required memory and learnable parameters. These sparse kernels are still found to be useful in detecting features such as edges which can be useful in many image processing tasks.

Repeated applications of sparse kernels in layers of deep CNNs allow more distant regions of an input image to contribute to the output of a deeper convolution as shown in the right hand side of figure 6.2. CNNs are implemented with kernels in each layer to allow multiple features to be learned.

CNNs also use a technique known as parameter sharing. In ANNs each weight is only applied to each pixel once. However, in CNNs each kernel is applied to every pixel in the image (barring edge effects) which allows a single set of weights to be learned for each kernel. This parameter sharing means CNNs have equivariance to translation. Convolutions create a map of where a given feature (represented by the weights in the kernel) is found in an image. When that feature is translated in an image, the representation of that feature will be translated by the same amount.

After applying a convolution and activation function, CNNs usually make use of a 'pooling function'. This replaces the output of the network by applying a function which summarises a local region of the network. A popular function is max pooling [136] which returns only the maximum value in a rectangular region of pixels, effectively selecting the most important local

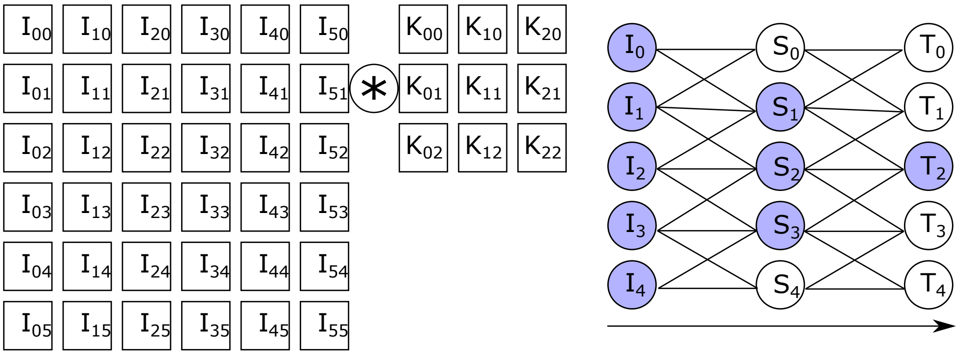


Figure 6.2: In a convolutional neural network a convolutional kernel, K with sparse connectivity is convolved with a multidimensional array, such as the original image, I, or vector of feature maps. The receptive field of repeated width 3 convolutions on a one-dimensional feature map is demonstrated on the right, with the features in node 2 of layer T being receptive to features in the nodes highlighted in blue. The weight parameters, K_{mn} in the applied kernel(s) are learned.

feature according to the kernel parameters. Pooling causes the network to be invariant to small translations of an input. In deep networks consisting of repeated applications of convolution, activation and pooling layers, the network can learn which transformations to become invariant to, such as rotations or reflections. Often pooling is applied with downsampling as the output is similar over the pooled region.

6.1.6 Practical Application of Convolution Kernels

The application of a convolutional layer may change the dimensions of the network. A convolution applied to a feature representation of width n pixels will produce an image with the dimensions n' along that axis:

$$n' = \frac{n + 2p - f}{s} + 1 \tag{6.17}$$

where p is the number of pixels added along the axis as padding, f is the size of the convolutional kernel along the axis and s is the ‘stride’ along the axis. Padding is where additional pixels are added. These are usually initialised either to 0, from the neighbouring pixels, or from the pixels on the opposite site of the image (wrap-around). Padding is often used to preserve the dimensions of the network when convolutional kernels with $f > 1$ are used. The stride denotes the distance in pixels between subsequent applications of the convolutional kernel. A stride of 1 means that the kernel will be applied to each subsequent pixel along this axis whilst a stride of 2 means that the kernel will only be applied to half of the pixels along the axis.

A convolutional layer may preserve the output dimensions of the previous layer, or it may be a downsampling or upsampling layer. Downsampling layers produce feature maps whose dimensions are smaller than the input to the layer. This may be achieved through pooling (usually taking the maximum value) of a group of consecutive pixels. Alternatively a convolutional layer itself can perform downsampling through the use of a strided convolutional kernel. Upsampling layers output feature maps whose dimensions are larger than the input to the layer. This is often achieved through the introduction of an extra row of pixels between each existing row. The pixel values are initialised from those of the existing pixels and this 'upsampling' of the feature map is followed by the application of a convolutional kernel.

6.1.7 Batch Normalisation

'Batch normalisation' is a network optimisation strategy. Through backpropagation a neural network determines how much to update the weights in a given layer based on the gradient propagated from the subsequent layer. This assumes that all weights in previous layers are fixed. Therefore the updated weights are not completely optimal. This effect is called 'internal covariate shift' and can be reduced using batch normalisation [137].

Given that the weights of all layers are updated at the same time, exponentially large or small gradients can result if the weights in subsequent layers are much larger or smaller than 1, causing the network to fail to converge. This effect is more prevalent for very deep networks. The optimisation landscape becomes smoother from the application of batch normalisation and it is this effect rather than internal covariate drift which Santukar et al [138] argue is the main reason for the improved performance.

Batch normalisation is performed on the values output by a minibatch after the application of a convolution. It can be applied either before or after the activation function, depending on in which case the values are more likely to be Gaussian distributed. Let $\boldsymbol{H}$ be these values arranged as a matrix with each image in the minibatch represented by a row in the matrix. To normalise $\boldsymbol{H}$ to $\boldsymbol{H}'$ the minibatch is normalised via [126]:

$$\boldsymbol{H}' = \frac{\boldsymbol{H} - \boldsymbol{\mu}}{\boldsymbol{\sigma}} \tag{6.18}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are implemented element-wise to each row, i of $\boldsymbol{H}$ and are determined from

$$\boldsymbol{\mu} = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{H}_i \tag{6.19}$$

and

$$\boldsymbol{\sigma} = \sqrt{\delta + \frac{1}{m}\sum_{i=1}^{m} (\boldsymbol{H} - \boldsymbol{\mu})_i^2}\,. \tag{6.20}$$

δ is a small number to prevent undefined divide operations and m is the number of images in the minibatch. These operations can be backpropagated through and ensure that updating the weights never increases the standard deviation or mean of the parameters in a single row of $\boldsymbol{H}$ only. The increased stability of training allows the learning rate used to be larger and therefore increases the rate of convergence during training. ReLU activation functions usually have batch normalisation performed on their inputs rather than their outputs whilst less linear activation functions such as tanh or sigmoid tend to have batch normalisation performed on the outputs of the activation function.

6.1.8 Skip Connections

Skip connections either add or concatenate the feature maps learned by one layer with those learned by a later layer. They are particularly useful in deep neural networks which are prone to vanishing gradients in their early layers, which can slow convergence of the model. Skip connections can be implemented over a short or long range.

Short range skip connections were introduced in the ResNet architecture [139]. These short range skip connections form ‘residual blocks’ which are composed of two consecutive convolutional layers. The inputs to the first layer are added to the outputs of the second layer, as shown in the residual block diagram in figure 6.3. The dimensions of the layers must be preserved to allow this. The residual blocks are shown to improve model convergence. They preserve information which has been extracted by earlier layers of the network to later layers, helping to limit the abstraction of the features deeper in the network but preserving the width probed by repeated convolutions.

Long range skip connections are most well known for their use in DenseNet [140] and U-Net [141]. DenseNet is able to improve upon the performance of ResNet with the use of less than 10% of the learnable parameters. The U-Net is an encoder-decoder architecture used for pixel-wise labelling (known as semantic segmentation) and shows that long range skip connections are effective for propagating localisation information so that it can be re-used in later layers. This allows very accurate localisation predictions from the output layers. Long range skip connections are usually implemented as concatenation of feature maps from an earlier layer to a later layer. This requires that the dimensions of these layers are the same, with the exception of the number of feature maps.

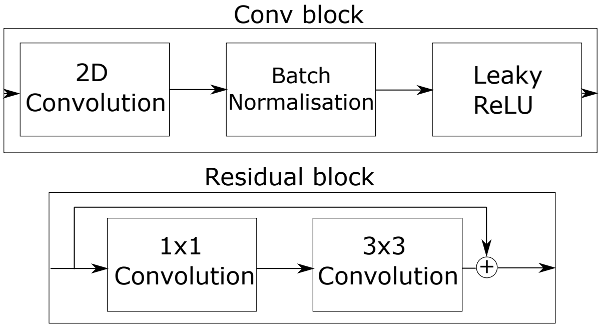


Figure 6.3: Composition of a convolutional block and a residual block. Convolutional blocks are composed of a 2D convolution, followed by batch normalisation and application of a subsequent leaky ReLU activation function. Residual blocks are composed of 2 convolutional blocks with a short range skip connection which adds feature maps together.

6.2 YOLOv3 Network

YOLOv3 [142] is a multi-reference, multi-resolution object detection network which makes object predictions in images. It makes predictions utilising multiple reference boxes (also referred to as anchor boxes) which guide the size and aspect ratio of the predictions. These predictions are also made at 3 different scales, allowing the network to be more sensitive to small objects than previous implementations of YOLO networks.

This section describes the architecture of the YOLOv3 network, including the inputs and outputs. The contributions to the loss function are then explained. The post-processing which is applied to the predictions is then set out, followed by a discussion of performance metrics.

6.2.1 YOLOv3 Network Architecture

The architecture of YOLOv3 is shown in figure 6.4. The input image is first passed to a Darknet-53 feature extraction network, highlighted in yellow, which is designed to output high quality feature maps. The path of the network follows a series of convolutions, denoted by solid dots, which downsample the inputs and produce feature maps. After several stages of convolution the feature maps are upsampled to double their dimensions using upsampling layers, coloured in green. Long range skip connections propagate localisation information to later layers of the network by concatenating their feature maps. Predictions are made at 3 stages, with the first set of predictions being made by the most compact stage of the network and later predictions being made after subsequent upsampling and concatenation of feature maps.

6.2.1.1 Network Inputs

The input to the YOLOv3 network is a 480x480 pixel image with 3 color channels. YOLOv3 is composed of only convolutional and upsampling operations, making it a fully convolutional network. All convolutions are applied with zero-padding to preserve the output dimensions of the network throughout.

6.2.1.2 Darknet-53

A feature extractor called Darknet-53 (referring to the 53 convolutional layers which it is comprised of) is first applied to the input. The spatial dimensions are not downsampled through pooling in the YOLOv3 network but through strided convolutions which helps to preserve small local features. The full Darknet-53 architecture is shown in table 6.1. A series of convolutional blocks, as illustrated in figure 6.3, each using a specified number of kernels are applied. Each convolutional block consists of a 2D convolution, followed by the application of batch normalisation and the leaky ReLU activation function. Many of these convolutional blocks are applied in residual blocks, with these short range skip connections being applied to the entries in table 6.1 which include two convolutional blocks followed by a residual addition.

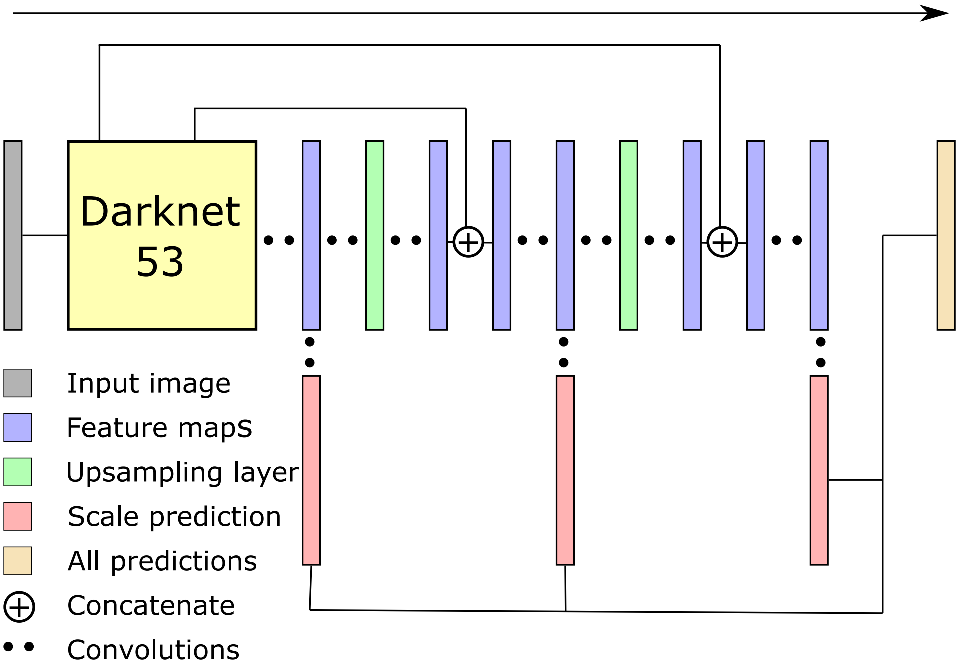


Figure 6.4: The YOLOv3 network architecture is composed of a Darknet-53 feature extractor, followed by additional convolutional layers which are each followed by batch normalisation and a leaky ReLU activation but without short range skip connections. Predictions are made at 3 scales, with 3 predictions per grid cell. From left to right the scales are 1,2,3 and there are 15, 30 and then 60 grid cells along each axis. Scales 1, 2 and 3 therefore have grid cells which correspond to 32, 16 and then 8 pixels in the original image respectively. The architecture follows an encoder-decoder architecture where 2 long range skip connections are made after hidden layers are upsampled. The predicted boxes from all 3 layers are the output of the network.

The network utilises 1x1 convolutions to extract the most salient features of the feature maps output by the 3x3 convolutional layers. This helps to minimise the number of learnable parameters whilst carrying forward the most important features. The application of a 1x1 convolution can be thought of as a weighted linear recombination of the feature maps.

Darknet-53 reduces the network to have width/height dimensions of 15x15. Each of these elements therefore corresponds to a 32x32 pixel region in the original image.

In the last layer before the network is downsampled to a width/height of 30x30, the feature maps are extracted so that they can be concatenated downstream in a long range skip connection. The same occurs before the network is downsampled to 15x15.

Each convolutional layer is initialised using He initialisation, as described in section 6.1.4. The convolutional blocks which are applied later in the network after the Darknet-53 architecture also use batch normalisation and leaky ReLU but have no short range skip connections.

Table 6.1: The Darknet-53 feature extraction network architecture composed of convolutional blocks, including convolutional layers with a stride of 2 which halve the dimensions of the subsequent layer. Grouped convolutional blocks have short range skip connections after each application. The number of kernels applied in each convolutional block, N_K, their size and the stride they are applied with are given in the second column. The third column denotes the number of times a set of layers is repeated before the next set of layers is applied. Long range skip connections extend from within Darknet-53 to later regions of the network.

Layer	Kernels (N_K, size, stride)	Repetition	Output size
Input image			480 × 480
Conv block	32, 3 × 3, 1	1	480 × 480
Conv block	64, 3 × 3, 2	1	240 × 240
Conv block Conv block Residual addition	32, 1 × 1, 1 64, 3 × 3, 1	1	240 × 240 240 × 240 240 × 240
Conv block	128, 3 × 3, 2	1	120 × 120
Conv block Conv block Residual addition	64, 1 × 1, 1 32, 3 × 3, 1	2	120 × 120 120 × 120 120 × 120
Conv block	256, 3 × 3, 2	1	60 × 60
Conv block Conv block Residual addition	128, 1 × 1, 1 256, 3 × 3, 1	8	60 × 60 60 × 60 60 × 60
Skip connection			60 × 60
Conv block	512, 3 × 3, 2	1	30 × 30
Conv block Conv block Residual addition	256, 1 × 1, 1 512, 3 × 3, 1	8	30 × 30 30 × 30 30 × 30
Skip connection			30 × 30
Conv block	1024, 3 × 3, 2	1	15 × 15
Conv block Conv block Residual addition	512, 1 × 1, 1 1024, 3 × 3, 1	4	15 × 15 15 × 15 15 × 15

6.2.1.3 Network Outputs

Additional convolutional blocks are applied to the network after Darknet-53 but the dimensions are preserved until the output is split into two paths of convolutions. The first path of convolutions acts on the feature maps to make predictions for the network in its current scale. The first set of predictions are made whilst the network is in its most compact form and are said to be made by 'scale 1'.

The outputs at each prediction scale are bounding boxes, described by a vector $\boldsymbol{B} = (x, y, w, h, o, \boldsymbol{c})$, where x and y are predicted corner locations and w, h are predicted box width and heights. The objectness score, o, represents the confidence with which the network predicts that the box holds an object. This may be an object from any class which is labelled in the training data. The confidence corresponds to the probability that the predicted box corresponds to any of the classes which were present in the training dataset. Finally, $\boldsymbol{c}$ is a vector holding the probabilities that the bounding box corresponds to each of the classes which the network was trained on. In the work presented in this thesis, only a single class is used for training and predicting objects. As $\boldsymbol{c}$ is predicted using a softmax activation function (equation 6.15), the output will always be a vector with a single entry whose value is 1. Therefore, the class score will not be optimisable.

The sizes and aspect ratios of the predictions are guided by 'anchor boxes'. Three anchor boxes are assigned to each scale of the network and each is used by every element of the network along the width/height axes to make a bounding box prediction. Each element of the network which makes a prediction is known as a 'grid cell'. For the width/height of 15x15, this leads to the first scale of the network making a total of 675 predictions.

The predictions for the x and y positions of each bounding box have a sigmoid activation function applied to them before the coordinate location of the corner of the grid cell in the image is added to correctly locate the corner of the predicted box. The network predicts the natural logarithm of the w and h terms. The exponential of these terms is taken rather than applying a conventional activation function. A sigmoid activation function is also applied to the object score prediction, o.

The second path of convolutions leads to an upsampling layer which doubles the network width and height dimensions to give it a width/height of 30x30. This upsampled layer is concatenated with the feature maps output from the second long range skip connection from Darknet-53. More convolutions are then applied until the convolutions split into two paths again. In the prediction path, 3 bounding boxes are predicted from each of the 900 grid cells. These predicted boxes are labelled as predictions from scale 2.

In the other path the upsampling and concantenation process is repeated, this time making use of a longer range skip connection from Darknet-53 so that 3 bounding boxes are predicted for each of the grid cells in a 60x60 grid, with these predictions being labelled as from scale 3. In total the network proposes 14175 bounding boxes per input image.

6.2.2 YOLOv3 Loss Function

The network processes one minibatch of images at a time during weight training, predicting 14175 bounding boxes for each of them. The loss function is calculated on a per-minibatch basis using minibatch SGD with an Adam optimiser. The loss is aggregated through averaging over all images in a minibatch and over all the minibatches in an 'epoch of training' (one pass through the entire training data set) using an average over all minibatches. The loss function for YOLOv3 is a combination of 5 terms relating to:

1. bounding box x, y

2. bounding box w, h

3. object score o for predicted boxes associated with a truth box

4. object score o for predicted boxes not associated with a truth box

5. class score prediction, c

where as mentioned in section 6.2.1.3 the class score prediction is uniformly 1 after the application of the softmax activation function for all predicted bounding boxes. Predicted boxes are associated with truth boxes which they are expected to provide the highest quality prediction for. The procedure for associating a truth box with a predicted box is explained in full in section 7.2.5.

The loss for each minibatch $l_{minibatch}$ is calculated for each prediction scale seperately and then aggregated over all 3 scales before being aggregated over the images in the minibatch:

$$l_{minibatch} = \frac{\sqrt{\sum_{i=1}^{3} w_i l_i}}{8} \tag{6.21}$$

where w_i is a scalar weight defined externally from the network for scale i and l_i is the loss computed for that scale. The w_i terms were set to 1 in all studies in this thesis. The division is by the number of images in a minibatch, which was 8 throughout the studies performed in this thesis.

For a given scale the loss components from each contribution to the loss are summed:

$$l_i = l_{x,y} + l_{w,h} + l_{obj} + l_{noobj} + l_{class} \tag{6.22}$$

where each of these loss terms are summed over all bounding box predictions from all grid cells but are not summed over the minibatch:

$$l_j = \sum {\alpha_j}^2 \tag{6.23}$$

where α_j is one of the loss contributions, $\alpha_{x,y}$, $\alpha_{w,h}$, α_{obj} or α_{noobj}, which are defined in sections 6.2.2.1-6.2.2.4 below. Due to the softmax and single class, α_{class} is always equal to 0, providing no contribution to the loss. The other α terms are MSE terms as the network is learning the non-linear regression between the input images and the properties of the truth boxes.

6.2.2.1 Positional Loss Term

The contribution to the loss from the x, y predictions, $\boldsymbol{B}_{x,y}$, made by the network is determined as:

$$\alpha_{x,y} = M_{x,y,w,h}(\hat{t}_{x,y} - \boldsymbol{B}_{x,y}) W_{w,h} w_{x,y,w,h}, \tag{6.24}$$

where $\hat{t}_{x,y}$ is the x, y truth value and $W_{w,h}$ is a scaling factor depending on the size of the truth box which is defined in section 6.2.2.5. These terms are arrays. $w_{x,y,w,h}$ is a scalar weight for the $l_{x,y}$ loss term which is set to 1 in all studies presented.

$M_{x,y,w,h}$ is a binary mask array. The mask depends on whether the minibatch is processed in a warm-up minibatch or a normal minibatch. The warm-up minibatch case is set out in section 6.2.2.6. In a normal minibatch this mask is 1 for all array entries where a truth box has been assigned to a given predicted box and is 0 elsewhere. This ensures that only predictions with an associated truth box contribute to $l_{x,y}$.

6.2.2.2 Size Loss Term

The contribution to the loss from the width and height predictions of the network is:

$$\alpha_{w,h} = M_{x,y,w,h}(\hat{t}_{w,h} - \boldsymbol{B}_{w,h})W_{w,h}w_{x,y,w,h}. \tag{6.25}$$

This is very similar to eqn 6.24, except that the widths and heights of the predicted boxes, $\boldsymbol{B}_{w,h}$ and the truth boxes, $\hat{t}_{w,h}$ are used to compute this term.

As is the case with the $\alpha_{x,y}$ term, contributions to the $\alpha_{w,h}$ term in normal training come only from predicted boxes which have an associated truth box.

6.2.2.3 Truth Associations Loss Term

Both the object score loss and the non-object score losses both depend on the predicted object scores. The object score loss has similarities and differences to the x,y and w,h loss terms. It is computed via:

$$\alpha_{obj} = M_{obj}(\hat{t}_{obj} - \boldsymbol{B}_{obj})w_{obj}, \tag{6.26}$$

where $\hat{t}_{obj}$ and $\boldsymbol{B}_{obj}$ are truth and predicted object score arrays and M_{obj} is an array mask which is 1 wherever a truth box is assigned and 0 otherwise. This is identical to $M_{x,y,w,h}$ during calculation of the normal loss but differs during warm-up minibatches, where M_{obj} is unchanged. $\hat{t}_{obj}$ takes a value of 0 or 1, depending on whether the given grid cell and anchor has an associated truth box, which happens to be exactly the same as M_{obj} in this case. The scalar weight $w_{obj} = 50$ is the only scalar weight not set to 1 in the calculation of the loss. As a result of the array masks, once again there are only contributions to this term in the loss from predicted boxes which have an associated truth box.

6.2.2.4 No Truth Associations Loss Term

The non-object score loss takes into account the maximal intersection over union (IOU) score (defined in section 6.2.4) for each prediction made by the network which does not have an associated truth box as well as the object score prediction. It is calculated as the product of the negative object mask, $(1 - M_{obj})$, the maximal IOU score for each prediction, IOU_{max} and a scalar weight $w_{noobj} = 1$:

$$\alpha_{noobj} = (1 - M_{obj})\boldsymbol{B}_{obj}(IOU_{max} < T_{IOU}^{loss})w_{noobj}. \tag{6.27}$$

The $(IOU_{max} < T_{IOU}^{loss})$ term is a mask which is zero wherever the maximal IOU is larger than a given threshold value for the IOU, T_{IOU}^{loss}, which was chosen to be 0.33 in the work presented in this thesis. This maximal IOU is determined using the truth boxes for the given network scale that the loss is being calculated for.

The overall effect of this term is designed to allow the network to recognise where predictions are made with little overlap with truth boxes using the T_{IOU} condition, avoiding training on cases where truth boxes overlap with a grid cell but have not been assigned to it. The object scores of all remaining predictions with no associated truth box, as well as their IOU scores are minimised through this term in training.

6.2.2.5 $\mathbf{W_{w,h}}$ Scaling Factor

The scaling factor applies a weight to each truth box based on the area of the truth box. It is defined as:

$$W_{w,h} = 2 - \frac{(e^{\hat{t}_w \hat{t}_h})}{n_{pixels}}, \tag{6.28}$$

where $\hat{t}_w$ and $\hat{t}_h$ are the truth values for the width and height of the truth boxes. The exponential of these values is taken because the natural logarithm is taken during preprocessing. n_{pixels} is the number of pixels in the input image (480x480). The total effect is for the scaling factor term to be 2 for very small truth boxes and 1 for truth boxes which are the same size as the image with the intention of allowing the network to learn more difficult cases.

6.2.2.6 Warm-up Loss Calculation

For the first 2 epochs of training, the loss is calculated differently than during following epochs in order to try and speed up the convergence towards the best weights. The training which occurred during these first 2 epochs is denoted by the term ‘warm-up training’. During warm-up training $l_{x,y}$ and $l_{w,h}$ were calculated differently than during normal minibatches. The equations to calculate the losses for these terms are different from section 6.2.2.1 and section 6.2.2.2 only in the fact that 3 terms were changed:

1. $\hat{t}_{x,y}$
2. $\hat{t}_{w,h}$
3. $M_{x,y,w,h}$.

During normal training $\hat{t}_{x,y}$ and $\hat{t}_{w,h}$ are only non-zero for predictions corresponding to array indices where predicted boxes have an associated truth box assigned to them. During normal training $M_{x,y,w,h}$ is a mask which is non-zero only for the same case.

When warm-up training, the $\hat{t}_{x,y}$ term has the location of the centre of each grid cell added to it for all array indices which do not correspond to a truth box. The $\hat{t}_{w,h}$ term is also altered to add ‘fake’ truth boxes to the array entries with no truth association. The fake truth box sizes are guided by the anchor boxes for each scale. Finally, the $M_{x,y,w,h}$ mask is set to 1 for every array index, regardless of whether there is a truth association or not.

The combined effect of these modifications to $t_{x,y}$, $\hat{t}_{w,h}$ and $M_{x,y,w,h}$ is designed to add ‘effective truth boxes’ for all the predictions made which have no associated truth box, in order to improve learning of the x, y, w, h parameters over the warm-up minibatches. It guides the randomly initialised weights towards predicting bounding boxes in the centre of the grid cells and with the same dimensions as the anchor boxes before allowing the network to specialise in making predictions of the input images.

6.2.3 Filtering Predictions

Once the network is fully trained, it can be used to predict bounding boxes in an image. It is useful to discard poor predictions based on their properties. Conventionally, a threshold is applied on the object score, T_{obj}, so that only high quality predictions are retained. In the work presented in chapter 7 a second threshold, T_{SPV}, on the sum of the pixel values contained in the predicted box was also used. The summed pixel value (SPV) is not a conventional part of most applications YOLOv3 but was found to be useful in these studies.

6.2.4 Non-maximum Suppression of Predicted Boxes

Even after the filtering of predicted boxes, there may be a large number of overlapping predicted boxes which remain for any given image. In order to determine which of the boxes are optimal, a technique known as greedy non-maximal suppression (NMS) is applied. This attempts to retain the highest quality predictions and discard other predictions which are sufficiently similar. NMS can be used to discard predicted boxes which have a high IOU with other predicted boxes, where the IOU is defined as the area of intersection between two boxes divided by the union of their area:

$$IOU = \frac{\boldsymbol{B}_1 \cap \boldsymbol{B}_2}{\boldsymbol{B}_1 \cup \boldsymbol{B}_2}. \tag{6.29}$$

The predicted boxes are sorted into reverse object score order. The box with the highest object score is kept and then has its IOU computed with each of the other boxes. If this value is larger than a predefined threshold, T_{NMS}, then the box is discarded. This algorithm is implemented iteratively until no boxes remain which have not been kept. Using a low T_{NMS} value results in few predicted boxes being retained, whilst at high T_{NMS} value requires predicted boxes to have a large overlap before being discarded. In the case that there is a single class label for the predicted boxes, the NMS algorithm is described in full in appendix A.5.

6.2.5 Determining Performance

Once NMS has been applied to the predicted boxes, the predicted boxes can be labelled as true positives (tps) and false positives (fps). The truth boxes can also be labelled as false negatives (fns) (to avoid confusion with trigger primitive (TP), the classification of true positives will be denoted as lower case tp and follow a similar convention for the other categories).

For every image with a truth box, the IOU of the truth boxes with each of the predicted boxes is determined. Starting with the predicted box with the highest object score, as long as the IOU is larger than a given threshold, T_{IOU}, this box is assigned to the truth box which it has the largest IOU with and is labelled as a tp. In all the studies presented in this thesis only a minimal IOU ($T_{IOU} = 1e-10$) is required to label a tp. A predicted box can only be assigned to a single truth box and a truth box can only be assigned to a single predicted box. If there is no unassigned truth box that a predicted box can be assigned to, it is labelled as a fp. Any truth boxes which were not assigned to a predicted box are labelled as fns. This process is described in full in appendix A.6.

In object proposal networks there is not a coherent concept of true negatives (tns). In the loss function, the grid cells that truth boxes were assigned to determined whether a prediction was associated with a truth box. The loss function is designed to optimise the ability of the network to make a prediction guided by a similarly sized anchor box centred in the same grid cell as the truth box. However, l_{noobj} does not penalise predictions from other grid cells, or using close but non-optimal anchors in the same grid cell due to the T^{loss}_{IOU} as explained in section 6.2.2.4. Clearly there is no guarantee that a prediction from a grid cell associated with a truth box will be the truth box with the highest object score, or IOU with the truth box. It may also be removed after application of NMS. This is unproblematic when determining the performance of the YOLOv3 network as the only quantities which need to be determined are the efficiency, ϵ, of predicting tps and the rate of predicting false positives. The efficiency is defined as the proportion of truth boxes which were assigned to a predicted box:

$$\epsilon = \frac{tp}{tp + fn}. \tag{6.30}$$

6.3 Summary

In this chapter the concept of machine learning as a way to approximate the MLE of a probability distribution has been introduced. How it can be applied to identify the optimum parameters for a given model has been explained. The theory underpinning ANNs and CNNs has been set out and the full YOLOv3 object proposal network architecture has been described, including the loss function, post-prediction processing and labelling of the predicted boxes.

CHAPTER 7

Impact of ML on the Trigger Pipeline

This chapter describes the implementation of the YOLOv3 network, the reasons behind its selection, and presents an analysis of its performance, both for detecting individual neutrino signals and using the predicted boxes as inputs to a SNB trigger. The YOLOv3 network was implemented using the Tensorflow [143] and Keras [144] Python packages and built upon an implementation of the network architecture by H. Anh et al [145].

Firstly, possible avenues to introduce ML into the SNB trigger will be discussed and the choice of the YOLOv3 network will be explained. Next the image preparation will be explained, followed by the choice of hyperparameters for the network and the training of the network weights, followed by a discussion of the training metrics. Subsequently, an analysis of the efficiency and false positive rate of detecting neutrino signals when applying thresholds to the network predictions is presented. Several promising working points are then used to determine the performance of the SNB trigger in comparison to the performance from the TCs identified using the rules based algorithm in chapter 5.

The YOLOv3 codebase was jointly developed by Joel Greer and Raul Stein, a collaborator within the Bristol DUNE group. The derivation of the optimal anchor boxes explained in section 7.3.1 was carried out by Raul Stein. The training which is presented in this chapter was run on an Nvidia T4 tensor core GPU.

7.1 Introducing ML to the Trigger Pipeline

7.1.1 DAQ Considerations

The possible scope to introduce ML methods to improve the SNB trigger is from the reception of the ADC values in the FELIX card to the HLF. It may make use of the raw ADCs, TPs, TC or other pre-processed information. Wherever ML is introduced it needs to be able to keep up with data reception and have a SNB trigger latency of less than 10 seconds. Each DAQ RU processes the ADC values on each APA separately but it is also possible to combine the data on the RU CPU server or add a GPU to the RU to accelerate image processing. Data from all RUs is passed to the MLT, which presents another opportunity to implement an improved trigger. Alternatively, it may be possible to implement a trigger pipeline which identifies high-interest detector regions and which causes the data from these to be read out to the HLF where it can be processed on a CPU or GPU cluster.

There are limitations in the bandwidth of data that can be passed between the different DAQ components as explained in section 3.3. To pass the collection wire readout from the front-end of a RU to its CPU server a $46\,\mathrm{Gbit\,s^{-1}}$ link would be required rather than $10\,\mathrm{Gbit\,s^{-1}}$. Therefore, it is necessary to either implement rapid data compression or to process and aggregate the data before passing it downstream. This could take the form of any region of interest (ROI) determination algorithm, including the existing TPG algorithm.

Application of CNNs to LArTPCs and liquid scintillator neutrino detectors has been studied in a number of experiments. Due to a TPCs being essentially a video readout of ionisation in a detector, CNNs are well suited to this data. Previous studies have used CNNs to label particle interactions (including neutrino interactions) by utilising simulated images produced using deconvolved and corrected ADCs from a number of detector channels over a number of detector ticks. In some cases ADC samples are used and in some it is hits derived from offline reconstruction algorithms.

7.1.2 CNNs Which Use Reconstruction Information

DUNE collaboration members have studied the performance of a convolutional visual network (CVN) [146] which utilises information from collection and induction channels for identification of beam neutrino interactions, the number of protons, charged and neutral pions and neutrons in the image as well as the type of charged current interaction and whether the neutrino was an antiparticle. Hits derived using offline reconstruction algorithms [147] are used to select a region of interest in each detector event record. From this an image of 500 wires by $3200\,\mu\mathrm{sec}$ (in 500 time bins) is created which is the input of the CVN. This approach was found to improve on the existing neutrino selection efficiency for neutrinos with a reconstructed energy less than $5\,\mathrm{GeV}$ but utilises information which is not available in online triggering, such as hits derived from offline reconstruction algorithms. Reconstruction of a single event can take on the order of seconds of computation time. If a similar approach were to be implemented making use of TPs to identify off-beam neutrino interactions, then it would likely need to be implemented in the HLF. The beam neutrino interactions have a much higher expected energy spectrum than SN neutrinos, meaning that a low energy trigger to pass neutrino candidate images to the HLF would need to be found. The rules-based clustering algorithm could be thought of as one such algorithm. In this case the additional use of a neural network to provide a probability score that an image contained a neutrino could be used to provide weights for the SNB counting trigger.

The use of pixel-wise labelling has been implemented at MicroBooNE, another LArTPC detector [148]. Once again, these studies use reconstructed information to create images which are processed by an 'Ultimate coupling of Registration and Segmentation with deep Nets' (U-ResNet) CNN. ADCs are deconvolved and 6 time ticks are summed together in the creation of these images (MicroBooNE uses the same 2 MHz sampling rate as DUNE). The reconstructed vertex information was used to select 512x512 pixel images as network inputs. This network was used to label tracks and electromagnetic showers which achieved a pixel misclassification rate of less than 5% for both classes and has applications in differentiating electron and muon neutrino interactions. This kind of pixel-wise labelling could allow better quality clusters to be determined by allowing the exclusion of hits from pixels which are labelled as background. When considered for online use, this approach has similar disadvantages to the previous one and also requires extra processing rather than replacing one or more steps in the baseline SNB trigger.

Another technique tried at MicroBooNE was the use of a 'Faster Region-based CNN' (Faster R-CNN) [149] architecture based on AlexNet [133] to detect beam neutrinos in images with cosmic ray backgrounds [150]. This utilises an object detection network which proposes rectangular bounding boxes to encapsulate the region containing the neutrino signal and achieved an efficiency of 87% at a purity of 73%. The images input to this network were raw ADC samples and used reweighted real cosmic ray background data as noise. 864x756 pixel images were created by downsampling by a factor of 4x and 8x for the channel and tick dimensions respectively. Implementing something similar for online neutrino identification and triggering in the DUNE DAQ system is still impractical as it would again require a separate low energy neutrino trigger to identify detector regions which could be sent to the MLT or HLF level to be processed. However, the proposal of a bounding box provides a possible avenue to replace at least the rules-based clustering stage of the baseline SNB trigger.

7.1.3 Boosted Decision Trees

The use of boosted decision trees (BDTs) has been successfully applied to classify particles. These use decision trees which make cuts on multiple variables to classify data. A small difference in the training data set can cause a large difference in the resulting decision tree. For this reason bootstrap aggregation, where many training datasets are created by uniformly subsampling the whole dataset (with replacement), along with the use of multiple trees, each created from a bootstrapped dataset can be used to improve the performance of the classifier [151]. Often a multitude of decision trees (called random forests) are used, which enforces the use of a random subset of the features in each decision tree. This helps to stop the trees used in bootstrap aggregation becoming correlated due to some features being overwhelmingly strong predictors. The output of the random forest is the class selected by the most trees. The MiniBooNE collaboration has studied the use of BDTs to identify and differentiate electron and muon neutrino interaction from a beam source using reconstruction information [152] [153]. They found that a BDT approach was more successful than a basic artificial neural network implementation. BDTs can only make decisions using the variables they are provided and as a result information aggregation in the DUNE DAQ is necessary for their use. The most basic aggregated information in the rules-based algorithm are the TPs and TC properties. A potential task for BDTs in the SNB trigger would be to classify whether a TC is a neutrino signal or not.

7.1.4 Sequence Information

A final area where ML could be introduced is in the final SNB trigger decision. There is the possibility to replace the SNB trigger with a BDT which uses the TCs in a time window as its input and classifies whether these were the result of a SNB or not. Alternatively, neural networks which look at sequence information such as recurrent neural networks (RNNs) or a long short-term memory (LSTM) network [154] could be used to classify a time sequence of TCs as a SNB. However, the signal expected from a SN depends on the mass hierarchy, progenitor properties and SN model parameters. Sequence information studies would require SN neutrino spectra to ensure the classifier is sufficiently generalised.

7.1.5 Neutrino Identification in DUNE DAQ

One other study has been done on implementing CNNs, making use of the collection plane wires to label images in DUNE as low energy (SN neutrino signal), high energy or background cases within the online DAQ system [155]. A CNN architecture taken from [156] was trained on images composed of raw ADCs after setting all pixels whose pixel intensity did not meet a threshold to zero. A ROI region was found around the non-zero pixels in an image of 480 channels and 2.25 ms, which was downsampled to a 64x64 pixel image used as the input to the CNN. This network is suitable for implementation in the HLF as it requires a data bandwidth of only 22 $\mathrm{Mbit\,s^{-1}}$ for each RU after the ROIs have been found. The network achieved an accuracy of 94% on identifying SN neutrino interactions and an accuracy of 99.53% on non-empty background noise ROIs. Using the ROI threshold, it was found that only 2% of background images had non-zero values, offering a factor of 50 in potential speedup of the network, whose inference time was 1.6 ms. This network approach was also implemented in an FPGA (Zynq Ultrascale + XCZU9EG) using high level synthesis (HLS) [157] tools to study the practicality of implementing it within the UD. It was determined that use of the FPGA improved the inference speed by a factor of only 1.7 and the power efficiency by a factor of 2.6. This study limits identification of a SN neutrino to a single interaction per drift time per TPC and may not be robust to the case of multiple interactions, as might occur from a nearby SNB. It could be used to replace the TP and TC steps in the rules-based SNB trigger.

7.1.6 Selection of the YOLOv3 Network

Taking inspiration from both the object proposal network which has been used at MicroBooNE to identify neutrino signals and from the use of a CNN to perform fast inference within the DAQ or HLF, in this thesis a study is presented on the use of a fast object proposal network which uses minimal input information. The TPs which were produced from FPGA-based TPG have been used to create pre-thresholded images where a YOLOv3 [142] architecture is used to predict bounding boxes for neutrino interactions. The intention was to study whether an object proposal network could be used to predict neutrino bounding boxes with a high enough accuracy and low enough false positive rate using only TPs as the network inputs. This can be thought of as replacing the clustering stage in the rules-based SNB trigger.

A YOLOv3 network was chosen over a region-based neural network (R-CNN) architecture mainly due to the inference speed. Faster R-CNN [149] has an inference speed of 5 frames per second (FPS) on a K40 GPU, whilst you-only-look-once (YOLO) architectures can achieve dozens of FPS. This is partly because YOLO networks make region (bounding box) proposals and perform classification at the same time, whereas R-CNN based networks utilise both a region proposal network (RPN), which tells a subsequent detection network where to look, and a classification network.

YOLOv3 runs at 34 FPS on a Titan X GPU but simpler YOLO architectures, such as You-Only-Look-Once version 2 (YOLOv2) can achieve speeds of 67 FPS or above depending on the hardware used to run them [158]. This work is intended to use YOLOv3 to evaluate whether object detection networks are feasible for this task with this data and then to adjust the architecture until the required inference speed to keep up with the DAQ is achieved. For example, much of the decrease in speed of YOLOv3 over YOLOv2 was a result of a more complex feature extraction network, using 53 convolutional layers instead of the 19 used in YOLOv2. This decreases the speed of the network but improves its performance.

YOLOv3 was chosen over You-Only-Look-Once version 4 (YOLOv4) [159] - the most recent update to the YOLO network family when these studies were carred out. YOLOv4 implements data augmentation methods which are not well-suited to the sparse images that were fed to the network in these studies. Therefore YOLOv3 was used instead, despite the modest performance improvement that YOLOv4 promises on non-sparse images.

There is a rich literature of simplified YOLO models, including smaller network architectures which can offer significant speed-up at the cost of some loss in performance as seen in table 7.1. It has been shown that YOLO networks can be implemented within FPGAs [160] [161], showing that they may have the potential for application in either the HLF or within the FPGA resources in the RUs. Tools such as High Level Synthesis for Machine Learning (HLS4ML) [162] or LeFlow [163] have made this process increasingly feasible in recent years.

Table 7.1: Performance metrics of YOLO architecture versions on a Titan X GPU. Values are taken from [164][142][158]. The mean average precision (mAP) is a metric commonly used for comparing the performance of object detection networks on a given image dataset. It is the area under the curve formed by plotting the efficiency of detections against the proportion of reported detections which were correct. The Visual Object Class (VOC) dataset is considered to be easier to achieve a high mAP than the Common Objects In Context (COCO) dataset.

Model	Layers	FLOPS($\times 10^9$)	FPS (GPU)	mAP (Dataset)
YOLOv1	26	not reported	45 (Titan X)	63.4 (VOC)
YOLOv1-Tiny	9	not reported	155 (not reported)	52.7 (VOC)
YOLOv2	32	62.94	67 (Titan X)	76.8 (VOC); 48.1 (COCO)
YOLOv2-Tiny	16	5.41	244 (not reported)	23.7 (COCO)
YOLOv3	106	140.69	34 (Titan X)	55.3 (COCO)
YOLOv3-Tiny	24	5.56	220 (not reported)	33.1 (COCO)

7.2 Image Preparation

YOLOv3 requires images as inputs. This section describes all of the image preparation steps used to create and transform the images which were fed into the network. These images were synthesised from the TPs generated by applying the FPGA-based TPG algorithm.

In order to train a YOLOv3 network, the images which include a supernova interaction signal also require a truth box which encompasses this signal. Once the truth box has been assigned to a particular scale of the network it can be used for training.

In this section the image and truth box creation algorithms are explained, followed by how truth boxes were assigned to the neutrino signals in the images. The downsampling algorithm and the assignment of truth boxes to a given scale of the network is then set out. The breakdown of images into training and testing datasets and finally the image augmentations applied to the training dataset are then explained.

7.2.1 Image Synthesis

TPs contain only simple summary information on a single channel in the detector. Each event contained the TPs for the 12 APAs in the 1x2x6 geometry for 2.246 msec (4492 detector ticks). Only the TPs from collection channels were used in image creation.

Images were created on a per APA basis. Each APA contains 960 collection channels, so the images created were 960x4492 pixels, with the x-axis representing the collection channel number in the APA and the y axis being the detector tick number in the event. The start and end ticks of the TPs were shifted 16 ticks earlier due to the average delay caused by the FIR filter (explained in section 5.1.3).

Collection wires are spaced along the z-axis and run parallel to the y-axis of the detector. The electric field is applied along the x-axis of the detector, meaning that the images are projections of the charge deposited in this volume over 1 drift time onto the x-z plane.

The TPs were mapped on to the image from their start tick to their end tick. The pixel value, $P_{i,j}$, assigned to all pixels on the channel over the tick extent of the TP was the SADC of the TP. The pixels values were then converted to the range 0-255 using a saturation value to preserve the occupancy of the pixel value spectrum. A saturation value of 5000 was chosen using figure 5.4(a). It was observed that whilst the SADC of TPs mapped to SN neutrino interactions exceeded 5000 5% of the time, the proportion of background TPs which exceeded a SADC of 5000 was less than 2×10^{-7}. The new pixel values, $P^*_{i,j}$, were determined by applying

$$P^*_{i,j} = P_{i,j} \times \frac{255}{5000} \tag{7.1}$$

and subsequently assigning any pixel values which remained above 255 to a value of 255.

The images fed to the network had 3 colour channels. The images were generated so that the pixel values were grayscale. Instead of changing the network architecture, the grayscale pixels were copied to all 3 color channels.

7.2.2 Truth Box Labelling

Each neutrino interaction may lead to TPs on multiple APAs but for each neutrino interaction on an APA only one truth box was generated. In this dataset, 3 neutrino interactions were simulated for each event in the detector simulation. These interactions were located randomly in the 1x2x6 geometry. This meant that a single APA (and therefore image) could have up to 3 truth boxes and a single neutrino could have multiple truth boxes, with 1 per APA it produced a TP on.

The source of each TP was determined via backtracking through the GEANT4 event record. For each APA with at least 1 neutrino interaction which resulted in 1 or more TPs, a truth box was created for each of the neutrino interactions. The extent of each truth box was set to correspond to the extent of the charge deposited from this neutrino in the APA. Padding of 3 channels and 11 ticks were added to each side of the truth box. Truth boxes which would extend beyond image boundaries were limited to the image boundary.

7.2.3 Downsampling Algorithm

A custom downsampling algorithm was used to downsample the 960x4492 pixel images to the 480x480 pixels required to be input to the YOLOv3 network. The images were downsampled by a factor of 2 along the channel axis and a factor of 9 along the tick axis. This required cropping the images by 172 ticks before downsampling. For each image the 172 tick region along the tick axis with the minimum sum of pixel values was determined across the entire width of the channel axis. The image was then cropped by removing this region. The signals should be equivariant to translation along the tick axis. The first pixel in the cropped image was taken to be the first pixel after the removed region and the last pixel was taken to be the pixel before the removed region.

The images were then downsampled by summing the pixel values of the cropped images according to:

$$P^{ds}_{k,l} = \sum_{i=1+(2k)}^{2+(2k)} \sum_{j=1+(9l)}^{9+(9l)} P^{*}_{i,j}, \tag{7.2}$$

where $P^{ds}_{k,l}$ is the pixel value for the downsampled image with the index k along the channel axis and l along the tick axis. i and j are the respective indexes of the cropped image.

Once $P^{ds}_{k,l}$ was determined, the pixel values were rescaled to be within an 8-bit range via:

$$P^{ds:*}_{k,l} = P^{ds}_{k,l} \times \frac{\max P^{*}_{i,j}}{\max P^{ds}_{k,l}} \tag{7.3}$$

and converted to integer values.

The truth boxes were shifted to account for the removed region of the image and also downsampled in size by 2 times along the channel axis and 9 times along the tick axis.

The downsampling factors of 2 and 9 were chosen primarily for practicality. The images represent the charge deposited in the TPC volume. The collection channels had a wire spacing of 4.75 mm and the free electrons in the TPC were simulated with a constant drift speed of 1.6 mm/µsec. Each of the pixels in the downsampled images therefore had an aspect ratio of 132:100 along the channel and tick axes, with each downsampled pixel containing the ionization deposited in a 9.5 mm by 7.2 mm area of the x-z plane.

7.2.4 Image Datasets

The images were generated from the '10k events containing SN neutrinos' dataset. The TPs which resulted from using an ADC threshold of 20 ADC counts were also used in the work presented in this chapter, allowing the performance of the baseline SNB counting trigger and the ML-based SNB counting trigger on the same TPs to be compared.

The images which included TPs from SN neutrino interactions were split into a training dataset, a validation dataset and an evaluation dataset. The training dataset was used to train the network. The validation dataset was used at the end of each epoch of training to determine the performance of the network on a dataset independent from the training dataset. This was useful to check for consistency with the loss and other metrics calculated on the training dataset and to identify any overfitting on the training dataset. The evaluation dataset was used to assess the performance of trained models and determine the efficiency for SN neutrino interactions. The training/validation/evaluation split was 80%/10%/10%, with 23,656 images in the training dataset, and 2952 images in each of the validation and evaluation datasets.

The images produced which only included TPs from background sources were not used for training the network. This was because the network was already able to make use of predictions which were not associated with a truth box in the training dataset to learn when predictions should be given a low objectness score. These images were instead used to determine false positive rates for the predictions made by the trained model. A dataset of 36,000 of these background only images was created.

7.2.5 Associating Predicted Boxes With Truth Boxes

Nine anchor boxes were used in the training of the network, with 3 assigned to each scale of the network. The anchor box dimensions (in units of pixels in the downsampled images) are presented in table 7.2. The derivation of the dimensions and number of these truth boxes is explained in section 7.3.1. To determine whether the network was training to make good predictions, metrics based on whether a predicted box was associated with a truth box were used. Truth boxes were associated with predictions made using the anchor box which had the greatest IOU with the truth box when both were fixed to have their bottom left corner at the origin. As each anchor box was associated with a given scale and therefore set of grid cells, the grid cell for this scale which the centre of the truth box was located in was assigned to be the one making the particular prediction (guided by the anchor box) which was associated with a truth box. The total number of truth boxes assigned to each dataset and the proportion assigned to each anchor box are also shown in table 7.2. The smallest and largest anchor boxes have the lowest occupancy but the proportion of truth boxes assigned to each scale was found to be $26.4 \pm 0.4\%$ for scale 3, $42.3 \pm 0.2\%$ for scale 2 and $31.1 \pm 0.3\%$ for scale 1. These proportions of truth boxes are not equal but no scale dependent weighting was used in the loss function because the goal was to train to get the optimal predictions from the dataset overall rather than to get equally good predictions from each scale.

Table 7.2: Size of anchor boxes for each network scale in units of pixels in the downsampled images, where A_w and A_h are the width and height of the anchor boxes. Also presented is the number of truth boxes, T_N, which were assigned to each anchor box for the training, validation and evaluation datasets, denoted with superscript t, v and e respectively.

Scale	Index	A_w	A_h	T_N^t	T_N^v	T_N^e
3	1	3	2	26 (<1%)	4 (<1%)	3 (<1%)
3	2	3	3	1290 (3.4%)	166 (3.5%)	156 (3.3%)
3	3	4	3	8757 (23.0%)	1123 (23.4%)	1081 (22.6%)
2	4	6	6	6637 (17.4%)	811 (16.9%)	791 (16.6%)
2	5	9	12	4931 (12.9%)	661 (13.8%)	688 (14.4%)
2	6	13	33	4542 (11.9%)	550 (11.5%)	551 (11.5%)
1	7	26	16	4524 (11.9%)	572 (11.9%)	599 (12.5%)
1	8	39	51	5371 (14.1%)	657 (13.7%)	662 (13.9%)
1	9	217	143	2020 (5.3%)	245 (5.1%)	242 (5.0%)
Sum				38098	4789	4773

7.2.6 Image Augmentations

Data augmentation was used to extend the effective size of the training dataset. This can help to avoid overfitting of the training dataset and improve the generalisation of the network performance.

Two different image augmentations were applied to the training dataset during training. The first of these was a random shift of the image and the second was a rotation of the image.

In the random shift, the image and the bounding boxes were translated randomly by up to half of the image width and up to half of the image height. Any pixels which were translated beyond the edge of the image were wrapped around to the other side of the image. If any truth boxes were shifted beyond the image boundary then the truth box and the image were shifted back so that they were instead only shifted up to the edge of the image. As each APA covers 2 TPCs, with 1 on either side of the APA, the random shift helped to ensure that the training dataset was equivariant to translation and should result in improved generalisation during training.

A rotation of 180 degrees was applied to 50% of the augmented images instead of a random rotation because the aspect ratio of the images was not 1:1.

7.3 Choice of Hyperparameters

The readiness of a network to minimise the loss during training is highly dependent on the hyperparameters which are used. These hyperparameters include all the values which have an impact on training, including the scaling factors introduced in section 6.2.2.

The number and size of the anchor boxes used to train a YOLOv3 network has to represent the training dataset well in order for truth boxes to be assigned to anchor boxes in balanced proportions. If an insufficient number of truth boxes are assigned to a particular scale, then the training dataset for the scale will be small. This is likely to result in increased overfitting when compared to the other scales of the network for the same number of epochs of training.

This section will start by explaining the technique used to derive the optimal anchor boxes for training on this dataset and report their sizes. The learning rate optimisation techniques and the early stopping conditions for training will then be explained. Apart from the anchor boxes, the tuning of hyperparameters has not been studied in detail.

7.3.1 Anchor Boxes

As explained in section 6.2, the anchor boxes are used to guide the predictions made by the network. For the network to be able to make good predictions, the truth boxes in the datasets must be well represented by the anchor boxes used.

The optimal anchor boxes were determined through the application of a k-median clustering algorithm which is set out in full in appendix A.7. This was applied to the truth boxes in the training dataset after the downsampling was applied to find the 9 anchor boxes which best represented the truth boxes. The final anchor boxes are plotted in figure 7.1 relative to the width and height of the downsampled images.

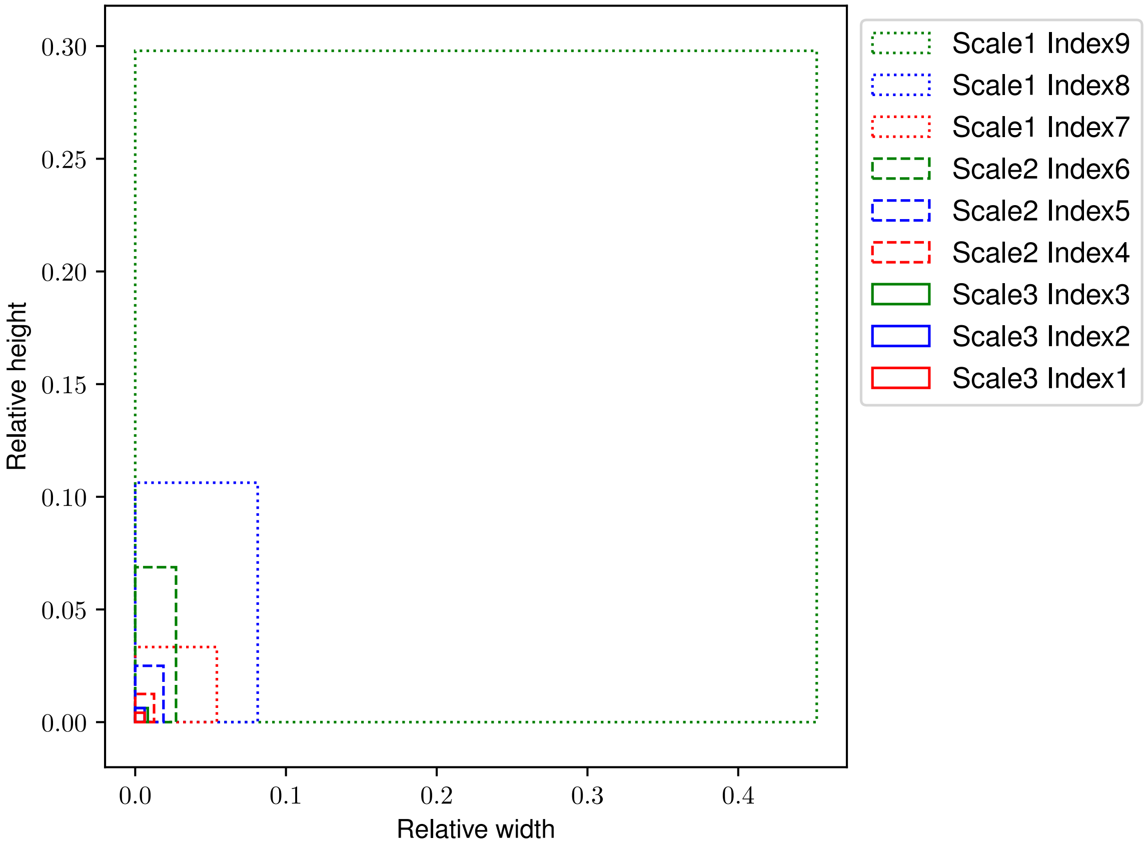


Figure 7.1: Visualisation of the optimal anchor boxes as determined via k-median clustering. Anchor box labels correspond to those in table 7.2. The anchor boxes are plotted relative to the downsampled image width and height. The width corresponds to the channel dimension and the height corresponds to the time dimension.

7.3.2 Learning Rate and Stopping Condition

The Adam optimiser set out in algorithm 3 was used to optimise training. The parameters used during training were set to: $\rho_1 = 0.9$, $\rho_2 = 0.999$ and $\epsilon = 1 \times 10^{-7}$.

The initial learning rate was set to $l_r = 1 \times 10^{-4}$. During training the loss value calculated from the validation dataset was monitored. If the validation loss did not decrease for 3 epochs the learning rate was decreased to 10% of its previous value. This approach was used to allow the network to probe the weight parameter space of the loss without skipping over local minima due to a high learning rate.

An early stopping condition was used to determine after how many epochs to stop training. If the validation loss did not decrease after 10 epochs then training was halted. The final model weights were then saved and used to evaluate the performance of the network.

7.3.3 Other Hyperparameters

During training the initial 2 epochs were set to use the warm-up loss defined in section 6.2.2.6. As set out in section 6.2.2, the only fixed scaling factor which was not set to 1 was used in the truth associations loss term defined in section 6.2.2.3. A minibatch size of 8 images was used for training.

7.4 Training Metrics

It is important to assess whether it was possible to successfully train the network on the dataset used. During training a number of metrics were computed on the training and validation datasets. In this section the progression of these metrics during training is studied and interpreted to show whether the training of the network was successful or not. These metrics were:

1. the overall training and validation losses, L^t and L^v respectively
2. the average object scores for predictions which were associated with a truth box for the predictions from each scale of the network, labelled $\bar{o}^t_{i:obj}$ for the training dataset ($i = \{1,2,3\}$ to denote the scale) and $\bar{o}^v_{i:obj}$ for the validation dataset.
3. the average object scores for predictions which were not associated with a truth box for the predictions from each scale of the network, labelled $\bar{o}^t_{i:noobj}$ for the training dataset and $\bar{o}^v_{i:noobj}$ for the validation dataset
4. the average IOU values for predictions which were associated with a truth box from each scale of the network, labelled as $\bar{I}^t_{i:obj}$ for the training dataset and $\bar{I}^v_{i:obj}$ for the validation dataset
5. the average positional loss term for each scale, labelled as $\bar{L}^t_{i:x,y}$ and $\bar{L}^v_{i:x,y}$ for the training and validation datasets respectively
6. the average size loss term for each scale, labelled as $\bar{L}^t_{i:w,h}$ and $\bar{L}^v_{i:w,h}$ for the training and validation datasets respectively
7. the average truth associations loss term for each scale, labelled as $\bar{L}^t_{i:obj}$ and $\bar{L}^v_{i:obj}$ for the training and validation datasets respectively
8. the average no truth associations loss term for each scale, labelled as $\bar{L}^t_{i:noobj}$ and $\bar{L}^v_{i:obj}$ for the training and validation datasets respectively

These metrics were computed by averaging across the predictions on all images in each minibatch and then taking the average for all minibatches in each epoch. The minibatch size stayed constant throughout training and validation. The metrics for the validation dataset were calculated after training on the epoch was complete. For the training dataset, these metrics were calculated during training on each minibatch. As minibatch SGD (described in section 6.1.3) was used for training, these metrics represent an average value for each of the metrics over the epoch of training. The effect of this was that the metrics for the training dataset were on average calculated half an epoch earlier than the metrics for the validation dataset.

It is expected that the overall loss during training will be minimised. As a YOLOv3 network has many different contributions to the loss, care needs to be taken when interpreting general metrics, such as the overall loss. The $\bar{o}_{i:obj}$ and $\bar{L}_{i:obj}$ metrics can be used to show whether predictions of the objectness score improve during training. Similarly the $\bar{o}_{i:noobj}$ and $\bar{L}_{i:noobj}$ metrics show whether the network learns to predict low objectness scores for regions of the image which do not contain a truth box. The $\bar{I}_{i:obj}$, $\bar{L}_{i:x,y}$ and $\bar{L}_{i:w,h}$ metrics can be interpreted to show whether the localisation of the predicted boxes improved during training.

7.4.1 Overall Loss

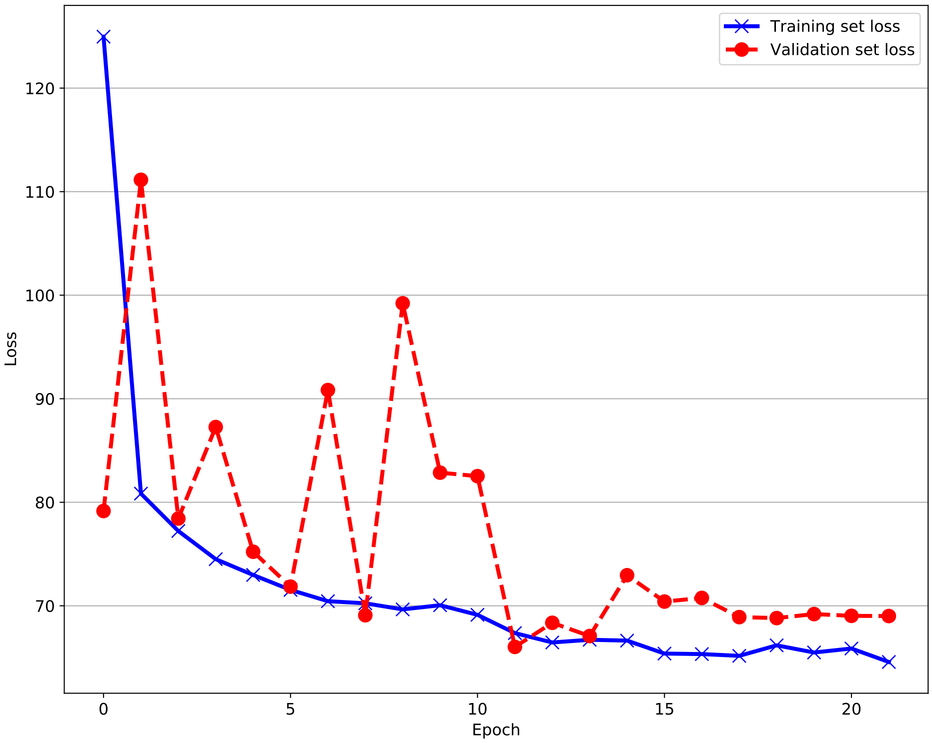


Figure 7.2: The average loss for each epoch of training calculated on the training dataset, L^t, and the validation dataset, L^v.

The L^t and L^v for each epoch of training is plotted in figure 7.2. The L^t decreased monotonically over each epoch whilst L^v fluctuated above and below L^t until a minimum value occured on epoch 11. After this point the validation loss after each epoch of training stabilised whilst L^t appeared to asymptote towards a value of 65. The L^t remained below L^v by a consistent amount from epochs 17-21, which indicates that the network began to overfit to the training dataset but remained well generalised to the validation dataset.

7.4.2 Predicted Object Scores

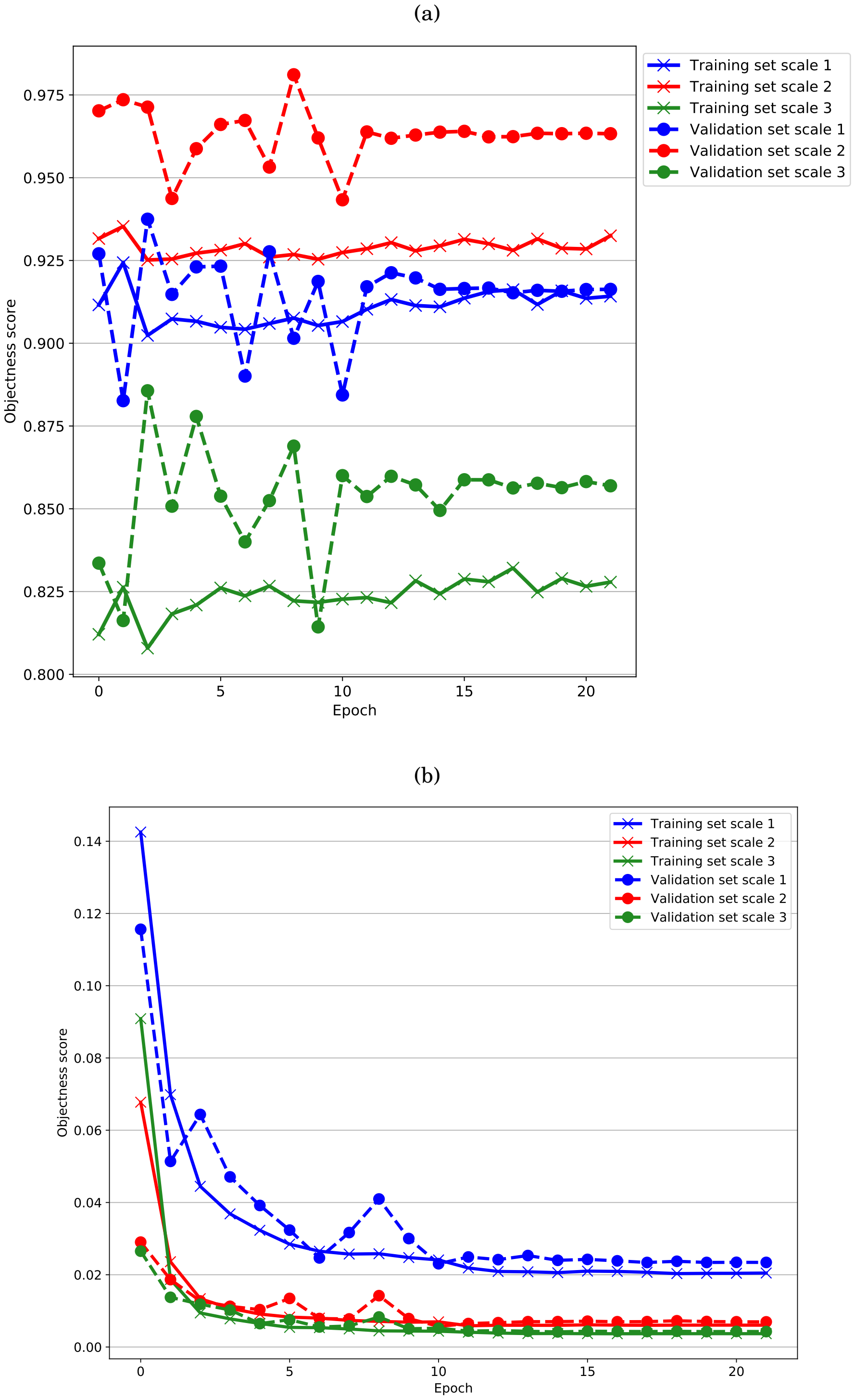
The $\bar{o}^t_{i:obj}$ and $\bar{o}^v_{i:obj}$ metrics predicted for scales $i = \{1,2,3\}$ are shown in figure 7.3(a). The $\bar{o}^t_{i:obj}$ and $\bar{o}^v_{i:obj}$ metrics did not increase significantly after the first epoch of training for most cases. The largest increase in $\bar{o}_{i:obj}$ occured for $\bar{o}^v_{3:obj}$, where it increased from 0.83 to 0.86.

It can be seen that the oscillation of the $\bar{o}^v_{i:obj}$ metrics, which was apparent over the early epochs decreased significantly after epoch 11, when the validation loss reached a minimum. The $\bar{o}^t_{i:obj}$ value was seen to keep increasing over the last few training epochs whilst the $\bar{o}^v_{i:obj}$ metrics stayed constant for each scale, which again suggested that the network started to overfit to the training dataset but was still well generalised to the validation dataset.

It can also be seen from figure 7.3(a) that the $\bar{o}^v_{i:obj}$ metrics for scales 1 and 3 were consistently greater than the $\bar{o}^t_{i:obj}$ metrics for scales 1 and 3. This was not as prominent for scale 2 but was still the case for the majority of epochs. It is not known why this was the case.

It was expected that the $\bar{o}_{i:obj}$ terms would optimise throughout training. The general trend of the $\bar{o}^t_{i:obj}$ terms was to slightly increase over the course of training after the initial warm-up epochs. The $\bar{o}^v_{i:obj}$ terms appeared to converge to a value and stop oscillating over the later epochs. It was theorised that due to the simplicity and sparsity of the images that the network was able to train to its optimal performance for $\bar{o}_{i:obj}$ predictions with its given hyperparameters and training dataset within a single normal training epoch. It may have been the case that the localisation of predicted boxes were only well optimised after epoch 11, which could account for the decrease in volatility of the $\bar{o}^v_{i:obj}$ metrics after this point.

The $\bar{o}^t_{i:noobj}$ and $\bar{o}^v_{i:noobj}$ metrics followed the expected behaviour during training more closely as can be seen in figure 7.3(b). These metrics rapidly tended towards zero in the warm-up epochs for both the validation and training datasets and continued to do so in the subsequent epochs of training. For scales 2 and 3 the $\bar{o}_{i:noobj}$ metrics decreased to <0.01. The $\bar{o}_{1:noobj}$ appeared to asymptote towards a value of 0.02. The $\bar{o}_{1:noobj}$ may be expected to have an increased value compared to $\bar{o}_{2;3:noobj}$ as a result of each grid cell covering a greater number of pixels in the input image and therefore containing some signal a higher proportion of the time. The network may have learned to associate high objectness scores with non-sparsity or high pixel values, within the scope of predictions by the grid cell. The truth boxes were assigned to the anchor boxes which were most similarly sized as explained in section 7.3.1. However, the signal remained in the image and may have led to predictions with high objectness scores which were guided by



(a)

(b)

FIGURE 7.3. Comparison between the training and validation datasets of average object scores for each epoch of training for predicted boxes with (a) and without (b) associations to truth boxes. The average objectness scores were calculated separately for each scale, with scales 1,2 and 3 being plotted in blue, red and green respectively. The values calculated for the training dataset are marked by a cross and joined by a solid line. The values calculated for the validation dataset are marked by a dot, joined by a dashed line.

other anchor boxes. An attempt to account for this was made in the l_{noobj} term, as explained in section 6.2.2.4. This discounted the contribution of any o_{noobj} predictions to the l_{noobj} term if the predicted box had an IOU>0.33 with a truth box. This would have helped to ensure that few predicted boxes devoid of any signal would lead to high $o_{i:noobj}$ predictions. However, these cases were not excluded from the $\bar{o}_{i:noobj}$ metric. As a result many $o_{i:noobj}$ predictions were likely to have high objectness scores. The low values of the $\bar{o}_{i:noobj}$ metrics despite this was reassuring because it showed that it was likely that the network had learned to differentiate between regions containing signal and regions without signal.

7.4.3 Prediction Localisation

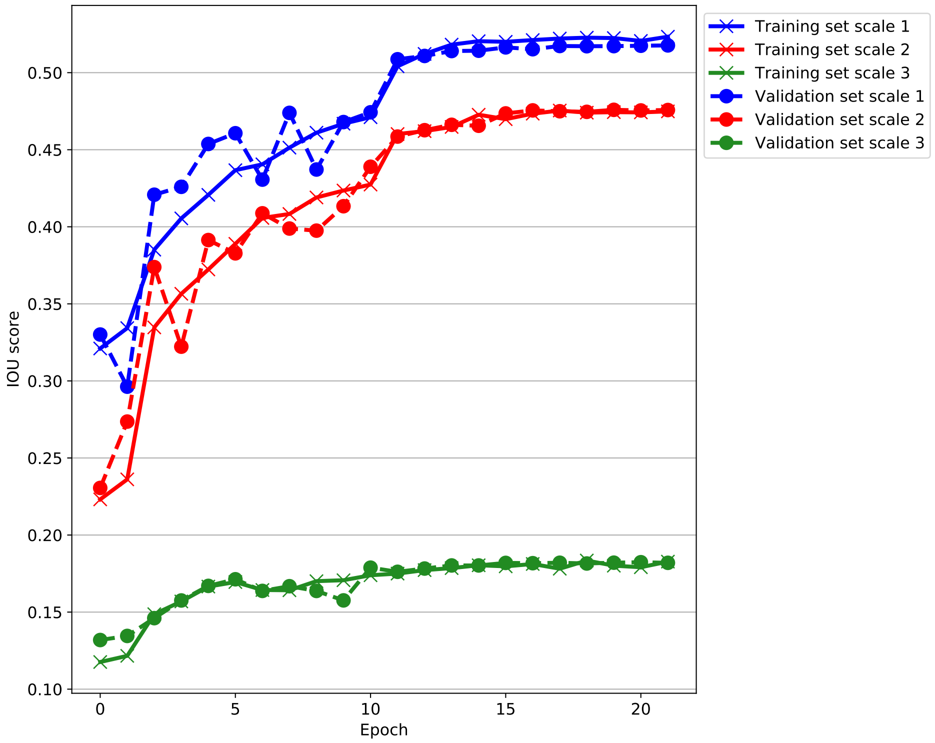


Figure 7.4: The average IOU score, $\bar{I}_{i:obj}$, for each epoch of training calculated on the training dataset and the validation dataset. The average IOU scores were calculated separately for predictions from each scale, with scales 1,2 and 3 being plotted in blue, red and green respectively. The $\bar{I}_{i:obj}$ values calculated for the training dataset are marked by a cross and joined by a solid line. The values calculated for the validation dataset are marked by a dot, joined by a dashed line.

The $l_{x,y}$ and $l_{w,h}$ loss terms should be minimised as the network weights were optimised to improve predictions of the localisation of signal regions. It has already been noted that the increased stability of the o_{obj} predictions on the validation set after epoch 11 has no basis when only considering $\bar{o}_{i:obj}$ metrics. It was observed from figure 7.4 that the $\bar{I}^t_{i:obj}$ and $\bar{I}^v_{i:obj}$ metrics improved monotonically for the training dataset as the network was trained. This continued beyond epoch 11 but there was little further improvement. Once again beyond epoch 11 the $\bar{I}^v_{i:obj}$ appeared to become more stable.

After the warm-up epochs, the network was trained to optimise predictions of localisation on only the truth boxes. The jump from the warm-up loss to normal loss resulted in the greatest improvement in $\bar{I}^v_{i:obj}$ over a single epoch for scales 1 and 2. The $\bar{I}^v_{1:obj}$ metric improved from 0.3 to 0.52 and $\bar{I}^v_{2:obj}$ improved from 0.27 to 0.47 during training. The localisation of predictions for scale 3 was poorer; $\bar{I}^v_{3:obj}$ increased from 0.14 to 0.18. The largest anchor box assigned to scale 3 was only 4 pixels across and 3 pixels tall in the downsampled images, which may be evidence that the network struggled to accurately predict the localisation of small charge deposits. When taken together, the combination of the lower values for $\bar{I}_{3:obj}$ and $\bar{o}^v_{3:obj}$ metrics suggested that scale 3 produced poorer predictions than scales 1 or 2.

7.4.4 Contributions To The Loss

The average contributions to the loss from $\bar{L}_{i:x,y}$, $\bar{L}_{i:w,h}$, $\bar{L}_{i:obj}$ and $\bar{L}_{i:noobj}$ were also investigated to help interpret the training of the network. It was observed from figures 7.5(a) and 7.5(b) that $\bar{L}_{i:x,y}$ and $\bar{L}_{i:w,h}$ were rapidly minimised over the warm-up epochs. This was expected because during the warm-up epochs the predicted boxes were being guided to match the anchor boxes placed in the centre of each grid cell. During the subsequent epochs of normal training, $\bar{L}_{3:x,y}$ did not significantly improve, indicating that the position of the predicted boxes from scale 3 had reached their maximum accuracy. Though $\bar{L}_{2:x,y}$ and $\bar{L}_{1:x,y}$ did continue to minimise, the $\bar{L}^v_{i:x,y}$ values stabilised after epoch 11. The $\bar{L}_{i:w,h}$ metrics were found to similarly continue to decrease during the normal training epochs. However, $\bar{L}_{3:w,h}$ continued to minimise until epoch 11 of training, unlike $\bar{L}_{3:x,y}$.

Whilst overfitting can be inferred from comparing $\bar{L}^t_{i:x,y}$ and $\bar{L}^v_{i:x,y}$ for a given scale, it is unfair to directly compare the magnitude of $\bar{L}_{i:x,y}$ metrics between scales due to the imbalance in the number of assigned truth boxes and the $W_{w,h}$ scaling factor used. The same is true for the $\bar{L}_{i:w,h}$ metrics.

The $\bar{L}_{i:obj}$ metrics were difficult to interpret, much like the $\bar{o}_{i:obj}$ metrics but followed a consistent pattern shown in figure 7.5(c). The $\bar{L}^v_{i:obj}$ metrics were smaller than the $\bar{L}^t_{i:obj}$ metrics for scales 1 and 3 for every epoch. After oscillating around the $\bar{L}^t_{2:obj}$ values before epoch 11, the $\bar{L}^v_{2:obj}$ values also became smaller than the $\bar{L}^t_{2:obj}$ values for later epochs. A loss term determined from the validation dataset is not expected to have a loss smaller than one determined from the training dataset as this indicates that the network weights performed better on the validation dataset than on the dataset they were trained on. That this was the case was confirmed by figure 7.3(a).

It may have been the case that there was not a significant gradient over the l_{obj} term in the weight parameter space of the loss. The observation that despite l_{obj} being overwhelmingly the largest contribution to the loss, the network consistently improved on the $\bar{I}_{i:obj}$ metric may suggest that the truth associations loss term indeed formed a plateau in the weight parameter space of the loss but that the $l_{x,y}$ and $l_{w,h}$ loss terms were able to follow a gradient.

The magnitude of the $\bar{L}_{i:obj}$ terms was significantly larger than the other loss terms likely due to the weight of $w_{obj} = 50$ applied in equation 6.26. This weight is squared in equation 6.23 used to determine l_{obj}, the term averaged to find $\bar{L}_{i:obj}$. This was a basic attempt to upweight the contribution to the loss of the obj scores from prediction with associations to truth boxes, so that they may contribute a similar amount as the predictions which had no association with a truth box.

It was not clear why the $\bar{L}^t_{i:obj}$ increased after the warmup epochs for all scales, rising from the value at epoch 1 to the value at epoch 2.

The $\bar{L}^t_{i:noobj}$ metrics monotonically decreased during training, showing that the predictions which were not associated with a truth box learned to predict lower and lower objectness scores. This was consistent with figure 7.3(b).

7.4.5 Summary of Training Metrics

The overall loss, L, provided the simplest interpretation of training. It appeared that the network began to overfit to the training dataset but remained well generalised until training was stopped. The $\bar{o}_{i:obj}$ metric suggested that the network rapidly learned what looked like a neutrino interaction and what did not using these simple, sparse images. As expected, the $\bar{o}_{i:noobj}$ metrics showed that the network learned to predict low objectness scores for predictions which were not associated with a truth box.

The improvement in the localisation of predictions was indicated by the $\bar{I}_{i:obj}$ metric and increased steadily over the course of training up until epoch 11, where it stabilised. When this occurred, $\bar{o}^v_{i:obj}$ predictions also became more stable, suggesting that the network weights had become near-optimal. The localisation predictions from scale 3 were consistently found to be significantly worse than those from scale 1 or 2.

The metrics measuring the components of the loss were consistent with the $\bar{o}_i$ and $\bar{I}_{i:obj}$ metrics. The overall impression from the $\bar{L}_{i:x,y}$ and $\bar{L}_{i:w,h}$ metrics was that the network was rapidly able to learn to predict truth boxes of the dimensions of anchor boxes during the warm-up epochs and was then able to progressively improve predictions of the localisation of neutrino interactions in the images during the normal training epochs. It may have been the case that once the localisation predictions became good enough, the predictions often contained at least some of the neutrino signal. This could explain why the $\bar{o}^v_{i:obj}$ became more consistent over later epochs.

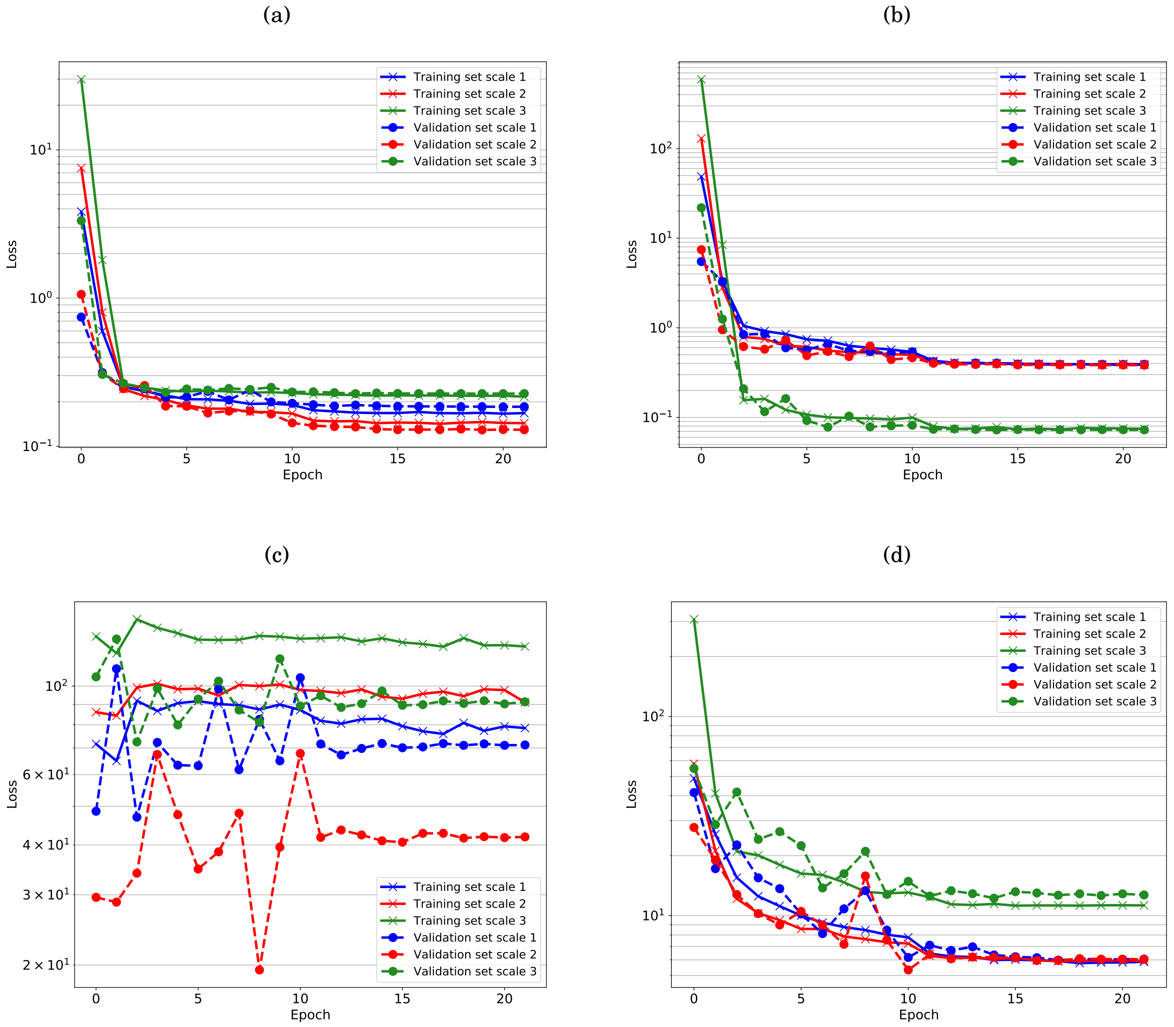


FIGURE 7.5. Comparison between the training and validation datasets of the average value the 4 contributions to the loss for each epoch of training. (a): The average loss from the positional loss term, $\bar{L}_{i:x,y}$. (b): The average loss from the size loss term, $\bar{L}_{i:w,h}$. (c): The average loss from the truth associations loss term, $\bar{L}_{i:obj}$. (d): The average loss from the no truth associations loss term, $\bar{L}_{i:noobj}$. The average loss values were calculated separately for each scale, with scales 1,2 and 3 being plotted in blue, red and green respectively. The values calculated for the training dataset are marked by a cross and joined by a solid line. The values calculated for the validation dataset are marked by a dot, joined by a dashed line.

7.5 Performance of YOLO Network

The images from the evaluation dataset containing SN neutrinos and the background-only dataset were passed to the network and the output predicted boxes were saved for analysis. Several stages of processing were applied to the predicted boxes to discard poor candidates. Firstly, a threshold on the objectness score, T_{obj} was applied to the predicted boxes. The sum of the pixel values encompassed by each remaining predicted box, the SPV, was then determined. In addition to the conventional T_{obj} applied in YOLOv3, a threshold on the SPV, T_{SPV}, was then applied to further reduce the number of predicted boxes and retain those which contained the greatest amount of coincident activity in the detector. Finally NMS was applied to the remaining predicted boxes. The final set of predicted boxes from the evaluation dataset were then labelled as tps and fns according algorithm 5. As the background dataset contained no truth boxes, the remaining predicted boxes could only be labelled as fps.

The efficiency and false positive rate of a number of combinations of T_{obj}, T_{SPV} and T_{NMS} were studied to determine how to optimise the performance of the predicted boxes from the network. Throughout this work a threshold of $T_{IOU} = 10^{-10}$ was used in the labelling of tps, meaning that predicted boxes only needed to be have a minimal overlap with a truth box in order to be labelled as a tp. It should be noted that although only a single predicted box could be mapped to a truth box as a tp, many predicted boxes could overwise pass the thresholds applied and the application of NMS. These were labelled as fps according to algorithm 5. The multiplicity of these boxes did not further contribute to the calculated efficiency but did contribute to the false positive rate. The multiplicity may, however, contribute to the performance of the SNB counting trigger and is discussed further in section 7.6.

In this section the results from a pilot study using the entire evaluation dataset and only 3000 images from the background dataset are first presented. This was used to determine the range of thresholds and prediction scales with which to process the full background dataset and to characterise the properties of the predicted boxes. The performance of the selected thresholds is then characterised for the full background dataset and the selection of thresholds to take forward to study the performance of the SNB counting trigger is made.

7.5.1 Characterising Predicted Boxes

The properties of the predicted boxes made by the network on the evaluation dataset were first investigated to more comprehensively determine whether the network had learned to differentiate between signals and background and to determine a set of T_{obj} values for which to investigate the network performance.

The distribution of $\boldsymbol{B}_{obj}$ predictions for predicted boxes with and without associated truth boxes are shown in figure 7.6. When considering predictions from all scales of the network, as shown in figure 7.6(a), it was clear that $\boldsymbol{B}_{obj}$ scores tended to be high for predictions with an

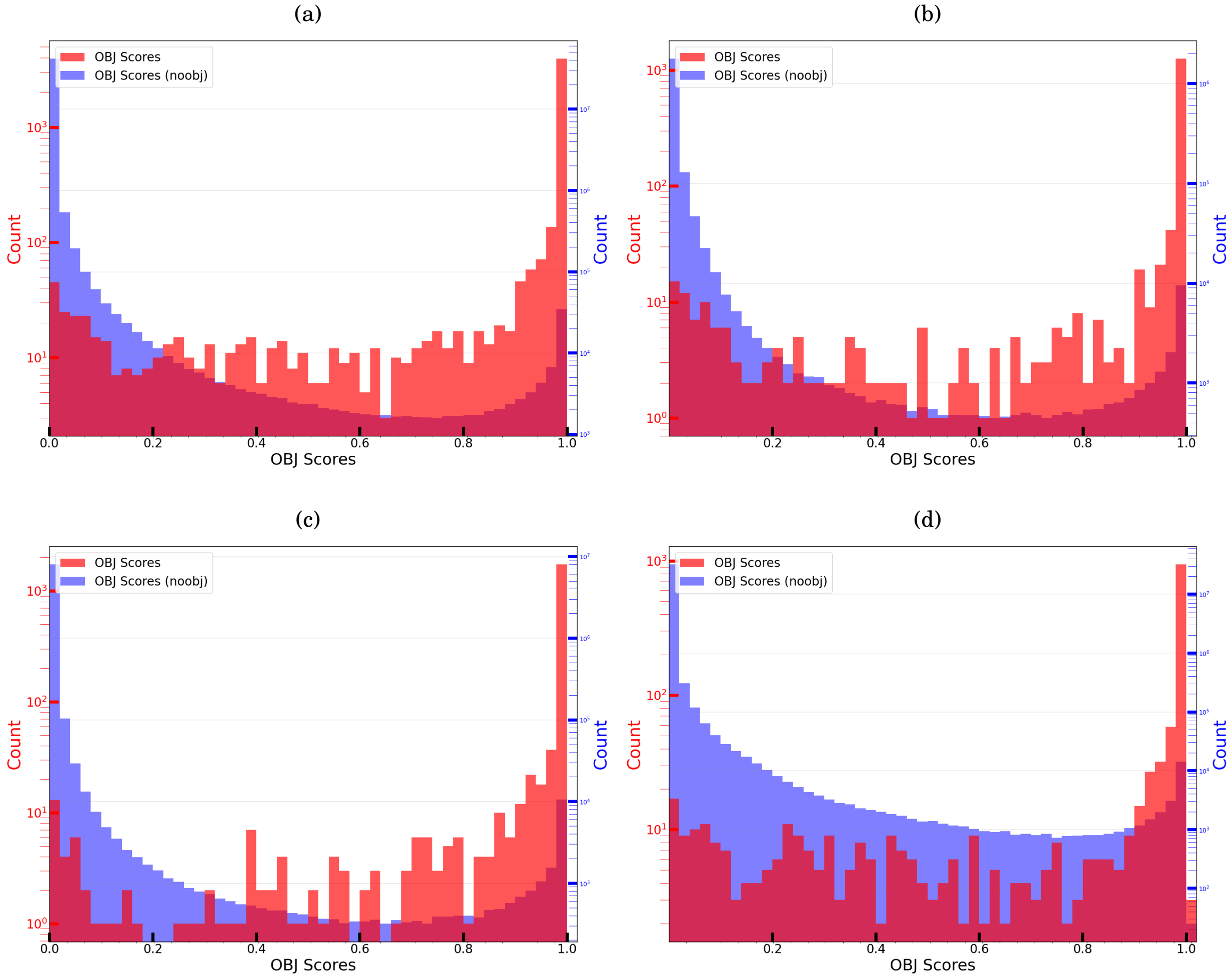


FIGURE 7.6. Predicted objectness scores, $\boldsymbol{B}_{obj}$, for predicted boxes with and without truth associations for the evaluation dataset from each network scale and combined. Predicted boxes with an associated truth box are plotted in red. Predicted boxes without an associated truth box are plotted in blue. The two sets do not share the same y-axis scale in order to show the contrasting distributions. (a): Predictions from all scales. (b): Predictions from scale 1. (c): Predictions from scale 2. (d): Predictions from scale 3.

associated truth box, with a mean value of 0.926. The vast majority of predicted boxes without an associated truth box were given very low $\boldsymbol{B}_{obj}$ scores, with a mean of 0.006 but the entire range was covered by both cases. There were peaks close to 0 and close to 1 for the predicted boxes without an associated truth box. It is possible that these predictions in the peak close to $\boldsymbol{B}_{obj} = 1$ were made by predictions local in an image to a SN neutrino signal or part of a SN neutrino signal but not assigned to a truth box.

The distribution of $\boldsymbol{B}_{obj}$ scores was found to be similar across all scales, as can be seen from figures 7.6(b) to 7.6(c). The proportion of $\boldsymbol{B}_{obj}$ scores for predictions without an associated truth

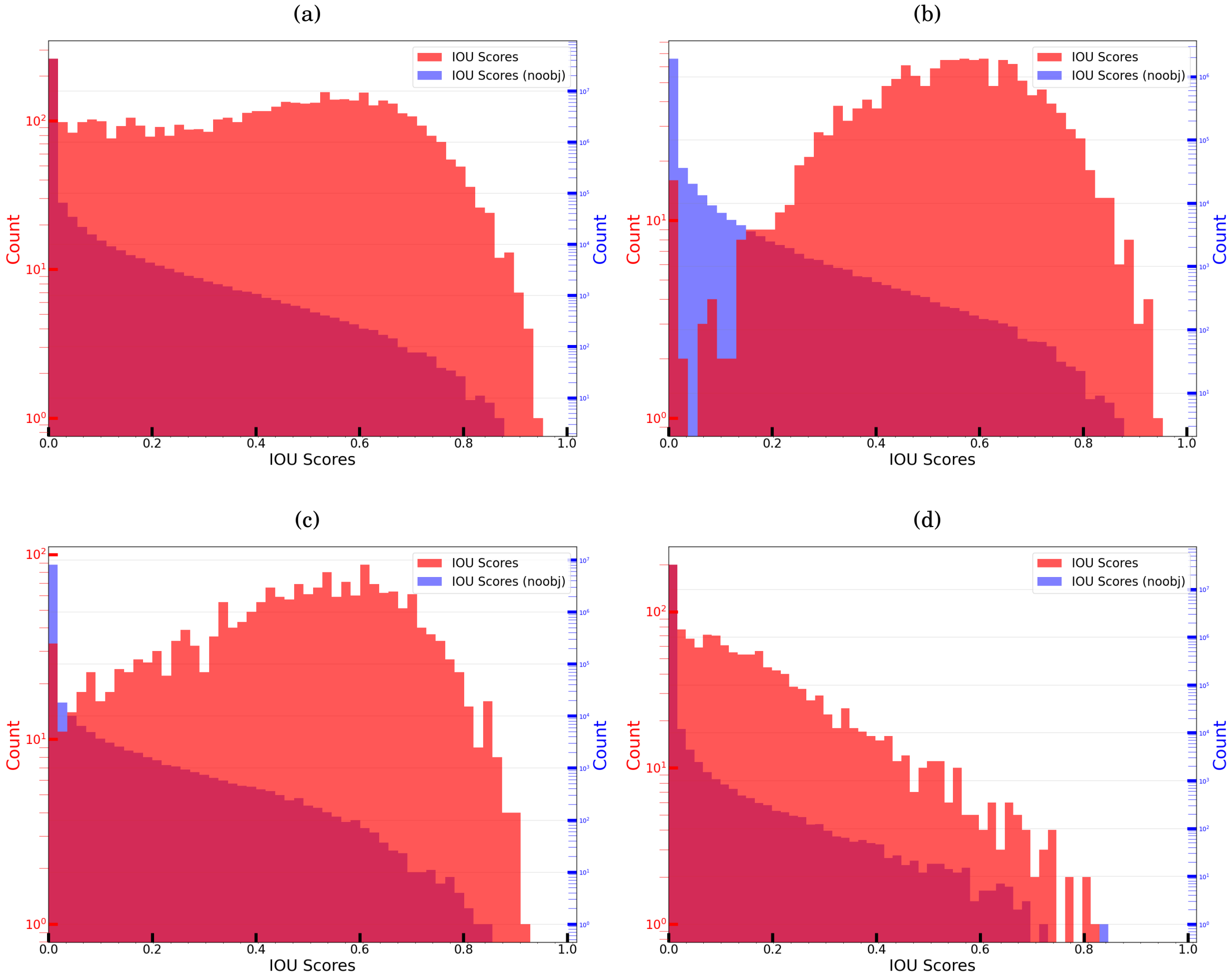


FIGURE 7.7. Maximum IOU with a truth box, $\boldsymbol{B}_{IOU}$, for predicted boxes with and without truth associations for the evaluation dataset from each network scale and combined. Predicted boxes with an associated truth box are plotted in red. Predicted boxes without an associated truth box are plotted in blue. (a): Predictions from all scales. (b): Predictions from scale 1. (c): Predictions from scale 2. (d): Predictions from scale 3.

box consistently appeared to reach a minimum around $T_{obj} = 0.8$ before increasing again. Below $T_{obj} = 0.8$ the distribution of $\boldsymbol{B}_{obj}$ scores for predictions with an associated truth box was roughly uniform. It appeared that predictions from scale 3 of the network were less able to differentiate between signal and background, with the predictions associated with a truth box having a mean $\boldsymbol{B}_{obj}$ of 0.873 compared to 0.925 and 0.965 for scales 1 and 2.

It was clear from figure 7.7(a) that the IOU scores from predictions with an associated truth box were successfully predicting the localisation of the SN neutrino signals to some extent, with a mean IOU of 0.411. It was clear that whilst the majority of predicted boxes without an associated truth box did not overlap truth boxes, with the mean IOU being 0.006. However, IOU scores up to 0.88 were found, indicating that some predictions which were not associated with a truth box were still successfully predicting boxes with a large overlap with truth boxes. Therefore it is likely that the network weights were able to somewhat generalise good predictions of localisation across different anchor boxes and potentially scales of the network.

There was a clear peak in the distribution of IOU scores for predicted boxes which were associated with a truth box from scale 1 around the mean IOU of 0.531 for these boxes as shown in figure 7.7(b). A similar case was found for scale 2, as shown in figure 7.7(c). However, for scale 3, this distribution more closely followed the distribution of IOU scores for predicted boxes which did not have an associated truth box. This suggested that the localisation of predicted boxes for predictions from scale 3 tended to be poorer than for the other scales.

In summary, it was clear that predicted boxes with an associated truth box predicted higher $\boldsymbol{B}_{obj}$ and IOU scores than predicted boxes without associated truth boxes. Whilst it was clear that the vast majority of predicted boxes without an associated truth box had low $\boldsymbol{B}_{obj}$ and IOU scores, there were indications that some of these predictions had larger $\boldsymbol{B}_{obj}$ scores than were typical. IOUs with truth boxes also covered a similar range to predicted boxes with truth associations, suggesting that at least some of these predicted boxes were well localised and that the network was able to generalise over scales when making localisation predictions. Predictions of localisation from scale 3 were poorer than from scales 1 or 2.

The rate of predicted boxes with $\boldsymbol{B}_{obj}$ scores from the background only dataset used in the pilot study is presented in figure 7.8. The distribution was similar to the $\boldsymbol{B}_{obj}$ distribution for predicted boxes with no associated truth box in figure 7.6(a), with the lowest rate of predictions found to occur close to $\boldsymbol{B}_{obj} = 0.8$. The vast majority of the rate for all $\boldsymbol{B}_{obj}$ bins was found to be from scale 3. However, even without the predictions from scale 3 and the use of a high object score threshold of 0.98 the rate of predicted boxes from background sources would only be reduced to 10 kHz per FD module.

7.5.2 Threshold Exploration

A set of T_{obj}, T_{SPV} and T_{NMS} thresholds were selected to perform a grid search and determine the optimal combinations to maximise the efficiency and minimise the FPR. This pilot study was also used to investigate the impact of only using predictions from scales 1 and 2. The set of objectness score thresholds was:

$$T_{obj} = \{0.8, 0.85, 0.9, 0.99, 0.999, 0.9999\}, \tag{7.4}$$

with the SPV thresholds

$$T_{SPV} = \{-1, 1, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000, 11000, 12000\} \tag{7.5}$$

and NMS thresholds

$$T_{NMS} = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.66, 0.75\}. \tag{7.6}$$

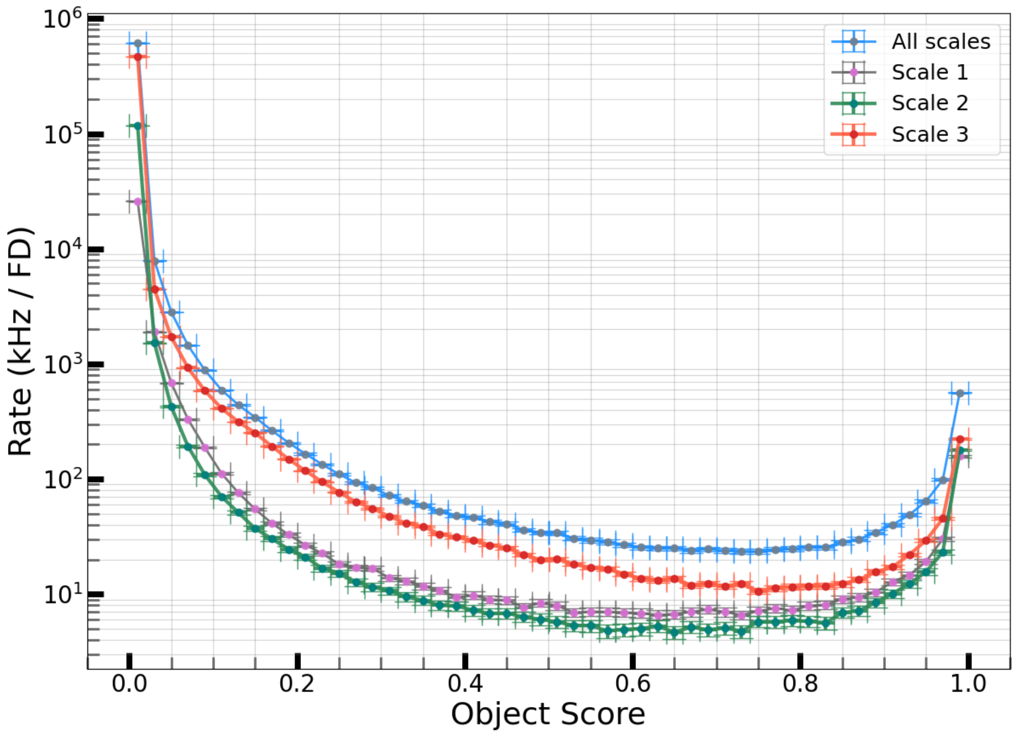


Figure 7.8: Distribution of rates (Hz per FD module) of predicted boxes from the background-only dataset as a function of predicted $\boldsymbol{B}_{obj}$ score. The rate from all predicted boxes is plotted in blue, from scale 3 in red, from scale 2 in green and from scale 1 in purple. 95% confidence error bars on the rate are too small to be visible for all scales except scale 1 only.

The loosest NMS threshold of $T_{NMS} = 0.75$ was used to characterise the impact of the SPV on the FPR. The FPR as a function of the T_{SPV} is plotted in figure 7.9(a). This pilot study used 3000 images from the background only dataset. The FPR which corresponded to a single fp prediction was 14.8 Hz per FD module. No predicted boxes had a SPV value of 5000 or larger in the pilot background-only dataset. However, it was clear that using a T_{SPV} was highly effective at reducing the FPR. The FPR for the loosest T_{obj}, 0.8, was reduced from 6.0×10^5 Hz/FD module to zero, whilst the tightest T_{obj}, 0.9999, was reduced from 2.6×10^3 Hz/FD module to zero. The use of $T_{SPV} = 1$ (discarding predicted boxes which did not contain any non-zero pixels) reduced the FPR to 2.2×10^5 Hz/FD module for $T_{obj} = 0.8$. Further limiting T_{obj} to 0.9999 reduced the FPR to 1.5×10^3 Hz/FD module. The large FPR which resulted with the use of high T_{obj} thresholds shows that there was still much scope for improvement in the training of the network. For predicted boxes resulting from the background only dataset, even when the predicted $\boldsymbol{B}_{obj}$ was high, the network was locating predicted boxes in empty regions of the image a high proportion of the time.

The impact of the T_{NMS} threshold was also investigated. Figure 7.9(b) shows how T_{NMS}

decreased the FPR after no T_{SPV} was applied. For $T_{obj} = 0.9999$, the FPR was only decreased from 2.6×10^3 Hz/FD module to 2.3×10^3 Hz/FD module when T_{NMS} was reduced from 0.75 to 0.1. It was more successful for $T_{obj} = 0.8$ where the FPR was reduced from 6.0×10^5 Hz/FD module to 4.1×10^5 Hz/FD module.

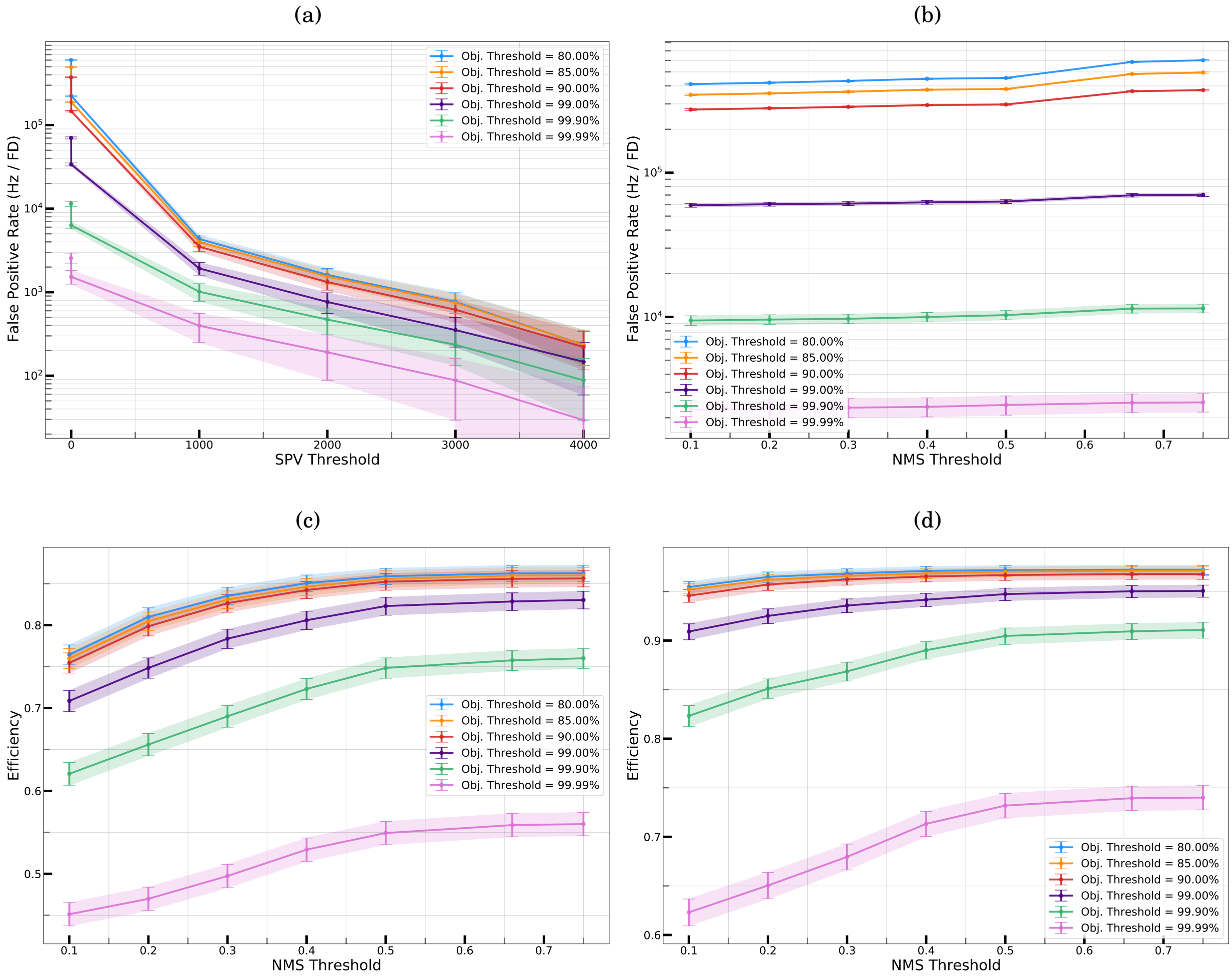


FIGURE 7.9. Exploration of thresholds and the resulting FPR and ϵ (pilot study). All error bars represent 95% confidence limits. (a): FPR for $T_{NMS} = 0.75$. (b): FPR without the application of a T_{SPV}. (c): ϵ for $T_{SPV} = 4000$. (d): ϵ without the application of a T_{SPV}.

As the FPR was least affected by variations in T_{NMS}, the response of the efficiency to T_{NMS} variations was characterised next. Figures 7.9(c) and 7.9(d) show how the efficiency decreased as T_{NMS} was decreased when using $T_{SPV} = 4000$ and no T_{SPV} threshold respectively. For both cases the efficiency appeared to asymptote towards a maximum value as the T_{obj} decreased across the entire range of T_{NMS} thresholds used. However, by increasing T_{obj} from 0.8 to 0.9999, the efficiency using $T_{NMS} = 0.75$ and no T_{SPV} dropped from 0.972 to 0.740. When using $T_{SPV} = 4000$

the drop was from 0.863 to 0.560.

The efficiency was observed in figure 7.9(c) to consistently drop by around 0.1 over the range of T_{NMS} thresholds where $T_{SPV} = 4000$ was used. A similar drop was observed in figure 7.9(d) for $T_{obj} = 0.9999$ but was less pronounced the higher the maximum efficiency was for other T_{obj} thresholds.

The efficiency was not observed to increase by more than 2% between $T_{NMS} = 0.5$ and $T_{NMS} = 0.75$ for any T_{obj}, appearing to have tended towards a maximum value. As T_{NMS} was significantly less effective at decreasing the FPR than T_{obj}, the efficiency was plotted against the FPR for $T_{NMS} = 0.75$ in figure 7.10(a). The resulting curve more clearly visualises the trade-off between reduced efficiency and reduced FPR when increasing the T_{obj} and T_{SPV} thresholds. The colors represent the different T_{obj} thresholds and the different points represent the different T_{SPV} thresholds used. The efficiency and FPR determined for all T_{SPV} thresholds up to 12000 were plotted and the points plotted for each T_{obj} are joined by a line. A logarithmic scale was used on the FPR axis. The highest efficiency at which no false positives were found in the pilot study is reported in table 7.3.

Table 7.3: Efficiency, ϵ, for T_{obj} thresholds when no false positives were found for the pilot study background dataset of 3000 images using $T_{NMS} = 0.75$. Efficiency values determined using predicted boxes from all scales is denoted by ϵ_{123} and from predicted boxes from scales 1 and 2 by ϵ_{12}. Efficiency values quoted with 95% certainty error bars.

T_{obj}	ϵ_{123}	ϵ_{12}
0.9999	$0.538^{+0.014}_{-0.014}$	$0.529^{+0.014}_{-0.014}$
0.999	$0.733^{+0.012}_{-0.012}$	$0.722^{+0.013}_{-0.013}$
0.99	$0.802^{+0.011}_{-0.011}$	$0.796^{+0.011}_{-0.012}$
0.9	$0.830^{+0.010}_{-0.011}$	$0.827^{+0.011}_{-0.011}$
0.85	$0.833^{+0.010}_{-0.011}$	$0.830^{+0.010}_{-0.011}$
0.8	$0.835^{+0.010}_{-0.011}$	$0.833^{+0.010}_{-0.011}$

The FPR could be reduced several orders of magnitude by increasing the SPV threshold. Whilst the precision of the FPR which could be probed in this pilot study was only 14.8 Hz/FD module it was possible to probe this precision and still achieve a efficiency greater than 0.8 for $T_{obj} \leq 0.99$. For these T_{obj} thresholds the FPR was reduced by more than 4 orders of magnitude, whilst for $T_{obj} = 0.999$ and $T_{obj} = 0.9999$, the FPR was reduced by 3 and 2 orders of magnitude respectively. The significance of the use of a T_{SPV} instead of a T_{NMS} can be seen by contrasting figure 7.10(a) with figure 7.10(b), where no T_{SPV} was applied before the application of NMS. The reduction in efficiency that resulted from the application of T_{NMS} thresholds was smaller than from the application of T_{SPV} thresholds but the reduction in FPR was comparatively insignificant.

Due to the high rate of predictions with high $\boldsymbol{B}_{obj}$ scores observed in figure 7.8 from scale 3 compared to scales 1 and 2, the performance of the network using only predictions from

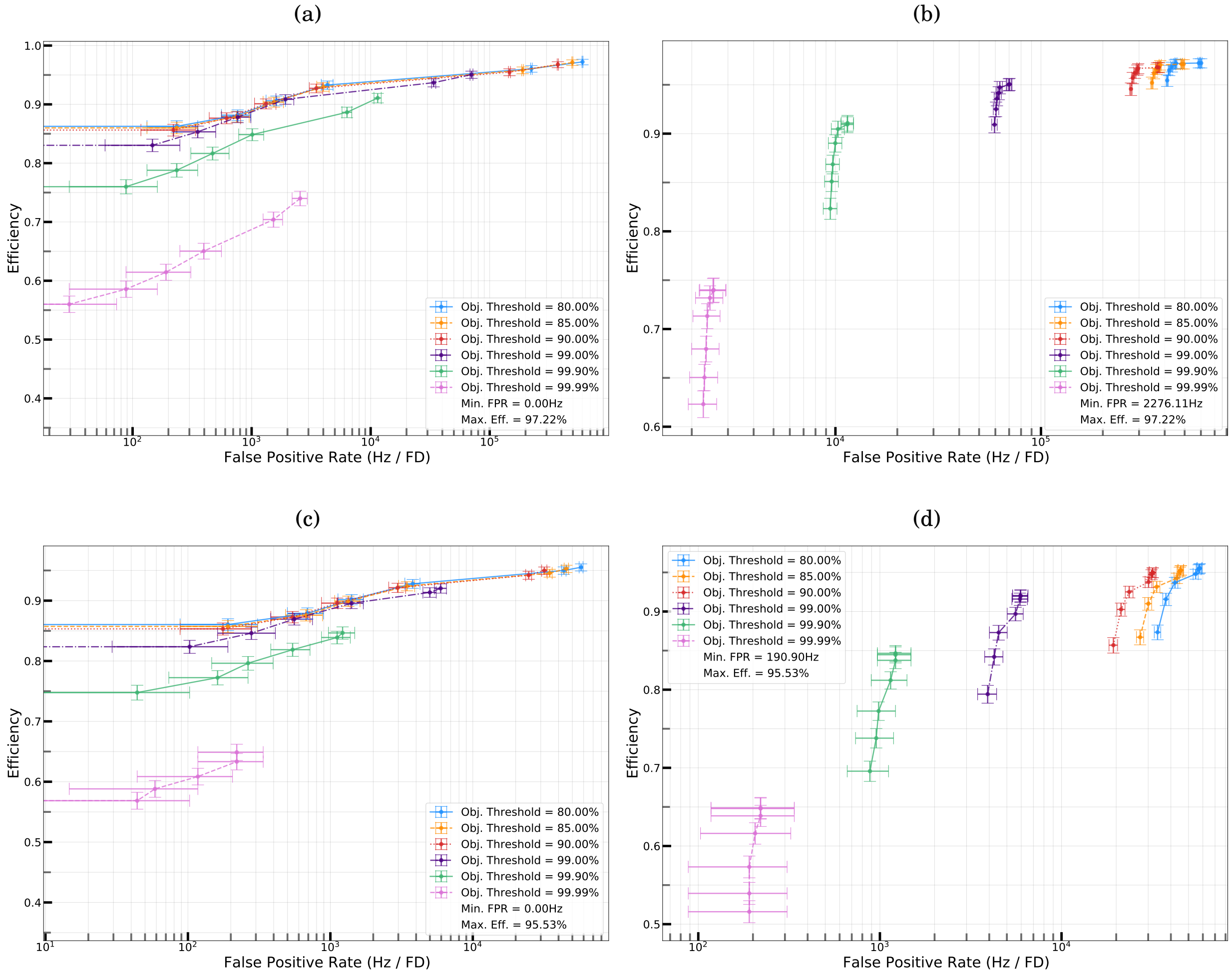
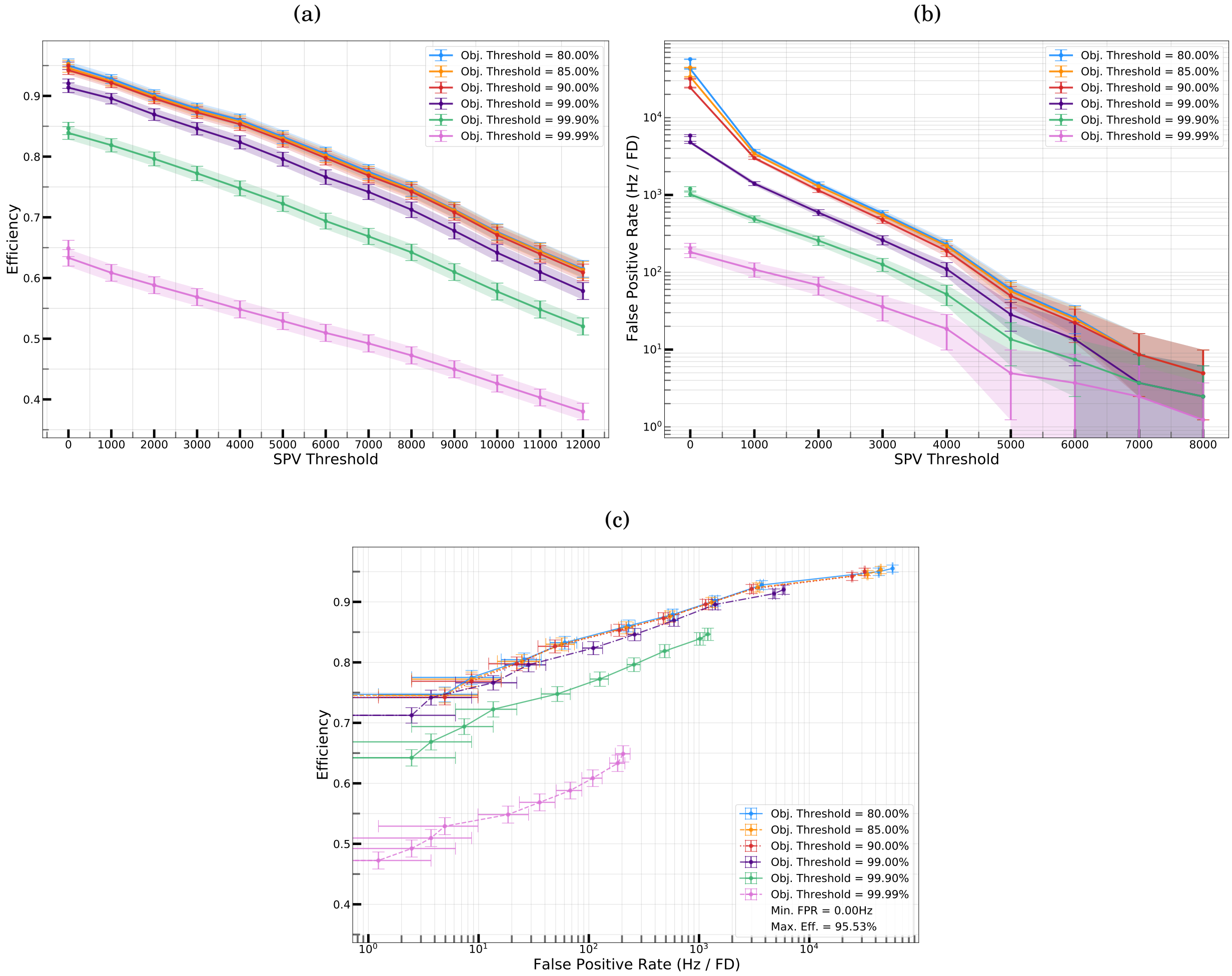


FIGURE 7.10. ϵ as a function of FPR comparing the use of predicted boxes from all scales and only scales 1 and 2 with the pilot study dataset. The error bars show 95% confidence regions. (a): Using $T_{NMS} = 0.75$ for predicted boxes from all scales. (b): Predicted boxes from all scales without the application of a T_{SPV}. (c): Using $T_{NMS} = 0.75$ for predicted boxes from scales 1 and 2. (d): Predicted boxes from scales 1 and 2 without the application of a T_{SPV}.

scales 1 and 2 was also investigated using the predicted boxes from the pilot study. The results are presented in figures 7.10(c) and 7.10(d) which use $T_{NMS} = 0.75$ and no T_{SPV} threshold respectively. These were compared to the respective figures, 7.10(a) and 7.10(b) which were found using predicted boxes from all scales of the network. It was found that when setting $T_{NMS} = 0.75$, discarding predicted boxes from scale 3 had little impact on the FPR for larger T_{SPV} thresholds. This can be seen in table 7.3, where $\epsilon_{1,2}$ values are within 1% of their ϵ_{123} counterparts. However, for $T_{SPV} = 1$ or no T_{SPV} threshold the FPR was reduced by an order of magnitude when only using predicted boxes from scales 1 and 2. This was also observed by comparing figure 7.10(d) to figure 7.10(b) where the reduction in FPR by an order of magnitude through the use of only


(a)

(b)

(c)

FIGURE 7.11. ϵ, FPR and ϵ as a function of FPR curves for full background dataset using $T_{NMS} = 0.75$ and scales 1 and 2. The error bars show 95% confidence limits. (a): ϵ for $T_{NMS} = 0.75$ for predictions from scales 1 and 2. (b): FPR for $T_{NMS} = 0.75$ for predictions from scales 1 and 2 using the large background dataset. (c): ϵ as a function of FPR for $T_{NMS} = 0.75$ for predictions from scales 1 and 2 using the large background dataset.

scales 1 and 2 was observed when no T_{SPV} was applied for all T_{NMS} thresholds.

7.5.2.1 Summary of Pilot Study

In summary, introducing a SPV threshold was found to be highly effective at reducing the FPR. The application of $T_{SPV} = 1$ was able halve the FPR, whilst $T_{SPV} = 5000$ discarded all the boxes predicted on the background-only dataset. The T_{NMS} was found to have comparatively little effect on the FPR but reducing T_{NMS} was found to lower the efficiency. Application of T_{obj} was able to reduce the FPR by orders of magnitude but reducing it by more than 3 orders of magnitude lowered the efficiency below 0.8. Dropping all predicted boxes from scale 3 was found to lower the efficiency by less than 1% whilst reducing the FPR by up to an order of magnitude.

Performance studies using the full 36,000 image background-only dataset allow a more precise determination of the FPR. The set of thresholds used to explore the performance of the network on this dataset were chosen based on the pilot study. The T_{SPV} had by far the largest impact on the FPR whilst preserving most of the efficiency, whilst smaller T_{NMS} and T_{obj} values had a poorer trade-off in performance. Therefore, the set of 16 combinations of thresholds for which to study the performance of the full background-only dataset around was chosen to be centred on a high T_{SPV} and loose T_{NMS} and T_{obj}. Each of these thresholds was then scanned individually to determine to whether other combinations were able to improve the performance. As a result of the efficiency of the network being so similar after dropping predicted boxes from scale 3, it was decided to discard those from scale 3 and only use those from scales 1 and 2.

7.5.3 Performance of YOLOv3 Network Using Full Background Dataset

The efficiency on the evaluation dataset and the FPR from the background-only dataset as a function of SPV are shown in figure 7.11 using $T_{NMS} = 0.75$. This background dataset of 36000 images had a precision of 1.23 Hz/FD module for each fp found. No fps were found with SPV values greater than 9000.

As can be seen in figures 7.11(a) and 7.11(b), the SPV only reduced the efficiency by a maximum of 25% when $T_{SPV} = 9000$ was applied but reduced the FPRs from 6.0×10^4Hz/FD module to <1.23Hz/FD module.

The ϵ as a function of FPR curve for the full background-only dataset is shown in figure 7.11(c) for $T_{NMS} = 0.75$ and different T_{SPV} thresholds. It was clear that simply using high T_{SPV} thresholds whilst keeping T_{NMS} and T_{obj} thresholds very loose remained the most effective way to preserve a high efficiency whilst minimising the FPR. The 16 different combinations of T_{obj}, T_{SPV} and T_{NMS} were chosen to be centred around $T_{obj} = 0.8$, $T_{SPV} = 9000$ and $T_{NMS} = 0.75$. This combination used loose T_{obj} and T_{NMS} and the lowest T_{SPV} which resulted in no fps. It was found that this combination resulted in the optimal efficiency of $\epsilon_{12} = 0.713$. The efficiency, ϵ, and FPR, μ_{FPR}, of the other combinations are reported in the fourth and fifth columns of table 7.4.

7.6 Impact of YOLO-based Supernova Neutrino Identification on Supernova Burst Trigger Performance

In the previous section only the efficiency of predicting at least 1 box which overlapped with the truth box was studied. Whilst the SNB trigger relies on the YOLOv3 network to be able to make predictions with a high efficiency, it also benefits from retaining multiple predicted boxes per neutrino interaction. This is because the SNB trigger simply counts the number of predicted boxes within a period of time and triggers when a threshold has been reached. The baseline

SNB trigger studied in section 5.4 counted the number of TCs instead of the number of predicted boxes.

The signal required by the toy MC to determine the SNB trigger efficiency, ϵ_{SNB} was the energy spectrum of $\hat{N}_E$, the average number of predicted boxes per neutrino interaction. This is known as the 'predicted boxes signal'.

The SNB trigger efficiency for a FPR of less than 1 per month was studied for the 16 permutations of T_{SPV}, T_{NMS} and T_{obj} set out in table 7.4. For each permutation the μ_{FPR} was used to determine the lowest threshold on the number of predicted boxes which led to a FPR of less than 1 per month. This threshold was then used to determine the SNB trigger efficiency, ϵ_{SNB} for that permutation. A number of these permutations found no fps from the 36,000 background images dataset. In these cases a lower limit on ϵ_{SNB} was determined by using the precision of the dataset, $\mu_{FPR} = 1.23\,\text{Hz/FD}$, as an upper limit on μ_{FPR} (this is the μ_{FPR} corresponding to 1 fp).

In this section the derivation of the energy spectrum of the average number of predicted boxes is explained. This will be referred to as the 'signal'. Following this, the computed efficiencies of the SNB trigger for each of the threshold permutations is reported.

Table 7.4: Efficiencies, (ϵ) and false positive rates (μ_{FPR}) for the T_{obj}, T_{NMS} and T_{SPV} working points selected for evaluation of the performance a SNB trigger using YOLOv3-derived clusters. The threshold permutations for which no fps were found are denoted with an asterisk. The threshold on the number of predicted boxes to achieve <1/month FPR for a SNB, T_{SNB} is also reported as well as the SNB trigger efficiency for SNB at the far edge of the galaxy and in the LMC. The reported errors are 95% confidence values.

T_{obj}	T_{NMS}	T_{SPV}	ϵ (%)	μ_{FPR} (Hz/FD)	T_{SNB}	30kpc ϵ_{SNB} (%)	50kpc ϵ_{SNB} (%)
Rules-based				$\leq 0.367^*$	15	$35.9^{+0.9}_{-1.0}$	$1.4^{+0.2}_{-0.2}$
0.8	0.75	5000	$83.3^{+1.1}_{-1.1}$	$60.49^{+17.28}_{-16.05}$	712	$80.7^{+0.7}_{-0.8}$	$3.6^{+0.3}_{-0.3}$
0.8	0.75	6000	$80.4^{+1.1}_{-1.1}$	$25.93^{+11.11}_{-9.88.}$	333	$96.9^{+0.3}_{-0.3}$	$12.7^{+0.7}_{-0.7}$
0.8	0.75	7000	$77.5^{+1.2}_{-1.2}$	$8.64^{+7.41}_{-6.17}$	132	$99.9^{+0.1}_{-0.1}$	$43.3^{+1.0}_{-1.0}$
0.8	0.75	8000	$74.7^{+1.2}_{-1.2}$	$4.94^{+4.94.}_{-3.70}$	83	$100^{+0.0}_{-0.1}$	$64.6^{+0.9}_{-0.9}$
0.8	0.75	9000	$71.3^{+1.3}_{-1.3}$	$\leq 1.23^*$	31	$100^{+0.0}_{-0.0}$	$92.5^{+0.5}_{-0.5}$
0.85	0.75	9000	$71.2^{+1.3}_{-1.3}$	$\leq 1.23^*$	31	$100^{+0.0}_{-0.0}$	$91.0^{+0.5}_{-0.6}$
0.9	0.75	9000	$70.8^{+1.3}_{-1.3}$	$\leq 1.23^*$	31	$100^{+0.0}_{-0.0}$	$88.9^{+0.6}_{-0.6}$
0.99	0.75	9000	$67.8^{+1.3}_{-1.3}$	$\leq 1.23^*$	31	$100^{+0.0}_{-0.1}$	$66.9^{+1.1}_{-1.0}$
0.999	0.75	9000	$61.0^{+1.4}_{-1.4}$	$\leq 1.23^*$	31	$98.7^{+0.2}_{-0.2}$	$28.6^{+1.1}_{-0.9}$
0.9999	0.75	9000	$45.0^{+1.4}_{-1.4}$	$\leq 1.23^*$	31	$58.6^{+0.9}_{-0.8}$	$2.1^{+0.3}_{-0.3}$
0.8	0.66	9000	$71.2^{+1.3}_{-1.2}$	$\leq 1.23^*$	31	$100^{+0.0}_{-0.0}$	$92.0^{+0.4}_{-0.6}$
0.8	0.5	9000	$70.8^{+1.3}_{-1.3}$	$\leq 1.23^*$	31	$100^{+0.0}_{-0.0}$	$90.2^{+0.5}_{-0.4}$
0.8	0.4	9000	$70.0^{+1.3}_{-1.3}$	$\leq 1.23^*$	31	$100^{+0.0}_{-0.0}$	$81.0^{+0.7}_{-0.8}$
0.8	0.3	9000	$68.0^{+1.3}_{-1.3}$	$\leq 1.23^*$	31	$99.9^{+0.0}_{-0.1}$	$57.1^{+0.9}_{-0.9}$
0.8	0.2	9000	$65.4^{+1.3}_{-1.4}$	$\leq 1.23^*$	31	$99.2^{+0.2}_{-0.2}$	$32.0^{+0.9}_{-1.0}$
0.8	0.1	9000	$61.0^{+1.4}_{-1.4}$	$\leq 1.23^*$	31	$94.3^{+0.5}_{-0.5}$	$13.6^{+0.7}_{-0.7}$

7.6.1 Predicted Boxes Signal

To determine the average number of predicted boxes per neutrino interaction, $\hat{N}_E$, as a function of neutrino energy, E_ν, the predicted boxes needed to be associated with a E_ν value. The efficiency for each threshold permutation depended only on whether a single tp was assigned to each truth box. A maximum of 1 predicted box was assigned to each truth box.

Whilst it is possible to assign an E_ν to a tp from the truth box it was matched to, there may remain predicted boxes which were assigned as fp in this classification scheme but result from the presence of a neutrino interaction. In addition, a particular neutrino interaction may lead to truth boxes on multiple images.

There were up to 3 neutrino interactions per image in the evaluation dataset. In order to find $\hat{N}_E$ for each energy bin, a subset of images from the evaluation dataset was selected. To allow each predicted box to be associated with a particular E_ν, all neutrinos which had truth boxes on the same image as another neutrino were discarded from the dataset. This allowed the remaining images to possess only a single truth box and therefore each could be associated with a particular neutrino and E_ν. The evaluation dataset contained 3032 images, with 2726 unique neutrino interactions. After discarding any neutrinos which had truth boxes on the same image as another neutrino, 1272 neutrino interactions and 1351 images remained which were used to derive $\hat{N}_E$ in each energy bin.

The total number of predicted boxes which survived the T_{obj}, T_{SPV} and application of NMS in each 'single-neutrino image' may have included predicted boxes which resulted from background sources in addition to those which resulted from a neutrino interaction. The total number of images corresponding to each energy bin allowed the average number of predicted boxes due to background sources for that bin, $\hat{N}_{bkg}$, to be estimated from μ_{FPR} for each threshold permutation.

The E_ν values were binned and for the neutrinos in each E_ν bin the total number of fps and tps was summed. The number of APAs (images) with truth boxes, in each E_ν bin was used to determine $\hat{N}_{bkg}$ in that bin. The $\hat{N}_E$ was then determined via:

$$\hat{N}_E = \frac{\sum_{i=1}^{n}(N_i^{tp} + N_i^{fp}) - \hat{N}_{bkg}}{n}, \tag{7.7}$$

where n is the number of neutrinos in the energy bin and N_i^{tp} and N_i^{fp} are the number of tps and fps for neutrino i.

The excess of fps is shown for the use of thresholds $T_{obj} = 0.8$ and $T_{NMS} = 0.75$ in figures 7.12(a) and 7.12(b), where $T_{SPV} = 1$ and $T_{SPV} = 9000$ were applied. The respective background rates of $\mu_{FPR} = 4.27 \times 10^4$ Hz/FD and $\mu_{FPR} = 1.23$ Hz/FD module were found to lead to $\hat{N}_{bkg}$ values far below the N^{fp} in each E_ν bin, with this becoming increasingly proportionally more severe as μ_{FPR} decreased.

The $\hat{N}_E$ as a function of E_ν for these two configurations are shown in figures 7.12(c) and 7.12(d), where it can be observed that $\hat{N}_E$ was much greater than determined for the rules-based

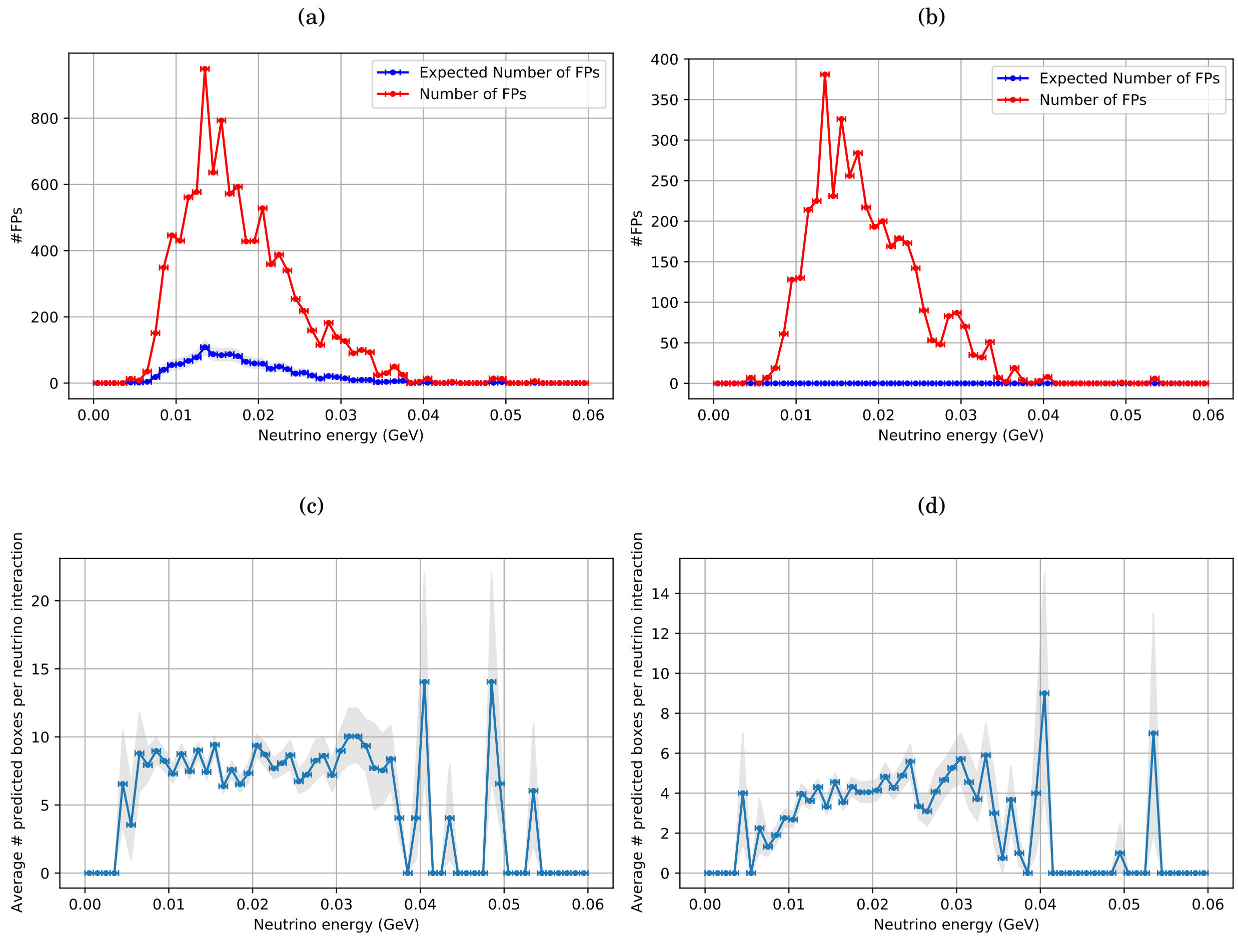


FIGURE 7.12. The number of fps found in each energy bin, from the evaluation dataset, N^{fp}, plotted with the expected number from the background-only dataset, $\hat{N}_{bkg}$. The resulting signal, $\hat{N}_E$, determined from equation 7.7, is also plotted. These plots all used thresholds of $T_{obj} = 0.8$ and $T_{NMS} = 0.75$ and used predictions from scales 1 and 2. The energy bin widths are denoted by the horizontal error bars and the grey shaded region quotes the 95% confidence limits on the y-axes. (a): N^{fp} and $\hat{N}_{bkg}$ for $T_{SPV} = 1$. (b): N^{fp} and $\hat{N}_{bkg}$ for $T_{SPV} = 9000$ (where $\mu_{FPR} = 1.23$ Hz/FD was used). (c): $\hat{N}_E$ for $T_{SPV} = 1$. (d): $\hat{N}_E$ for $T_{SPV} = 9000$ (where $\mu_{FPR} = 1.23$ Hz/FD was used).

algorithm (figure 5.8). The low statistics used to create these plots limited the strength of the conclusions which could be drawn from them but there were indications that increasing T_{SPV} resulted in lower $\hat{N}_E$ values for low E_ν bins when compared to the average $\hat{N}_E$ over higher E_ν bins.

To summarise the predicted boxes signal, increased multiplicity of predicted boxes in a single image could be interpreted as an increased probability of 1 or more neutrino interactions, particularly after the use of a high SPV threshold. As these predictions remained after NMS it was evident that the network did not learn to predict localization of a neutrino signal perfectly. However, the use of multiple predictions provides a tool to enhance the number of predicted boxes which may be counted in a SNB counting trigger despite the removal of the predictions with the highest IOU with each other.

7.6.2 SNB Trigger Performance

The $\hat{N}_E$ as a function of E_ν plots and μ_{FPR} values from table 7.4 were used as inputs to the SNB counting trigger, with the other parameters used being the same as stated in section 5.4.1. Two months of background was simulated and used to determine T_{SNB}, the threshold on the number of predicted boxes at which a SNB trigger with a FPR of less than once per month would be issued.

The impact of varying the T_{obj}, T_{NMS} and T_{SPV} thresholds on the efficiency of the SNB trigger when it was limited to a maximum FPR of once per month was studied. The results for the SNB trigger when using rules-based clustering were compared to determine whether the SNB trigger performance was improved through the use of YOLOv3-derived clusters (ie: predicted boxes). The errors reported are statistical and for the 95% confidence level throughout this section.

The impact of changing T_{obj}, T_{NMS} and T_{SPV} is shown in figures 7.13(a), 7.13(b) and 7.13(c) respectively. It was apparent that the SNB counting trigger had a greater ϵ_{SNB} when using ML-based clustering than rules-based clustering for all of the threshold permutations plotted.

The LMC is located at 49.6 kpc, the solar system is 8.2 kpc from the centre of the Milky Way galaxy and the distribution of expected supernovae as a function of galactocentric distance falls to near zero beyond 18 kpc as stated in the introduction to chapter 5. The efficiency of the rules-based SNB trigger was 100% to distances up to 15 kpc, dropped to $97.7^{+0.3}_{-0.3}$% at 20 kpc and to $71.3^{+0.9}_{-0.9}$% at 25 kpc, around the distance that the SNB distribution is expected to be near-zero on the opposite side of the galaxy. For the vast majority of the threshold permutations for the SNB trigger which used ML-derived clusters, ϵ_{SNB} at 25 kpc is 100%. The efficiencies at 30 kpc and 50 kpc are reported in table 7.4 illustrating which permutations have the worst and best ϵ_{SNB} up to the LMC. Beyond the LMC the nearest galaxy, M31 (Andromeda), is 744±33 kpc away [165], which has $\epsilon_{SNB} = 0$ for all the permutations due to the negligible neutrino flux.

The poorest ϵ_{SNB} from the ML-derived clusters was found to result from the use of a very high T_{obj} or the use of a low T_{SPV}. When scanning ϵ_{SNB} for different values of T_{obj}, the upper limit of μ_{FPR} =1.23 Hz/FD was used for each, leading to the same T_{SNB} = 31 value being required to issue a trigger command. The same was true when scanning different T_{NMS} values. These threshold permutations therefore led to lower limits on ϵ_{SNB} whose differences for each permutation depended on the predicted boxes signal and statistical uncertainties only. As can be observed in table 7.4, the statistical uncertainties are small, usually less than 1%. The ϵ_{SNB} at different SNB distances are plotted for a scan over T_{obj} values in figure 7.13(a), where ϵ_{SNB} was found

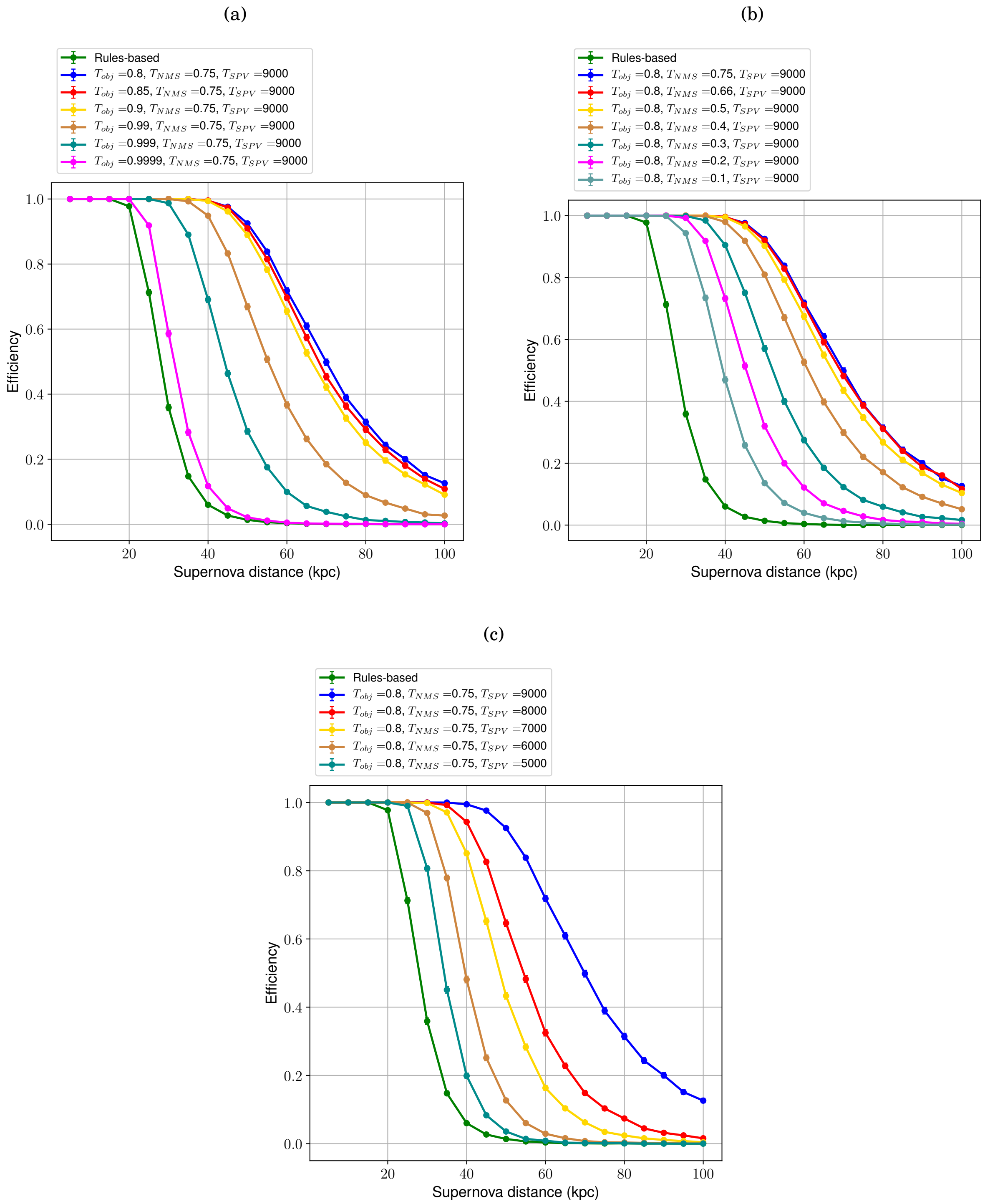


FIGURE 7.13. Efficiency, ϵ_{SNB}, for a FPR of <1/month as a function of SNB distance using predicted boxes from YOLOv3. Predicted boxes from scale 3 were discarded. In each case, the performance of the rules-based SNB trigger is plotted in green and the performance of $T_{obj} = 0.8$, $T_{NMS} = 0.75$ and $T_{SPV} = 9000$ is plotted in blue. 95% confidence error bars are plotted but appear negligible. (a): ϵ_{SNB} scanning across T_{obj} for $T_{NMS} = 0.75$ and $T_{SPV} = 9000$. (b): ϵ_{SNB} scanning across T_{NMS} for $T_{obj} = 0.8$ and $T_{SPV} = 9000$. (c): ϵ_{SNB} scanning across T_{SPV} for $T_{obj} = 0.8$ and $T_{NMS} = 0.75$.

to be similar for $T_{obj} = 0.8$ to $T_{obj} = 0.9$ but significantly declines for $T_{obj} \geq 0.99$. Figure 7.13(b) similarly plots the ϵ_{SNB} for a scan over the T_{NMS} values. In this case, ϵ_{SNB} is similar for $T_{NMS} \leq 0.5$ but declines as T_{NMS} decreases. These figures suggest that the values of $T_{obj} = 0.8$ and $T_{NMS} = 0.75$ provide reasonable thresholds as it is clear that loosening them further would result in diminishing returns.

When scanning different T_{SPV} values, the μ_{FPR} also varied. Decreasing T_{SPV} from 9000 to 8000 increased μ_{FPR} by at least a factor of 4. This resulted in a drop of ϵ_{SNB} from a lower limit of $92.5^{+0.5}_{-0.5}\%$ for $T_{SPV} = 9000$ to $64.6^{+0.9}_{-0.9}\%$ at the LMC. The ϵ_{SNB} for a range of SNB distances over a scan of T_{SPV} values is shown in figure 7.13(c). The ϵ_{SNB} was found to be clearly optimal for $T_{SPV} = 9000$, which corresponds to the lowest μ_{FPR}. This suggests that ϵ_{SNB} is best optimised by reducing μ_{FPR}.

When comparing ϵ_{SNB} between rules-based clustering and ML-derived clusters it should be remembered that the ϵ_{SNB} determined for the rules-based clustering was also a lower limit. Although it was derived from the same SN neutrino dataset, the rules-based algorithm did not require a large part of the dataset to be set aside for training purposes. This led to an upper limit of $\mu_{FPR} = 0.367$Hz as explained in section 5.4. The cluster signal energy spectrum was significantly lower for the rules-based clustering, as can be observed by comparing figures 5.8 and 7.12(d). The increased predicted boxes signal per neutrino interaction appeared to allow the YOLOv3-based SNB trigger to reach greater ϵ_{SNB} than the rules-based algorithm despite $T_{SNB} = 31$ being required for the YOLOv3-based SNB trigger and only $T_{SNB} = 15$ being required for the rules-based SNB trigger.

In summary, upper limits on μ_{FPR} determined from the precision of the rules-based and evaluation datasets were used to determine lower limits on ϵ_{SNB}. It was found that whilst for the rules-based SNB trigger the efficiency at the far side of the galaxy (25kpc), $\epsilon_{SNB} \geq 0.71$ and $\epsilon_{SNB} \geq 0.01$ at the LMC. However, for the YOLOv3-based SNB trigger it was found that $\epsilon_{SNB} = 1$ at the far side of the galaxy and $\epsilon_{SNB} \geq 0.92$ at the LMC.

7.7 Open Ended Future Work

The work presented in this chapter has shown that a YOLO-based approach to SNB triggering has the potential to perform comparably or better than the rules-based algorithm. However, there remains a large amount of work to be done in order show that a YOLO-based approach can be implemented successfully and keep up with processing in the DUNE DAQ system.

In this section suggested next steps for the study of the use of a YOLO network are briefly outlined, starting with further studies which could be done on the current implementation without changing the network architecture, followed by possible approaches to simplify the architecture (resulting in faster throughput) and limit the number of images which the network needs to process (relaxing throughput constraints). Other techniques which may be used to speed up the computational processing or improve the training of the network are also discussed.

7.7.1 Further Studies Using Existing Architecture

In section 7.4 it was shown that L_{obj} did not minimise over multiple epochs of training, potentially due to the weighting applied. It would clearly be advantageous to study the optimisation of w_{obj} to better understand this.

The existing training could also be improved through weight regularisation, which can reduce overfitting. This has the effect of smoothing the response of the network by making it more linear. Depending on the method of weight regularisation, the weights can either be minimised or sparsified.

Another area which may allow for optimisation is in the creation and downsampling of the images. The images are currently composed of collection channels from both sides of an APA. It would make sense to train the network using images created from the channels on only one side of an APA and re-evaluate the performance. Further to this, it may be useful to generate and downsample images to preserve a 1:1 aspect ratio to determine whether this improves the generalisation of the network. In addition, several different choices for the size of the initial image could be made along with suitable downsampling factors along each axis to preserve a 1:1 aspect ratio to determine if the efficiency of the YOLOv3 network can be improved.

Each selection of initial image size would require the inference speed of the network to be able to keep up with a different rate. For the images studied in this chapter, the scope was APA for 1 drift time (2.246 ms). If all images produced by the APA were processed on a single GPU, the inference rate would be required to be 456 FPS rather than the inference rate of around 34 FPS (from table 7.1) on a GPU. As the majority of the performance increase of the network appeared to occur through the application of SPV thresholds, it would be good to study if SPV over an entire downsampled image could be used discard the majority of background-only images whilst preserving the efficiency of the network.

There is also the option to introduce other interaction types such as solar neutrinos or the high energy beam neutrinos. The network currently is only trained to differentiate the low energy SN neutrino interactions from backgrounds there is no reason why it could not be extended to other sources and used to classify signals as high or low energy.

7.7.2 Network Architecture Optimisation

Regardless of studies to reduce the required FPS of YOLOv3, the inference and a subsequent optimised application of thresholding and NMS remain to be benchmarked.

Discarding predictions from scale 3 of the network was found to be advantageous. This scale makes 10800 out of the total 14,175 predictions and a significant part of the network architecture after the Darknet-53 feature extractor. It would be useful to determine the performance and inference speed of the network after removing scale 3 from the architecture. Additionally, the

Darknet-53 feature extractor used in YOLOv3 is significantly more complex than the Darknet-19 used in YOLOv2. It would be useful to determine the performance of the network using a simpler feature extractor to determine whether it actually helps for these very sparse images.

Once an architecture has been chosen by balancing the efficiency against the inference speed, the network architecture may be further simplified by pruning. This is the removal of any network weights which are close to zero and may be particularly effective when using L1 regularisation in the loss.

Another potential technique for speeding up inference for sparse image data is the use of submanifold sparse convolutional networks (SSCNs) [166], which have been used to achieve efficiencies of >93% when classifying Michel electrons in LArTPC volumes [167]. This work showed that for images constructed from raw ADC information the memory and inference times were reduced by factors of 93 and 3.1 respectively through sparse convolutions. As the images constructed from TP information are even more sparse, the benefits are likely to be even greater.

7.7.3 Network Implementation

As discussed in section 7.1.1, potential locations for this network may be in the HLF, a GPU added to DAQ RUs or within any spare FPGA resources in the DAQ RUs. The best approach depends on investigations suggested in section 7.7.2, as the RAM constraints on FPGAs require networks to be small or heavily optimised and use limited floating point precision. Simple approaches such as applying a SPV or similar threshold on images in order to decide whether to pass them to the network should be investigated first as they may decrease any required speed-up significantly.

7.7.4 Additional Network Arguments

Whilst the current network has 3 color channels, all were filled with the same pixel values in these studies. The induction wires are wrapped on an APA to cross each collection wire once and provide information which helps in reconstructing the location of TPs in the 3D TPC volume. Similarly to previous LArTPC studies such as [146] and [150], induction channel information and TPG in the UD could be studied and provided to the network to provide more context for the network to improve its localisation predictions.

7.8 Conclusion

In conclusion, it has been shown that the use of a YOLOv3 network to identify neutrino interactions was able to achieve SNB trigger efficiencies of 100% at the far side of the galaxy and 92% at the LMC using thresholds of $T_{obj} = 0.8$, $T_{SPV} = 9000$ and $T_{NMS} = 0.75$. Due to limits on statistics, these were taken to be lower estimates of the efficiency. These results improved upon the SNB trigger efficiencies determined for the rules-based SNB trigger of 71% at the far side of the galaxy and 1% at the LMC although these were also statistics-limited lower bounds for the efficiency.

Images were prepared on a per-APA basis over a single drift window in the detector, providing scope for this algorithm to be applied within a DAQ RU or the HLF. The hyperparameters were not significantly explored, except for the anchor boxes which were derived from the truth boxes in the dataset. The network appeared to undergo training successfully, minimising the overall loss until there were indications of a small amount of overfitting. Whilst the average IOU of predicted boxes with truth associations increased over the epochs of training, the average objectness score of these boxes did not but remained high. An explanation for this was not found and is a candidate for future work.

The impact of different choices of T_{obj}, T_{SPV} and T_{NMS} thresholds on the efficiency of neutrino interaction signals and false positive rates was characterised with a pilot study, finding that the T_{NMS} had the least ability to decrease the false positive rate, whilst T_{SPV} was the most significant threshold for this, decreasing it by orders of magnitude with moderate impact on the efficiency.

The vast majority of predictions made by the network were from scale 3. For the pilot study dataset it was determined that discarding the predictions from scale 3 had a negligible impact on the efficiency, whilst being able to decrease the false positive rate by another order of magnitude. A number of threshold permutations were selected based around $T_{obj} = 0.8$, $T_{SPV} = 9000$ and $T_{NMS} = 0.75$, which was determined to have the highest efficiency for the lowest false positive rate. The effect of varying each of the different thresholds on the efficiency of the SNB trigger was investigated and the identified set of thresholds was indeed found to have the best performance. The improved performance over the rules-based algorithm was theorised to likely have been a result of the ability of the YOLOv3 approach to propose and retain multiple predicted boxes for the signal from a given neutrino interaction.

These results showed that a YOLOv3-based SNB trigger using TPs generated from FPGA-based TPG could be used to trigger on SNB events within the galaxy with 100% efficiency, satisfying the requirement that any SNB with 60 neutrino interactions or more will be detected with >95% efficiency.

Chapter 8

Conclusions

Neutrino oscillations are a crucial area of study for physics beyond the standard model. They may provide evidence for solutions of the baryon asymmetry problem but neutrino mixing parameters have not yet been precisely determined. The neutrino signal from a core collapse SNB in the Milky Way should be observable using the DUNE detector and is the only signal capable of studying the internal mechanisms of this process.

DUNE will have constraints on the amount of data it can record and store to disk. In order to select which data to store for further processing it will require a set of triggering algorithms, with each corresponding to particular phenomena. These triggering algorithms are required to reduce the data collection rate from $1.2\,\mathrm{TB\,s^{-1}}$ per FD module to a $30\,\mathrm{PB}$ per year set of useful physics events. The approach which will be used in the DUNE SP FD module(s) will utilise FPGA resources to process the input data into TPs which will be combined in the subsequent stages of triggers and used to form trigger decisions.

The first goal of the work presented in this thesis was to demonstrate the validation of the FPGA-based TPG in the front-end DAQ and the second goal was to determine its performance when utilised as the first stage of a SNB trigger algorithm. The third goal was to explore whether a machine learning approach could be utilised to improve the SNB trigger performance.

To faciliate the development and testing of firmware which implements TPG in FPGA logic, validation tools with a bitwise precision were written. The validation studies presented in this thesis culminated in looking for remaining bugs in the TPs generated when the FPGA-based TPG was implemented in ProtoDUNE, allowing LArTPC data from an active detector to be explored. Comparing the TPs generated by the firmware implemented in ProtoDUNE and the TPs generated from the bitwise emulation, it was found that 51.8% were a perfect match. A small number of characteristic bugs were identified and feedback was provided to the TPG firmware developers who were able to reproduce most of them and implement fixes.

The technical requirements for the SNB trigger implemented in the DUNE FD were that it should be able to trigger on a SNB which results in at least 60 interactions with a visible energy of >10 MeV. Sixty neutrino interactions are expected to occur in a single SP FD module at a distance of 20 kpc in the case of a $11.2m_{\odot}$ progenitor star. Using detected ionisation in the TPC module a baseline SNB trigger algorithm was determined to have an efficiency of $97.7^{+0.2}_{-0.3}\%$ at 20 kpc. This was achieved without restricting the true neutrino energies to be larger than 10 MeV, and achieved the technical requirements set out for the DUNE SNB trigger. The thresholds used to generate TPs in this algorithm were set so that the data rate of TPs generated per year was limited to 2 PB, fulfilling the requirement to save all TPs to disk.

Further studies have been completed to explore how the performance of a SNB trigger may be further improved through the use of a ML algorithm. A bounding box proposal network, YOLOv3, was used. It replaced the clustering stage of the baseline SNB trigger. Thresholds on objectness scores, summed pixel values and on the intersection-over-union during application of non-maximum suppression were applied to the predicted boxes during post-processing. This parameter space was explored and the optimal combination was found to result in a SNB trigger efficiency of 100% up to the far side of the Milky Way galaxy and remained at $92.5^{+0.5}_{-0.5}\%$ in the LMC.

There were some limitations to the studies performed and there are improvements which could be made. The SNB triggering studies presented used only a single core collapse SN model and could be repeated for a range of different SNB candidates to ensure that the trigger generalises well. Also, it is noted that the efficiencies gained from both the baseline and ML SNB trigger approaches are expected to be lower estimates. In the case of the baseline TPG no false positive TCs were found in the dataset of 30,000 neutrino interactions due to limitations on the statistics of these studies. The studies could be repeated with larger statistics to more precisely determine the efficiency and false positive rates.

In conclusion, this work has enabled the improvement of the FPGA-based TPG; has demonstrated a baseline SNB trigger which meets the DUNE requirements and has successfully explored the application of a ML approach to further optimise the SNB trigger performance.

APPENDIX A

APPENDIX A

This appendix includes the algorithms used in this work which are not otherwise fully described.

A.1 Clustering Algorithm

To identify local regions of charge deposition which span one or more wires, the TPs are clustered together into TCs. TPs are ordered by wire number (adjacent wires across the APA) and the TPs which are within a wire tolerance, t_W, are formed into a protocluster. The TPs in this protocluster are ordered by their H_S. The TPs in the protocluster which are within a tick tolerance, t_T, are grouped together into a TC. If any thresholds on TPs properties were set, TPs which fail them are discarded from the TC. If any thresholds on TCs properties are set, TCs which fail them are discarded. The full algorithm is described in algorithm 1.

A.2 Noise Simulation

This algorithm was implemented in LArSoft [119] by David Adams and was used as the standard model to generate a random noise spectrum which was sampled from to add noise the signals on APA wires.

The noise was generated using an exponential distribution in the frequency domain, with the value in each frequency bin of the noise distribution, N_i being determined by equation A.1:

$$N_i = Ae^{-\frac{f_i}{\lambda_d}} \tag{A.1}$$

with an amplitude, A, of 3.16 ADC counts and a noise damping parameter $\lambda_d = 2$ kHz^{-1}. f_i is the value of the frequency in bin i. A low frequency cut-off value, f_{low} was set to be 7.5 Hz. This was

Algorithm 1 Clustering algorithm to form TCs

```
Require: TPs collected in the detector module over a period of time
Require: A wire tolerance, $t_W$ , to group TPs over
Require: A tick tolerance, $t_T$ , to group TPs in protoclusters over
Require: Thresholds on TP properties
Require: Thresholds on TC properties
Ensure: TPs are channel ordered
  for TP in channel ordered list do
    if adjacent TP(s) within $t_W$ of each other then Create protocluster from TPs
Ensure: TPs in protocluster are time-ordered
      for TP in time ordered list do
        if adjacent TP(s) within $t_T$ of each other then
          Group TPs into a TC
          if Thresholds on TP properties applied then
            Discard TPs which do not pass threshold
          end if
          if Thresholds on TC properties applied then
            Discard TC if it does not pass thresholds
          end if
        end if
      end for
      Remove TPs within $t_W$ of each other
    end if
  end for
```

used to filter out low frequency noise. 10% of the noise in the bin was also randomized. Both of these steps were applied in each frequency bin as follows in equation A.2:

$$N_i = N_i \times F \times (0.9 + 0.2 \times rnd), \tag{A.2}$$

where rnd is a random number between 0 and 1 drawn from a uniform distribution. F is a term to filter out low frequency noise given by:

$$F = \frac{1}{1 + exp\Big(-2\big(\frac{i-f_{low}}{b_w}\big)\Big)} \tag{A.3}$$

where b_w, the frequency bin width, is given by $1/(n_{tick} \times r_s)$, with n_{tick} being the number of ticks used in the subsequent fast Fourier transform (FFT) and r_s being the sampling rate of the detector. Each bin of the noise distribution in the frequency spectrum was then given a random phase before the time spectrum was determined through the application of an inverse FFT.

A.3 Minibatch SGD

For an ANN with inputs $\boldsymbol{x}$, weights $\boldsymbol{w}$, targets $\boldsymbol{t}$, a minibatch can be used to increment the weights through the application of the SGD:

Algorithm 2 Minibatch stochastic gradient descent

```
Require: Learning rate $\epsilon$
Require: Initial weights $\boldsymbol{w}$
Require: Stopping criterion
  $k \leftarrow 1$
  while stopping criterion is not met do
      Randomly sample minibatch of $m$ input-target pairs
      Compute the gradient estimate: $\hat{\boldsymbol{g}} \leftarrow \frac{1}{m}\nabla_{\boldsymbol{w}} \sum_i L(f(\boldsymbol{x}, \boldsymbol{w}), \boldsymbol{t})$
      Update the weights: $\boldsymbol{w} \leftarrow \boldsymbol{w} - \epsilon\hat{\boldsymbol{g}}$
      $k \leftarrow k+1$
  end while
```

Any stopping condition can be defined; common examples include a number of minibatches, a convergence of the loss or a training time.

A.4 Adam Optimiser

Adam is a widely used variant of SGD which allows a network to maintain a momentum in a given direction in the loss parameter space via a velocity term, v. The number of previous iterations which are taken into account depends on exponential decay rates, ρ_1 and ρ_2.

Algorithm 3 Adam optimiser

```
Require: Step size $\epsilon$
Require: Exponential decay rates for momentum estimate $\rho_1$ and $\rho_2$ which are in the range
  [0,1)
Require: A small constant $\delta$, used for numerical stability
Require: Initial parameters $\boldsymbol{w}$
  Initialise the moment variables $\boldsymbol{s} = 0$ and $\boldsymbol{r} = 0$
  Initialise a time step $t = 0$
  while stopping criterion is not met do
      Randomly sample minibatch of $m$ input-target pairs
      Compute the gradient estimate: $\hat{\boldsymbol{g}} \leftarrow \frac{1}{m}\nabla_{\boldsymbol{w}} \sum_i L(f(\boldsymbol{x}, \boldsymbol{w}), \boldsymbol{t})$
      $t \leftarrow t+1$
      Update the biased first moment estimate: $\boldsymbol{s} \leftarrow \rho_1 \boldsymbol{s} + (1-\rho_1)\boldsymbol{g}$
      Update the biased second moment estimate: $\boldsymbol{r} \leftarrow \rho_1 \boldsymbol{r} + (1-\rho_2)\boldsymbol{g} \odot \boldsymbol{g}$
      Correct for bias in the first moment: $\hat{\boldsymbol{s}} \leftarrow \frac{\boldsymbol{s}}{1-\rho_1^t}$
      Correct for bias in the second moment: $\hat{\boldsymbol{r}} \leftarrow \frac{\boldsymbol{r}}{1-\rho_2^t}$
      Determine the update to apply from: $\Delta\boldsymbol{w} = -\epsilon\frac{\hat{\boldsymbol{s}}}{\sqrt{\hat{\boldsymbol{r}}}+\delta}$
      Apply the update to the parameters: $\boldsymbol{w} \leftarrow \boldsymbol{w} + \Delta\boldsymbol{w}$
  end while
```

A.5 Non-maximal Suppression

For an array of bounding boxes, the non-maximal suppression algorithm can be used to remove boxes which overlap by more than a given threshold, T_{NMS} and preferentially retain those with the highest object scores.

Algorithm 4 NMS algorithm for discarding overlapping predicted boxes

```
Require: Array of predicted bounding boxes, $\boldsymbol{B}$ for an image, reverse ordered by objectness score
Require: An IOU threshold, $T_{NMS}$
  while There is 1 or more predicted boxes remaining in the array do
     Step 1: Select the first bounding box, $\boldsymbol{B}_{high}$, in the array, with the highest object score.
       Remove it from the array. Place it into an array of kept bounding boxes.
     Step 2: Compute IOU value of $\boldsymbol{B}_{high}$ with $\boldsymbol{B}_i$ in the array.
     if $IOU_i \geq T_{NMS}$ then
        Discard $\boldsymbol{B}_i$ from the array of bounding boxes
     end if
  end while
```

A.6 Assigning Predicted Boxes To Truth Boxes

Predicted boxes may be assigned to be tps, fps or fns. Each truth box may be assigned to only a single predicted box and vice versa. Algorithm 5 shows how truth boxes and predicted boxes are assigned to each other.

A.7 k-median Clustering For Anchor Boxes

Anchor boxes are derived through a k-median clustering algorithm. The number of clusters, N_c, which will be determined from the application of k-median clustering must be specified as an input to the algorithm.

As explained in section 7.3.1 and shown in figure 7.1, each 'cluster' derived through the k-median clustering algorithm corresponded an anchor box of a given pixel width and height. The k-median clustering algorithm for determining the optimal anchor boxes in full is set out in algorithm 6.

Algorithm 5 Assigning predicted boxes as tp or fp and truth boxes as fn

```
Require: Array of predicted bounding boxes $\boldsymbol{B}$ for an image, reverse ordered by objectness score
    after application of $T_{obj}$ and any other pre-NMS thresholds.
Require: Array of truth boxes $\hat{\boldsymbol{B}}$ for an image
Require: $T_{IOU}$, an IOU threshold required to label a $\boldsymbol{B}$ as a tp
  $i \leftarrow 0$
  for $\boldsymbol{B}_i$ do
      Compute IOU with all $\hat{\boldsymbol{B}}$ boxes for $\boldsymbol{B}_i$
      Arrange array of $\hat{\boldsymbol{B}}$ in reverse IOU order
      $j \leftarrow 0$
      for $\hat{\boldsymbol{B}}_j$ do
          if $\hat{\boldsymbol{B}}_j$ previously assigned to a $\boldsymbol{B}$ then
              $j \leftarrow j+1$
              Continue
          else
              if $\boldsymbol{B}_i$ has IOU > $T_{IOU}$ with $\hat{\boldsymbol{B}}_j$ then
                  Assign $\boldsymbol{B}_i$ and $\hat{\boldsymbol{B}}_j$ to each other
                  Assign $\boldsymbol{B}_i$ as a tp
                  Exit loop over $j$
              end if
          end if
          $j \leftarrow j+1$
      end for
      if $\boldsymbol{B}_i$ has IOU $\le T_{IOU}$ with all $\hat{\boldsymbol{B}}$ or all $\hat{\boldsymbol{B}}_j$ already assigned to a $\boldsymbol{B}$ then
          Label $\boldsymbol{B}_i$ as a fp
      end if
      $i \leftarrow i+1$
  end for
  if there are any remaining $\hat{\boldsymbol{B}}_j$ not assigned to a $\boldsymbol{B}$ then
      Label remaining $\hat{\boldsymbol{B}}_j$ as fn.
  end if
```

Algorithm 6 k-median clustering algorithm to determine optimal anchor boxes

```
Require: A list of truth boxes
Require: A number of anchor boxes, $N_c \leftarrow 9$ to determine
  Step 1: Initialise $N_c$ anchor boxes with widths ($w$) and heights ($h$) which are a random
  proportion of the image size
  Step 2: Determine the IOU, $I$ of each truth box with each anchor box
  Step 3: Assign each truth box to the anchor box which it has the greatest IOU with
  Step 4: Determine the median $w$ and $h$ values of the truth boxes assigned to each anchor box
  and assign these values to be the dimensions of the anchor boxes used for the next iteration
  Step 5: Repeat steps 2, 3 and 4 until no anchor box $w$ or $h$ changes after the application of
  steps 2,3 and 4
```

BIBLIOGRAPHY

[1] Y. Fukuda, T. Hayakawa, E. Ichihara, K. Inoue, et al.
Evidence for Oscillation of Atmospheric Neutrinos.
Phys. Rev. Lett., 81:1562–1567, Aug 1998.

[2] Q. R. Ahmad, R. C. Allen, T. C. Andersen, J. D. Anglin, et al.
Measurement of the Rate of $v_e + d \to p + p + e$-Interactions Produced by ^{8}B Solar Neutrinos at the Sudbury Neutrino Observatory.
Physical Review Letters, 87(7), Jul 2001.

[3] A. D. Sakharov.
Violation of CP Invariance, C asymmetry, and baryon asymmetry of the universe.
Pisma Zh. Eksp. Teor. Fiz., 5:32–35, 1967.

[4] L. N. Chang and C. Soo.
Standard model with gravity couplings.
Physical Review D, 53(10):5682–5691, May 1996.

[5] V. Trimble.
Existence and Nature of Dark Matter in the Universe.
Annual Review of Astronomy and Astrophysics, 25(1):425–472, 1987.

[6] P. J. E. Peebles and B. Ratra.
The cosmological constant and dark energy.
Reviews of Modern Physics, 75(2):559–606, Apr 2003.

[7] W. De Boer.
Grand unified theories and supersymmetry in particle physics and cosmology.
Progress in Particle and Nuclear Physics, 33:201–301, Jan 1994.

[8] C. S. Wu, E. Ambler, R. W. Hayward, D. D. Hoppes, and R. P. Hudson.
Experimental Test of Parity Conservation in Beta Decay.
Phys. Rev., 105:1413–1415, Feb 1957.

[9] J. H. Christenson, J. W. Cronin, V. L. Fitch, and R. Turlay.
Evidence for the 2π Decay of the K_2^0 Meson.

Phys. Rev. Lett., 13:138–140, Jul 1964.

[10] V. Fanti, A. Lai, D. Marras, L. Musa, et al.
A new measurement of direct CP violation in two pion decays of the neutral kaon.
Physics Letters B, 465(1):335–348, 1999.

[11] A. Alavi-Harati, I. F. Albuquerque, T. Alexopoulos, M. Arenton, et al.
Observation of Direct CP Violation in $K_S,L \to \pi\pi$ Decays.
Phys. Rev. Lett., 83:22–27, Jul 1999.

[12] K. Abe, K. Abe, R. Abe, I. Adachi, et al.
Observation of Large CP Violation in the Neutral B Meson System.
Phys. Rev. Lett., 87:091802, Aug 2001.

[13] R. Aaij et al.
Observation of CP Violation in Charm Decays.
Phys. Rev. Lett., 122:211803, May 2019.

[14] M. E. Shaposhnikov.
Structure of the High Temperature Gauge Ground State and Electroweak Production of the Baryon Asymmetry.
Nucl. Phys. B, 299:797–817, 1988.

[15] K. Abe, R. Akutsu, A. Ali, C. Alt, et al.
Constraint on the matter–antimatter symmetry-violating phase in neutrino oscillations.
Nature, 580(7803):339–344, Apr 2020.

[16] E. Fermi.
Versuch einer Theorie der Beta-Strahlen.
I. Z. Physik, 88:161–177, 1934.

[17] J. Chadwick.
The intensity distribution in the magnetic spectrum of beta particles from radium (B + C).
Verh. Phys. Gesell., 16:383–391, 1914.

[18] F. Reines and C. Cowan.
The Neutrino.
Nature, 178:446–449, 1956.

[19] G. Danby, J-M. Gaillard, K. Goulianos, L. M. Lederman, et al.
Observation of High-Energy Neutrino Reactions and the Existence of Two Kinds of Neutrinos.
Phys. Rev. Lett., 9:36–44, Jul 1962.

[20] K. Kodama, N. Ushida, C. Andreopoulos, N. Saoulidou, et al.
Observation of tau neutrino interactions.
Physics Letters B, 504(3):218–224, Apr 2001.

[21] The ALEPH Collaboration et al.
Precision electroweak measurements on the Z resonance.
Physics Reports, 427(5–6):257–454, May 2006.

[22] R.N. Mohapatra.
Seesaw mechanism and its implications.
Seesaw 25, Apr 2005.

[23] B. Pontecorvo.
Neutrino Experiments and the Problem of Conservation of Leptonic Charge.
Zh. Eksp. Teor. Fiz., 53:1717–1725, 1967.

[24] J. N. Bahcall, M. H. Pinsonneault, and S. Basu.
Solar Models: Current Epoch and Time Dependences, Neutrinos, and Helioseismological Properties.
The Astrophysical Journal, 555(2):990–1012, Jul 2001.

[25] B. T. Cleveland, T. Daily, R. Davis, Jr., et al.
Measurement of the solar electron neutrino flux with the Homestake chlorine detector.
Astrophys. J., 496:505–526, 1998.

[26] F. Kaether, W. Hampel, G. Heusser, J. Kiko, and T. Kirsten.
Reanalysis of the GALLEX solar neutrino flux and source experiments.
Phys. Lett. B, 685:47–54, 2010.

[27] J. N. Abdurashitov et al.
Solar neutrino flux measurements by the Soviet-American Gallium Experiment (SAGE) for half the 22 year solar cycle.
J. Exp. Theor. Phys., 95:181–193, 2002.

[28] M. H. Ahn, E. Aliu, S. Andringa, et al.
Measurement of neutrino oscillation by the K2K experiment.
Physical Review D, 74(7), Oct 2006.

[29] M Kobayashi and T Maskawa.
CP-violation in the renormalizable theory of weak interaction.
Progr. Theor. Phys. (Kyoto), v. 49, no. 2, pp. 652-657, 49(2), 2 1973.

[30] C. Giuntu and C. Kim.

Fundamentals of Neutrino Physics and Astrophysics.
Oxford University Press, Oxford, UK, 1 edition, 2007.

[31] A. Gando, Y. Gando, H. Hanakago, et al.
Reactor On-Off Antineutrino Measurement with KamLAND.
Phys. Rev. D, 88:033001, 08 2013.

[32] K. S. Hirata et al.
Observation of a small atmospheric muon-neutrino / electron-neutrino ratio in Kamiokande.
Phys. Lett. B, 280:146–152, 1992.

[33] D. Casper, R. Becker-Szendy, C. B. Bratton, et al.
Measurement of atmospheric neutrino composition with the IMB-3 detector.
Phys. Rev. Lett., 66:2561–2564, May 1991.

[34] M. G. Aartsen et al.
Neutrino oscillation studies with IceCube-DeepCore.
Nucl. Phys. B, 908:161–177, 2016.

[35] P. Adamson, L. Aliaga, D. Ambrose, et al.
Measurement of the Neutrino Mixing Angle θ_{23} in NOvA.
Physical Review Letters, 118, 04 2017.

[36] N. Agafonova, A. Aleksandrov, A. Anokhina, S. Aoki, et al.
Discovery of τ Neutrino Appearance in the CNGS Neutrino Beam with the OPERA Experiment.
Phys. Rev. Lett., 115:121802, Sep 2015.

[37] M. Apollonio et al.
Search for neutrino oscillations on a long baseline at the CHOOZ nuclear power station.
Eur. Phys. J. C, 27:331–374, 2003.

[38] F. Boehm, J. Busenitz, B. Cook, G. Gratta, et al.
Final results from the Palo Verde Neutrino Oscillation Experiment.
Physical review D: Particles and fields, 64, 08 2001.

[39] F. P. An, A. B. Balantekin, H. R. Band, M. Bishai, et al.
Improved measurement of the reactor antineutrino flux and spectrum at Daya Bay.
Chinese Physics C, 41(1):013002, jan 2017.

[40] Y. Abe, J. C. dos Anjos, J. C. Barriere, E. Baussan, et al.
Improved measurements of the neutrino mixing angle θ_{13} with the Double Chooz detector.
Journal of High Energy Physics, 2014(10), Oct 2014.

[41] J. K. Ahn et al.
RENO: An Experiment for Neutrino Oscillation Parameter θ_{13} Using Reactor Neutrinos at Yonggwang.
arXiv:1003.1391 [hep-ex], 3 2010.

[42] C. Giganti, S. Lavignac, and M. Zito.
Neutrino oscillations: The rise of the PMNS paradigm.
Prog. Part. Nucl. Phys., 98:1–54, 2018.

[43] S. P. Rosen and J. M. Gelb.
Mikheyev-Smirnov-Wolfenstein enhancement of oscillations as a possible solution to the solar-neutrino problem.
Phys. Rev. D, 34:969–979, Aug 1986.

[44] J. Arafune, M. Koike, and J. Sato.
CP Violation and Matter Effect in Long Baseline Neutrino Oscillation Experiments.
Physical Review D, 56:3093, 03 1997.

[45] M. Bass, M. Bishai, D. Cherdack, M. Diwan, et al.
Baseline optimization for the measurement of CP violation, mass hierarchy, and θ_{23}octant in a long-baseline neutrino oscillation experiment.
Physical Review D, 91(5), Mar 2015.

[46] I. Esteban, M.C. Gonzalez-Garcia, M. Maltoni, T. Schwetz, and A. Zhou.
The fate of hints: updated global analysis of three-flavor neutrino oscillations.
Journal of High Energy Physics, 2020(9), Sep 2020.

[47] Babak Abi et al.
Deep Underground Neutrino Experiment (DUNE), Far Detector Technical Design Report, Volume II: DUNE Physics.
arXiv:2002.03005v2 [hep-ex], 2 2020.

[48] C. Rubbia.
The Liquid Argon Time Projection Chamber: A New Concept for Neutrino Detectors.
CERN-EP-INT, 5 1977.

[49] C. Thorn.
Properties of LAr.
Technical report, Brookhaven National Laboratory, 2009.

[50] Babak Abi et al.
Deep Underground Neutrino Experiment (DUNE), Far Detector Technical Design Report, Volume I Introduction to DUNE.

JINST, 15(08):T08008, 2020.

[51] K.A. Olive.
Review of Particle Physics.
Chinese Physics C, 40(10):100001, 10 2016.

[52] Y. Li et al.
Measurement of Longitudinal Electron Diffusion in Liquid Argon.
Nucl. Instrum. Meth. A, 816:160–170, 2016.

[53] T. Doke, A. Hitachi, S. Kubota, A. Nakamoto, and T. Takahashi.
Estimation of Fano factors in liquid argon, krypton, xenon and xenon-doped liquid argon.
Nuclear Instruments and Methods, 134(2):353–357, 1976.

[54] S. Amoruso et al.
Study of electron recombination in liquid argon with the ICARUS TPC.
Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, 523(3):275–286, 2004.

[55] S. Kubota, A. Nakamoto, T. Takahashi, T. Hamada, et al.
Recombination luminescence in liquid argon and in liquid xenon.
Phys. Rev. B, 17:2762–2765, Mar 1978.

[56] B. Aimard, L. Aizawa, C. Alt, J. Asaadi, et al.
Study of scintillation light collection, production and propagation in a 4 tonne dual-phase LArTPC.
Journal of Instrumentation, 16(03):P03007, Mar 2021.

[57] A. Hitachi, T. Takahashi, N. Funayama, K. Masuda, et al.
Effect of ionization density on the time dependence of luminescence from liquid argon and xenon.
Phys. Rev. B, 27:5279–5285, 1983.

[58] K. Scholberg.
Supernova Neutrino Detection.
Annual Review of Nuclear and Particle Science, 62(1):81–103, Nov 2012.

[59] I Gil-Botella and A Rubbia.
Oscillation effects on supernova neutrino rates and spectra and detection of the shock breakout in a liquid argon TPC.
Journal of Cosmology and Astroparticle Physics, 2003(10):009–009, Oct 2003.

[60] A. Bueno, I. Gil-Botella, and A. Rubbia.

Supernova Neutrino Detection in a liquid Argon TPC.
arXiv: hep-ph/0307222v1 [hep-ph], Aug 2003.

[61] K. Ratnatunga and S. Bergh.
The rate of stellar collapses in the galaxy.
The Astrophysical Journal, 343:713–717, 07 1989.

[62] P. M. Dragicevich, D. G. Blair, and R. R. Burman.
Why are supernovae in our Galaxy so frequent?
Monthly Notices of the Royal Astronomical Society, 302(4):693–699, 02 1999.

[63] B. C. Reed.
New Estimates of the Solar-Neighborhood Massive Star Birthrate and the Galactic Supernova Rate.
The Astronomical Journal, 130(4):1652–1657, Oct 2005.

[64] K. Rozwadowska, F. Vissani, and E. Cappellaro.
On the rate of core collapse supernovae in the milky way.
New Astronomy, 83:101498, Feb 2021.

[65] A. Ankowski et al.
Supernova Physics at DUNE.
arXiv:1608.07853v1 [hep-ex], 2016.

[66] K. Hirata, T. Kajita, M. Koshiba, M. Nakahata, et al.
Observation of a neutrino burst from the supernova SN1987A.
Phys. Rev. Lett., 58:1490–1493, Apr 1987.

[67] R. M. Bionta, G. Blewitt, C. B. Bratton, D. Casper, et al.
Observation of a neutrino burst in coincidence with supernova 1987A in the Large Magellanic Cloud.
Phys. Rev. Lett., 58:1494–1496, Apr 1987.

[68] E. N. Alekseev, L. N. Alekseeva, V. I. Volchenko, and I. V. Krivosheina.
Possible detection of a neutrino signal on February 23, 1987 with the Baksan underground scintillation telescope of the Nuclear Research Institute of the Soviet Academy of Sciences.
Pisma v Zhurnal Eksperimentalnoi i Teoreticheskoi Fiziki, 45:461–464, May 1987.

[69] D. N. Schramm and J. W. Truran.
New physics from Supernova SN1987A.
Phys. Rept., 189:89–126, 1990.

[70] G. G. Raffelt.
Particle physics from stars.
Annual Review of Nuclear and Particle Science, 49(1):163–216, Dec 1999.

[71] T. J. Loredo and D. Q. Lamb.
Bayesian analysis of neutrinos observed from supernova SN 1987A.
Physical Review D, 65(6), Feb 2002.

[72] G. Raffelt and D. Seckel.
Bounds on exotic-particle interactions from SN1987A.
Phys. Rev. Lett., 60:1793–1796, May 1988.

[73] R. Barbieri and R. N. Mohapatra.
Limit on the Magnetic Moment of the Neutrino from Supernova 1987A Observations.
Phys. Rev. Lett., 61:27–30, Jul 1988.

[74] G. Barbiellini and G. Cocconi.
Electric charge of the neutrinos from SN 1987A.
Nature, 329(6134):21–22, September 1987.

[75] S. A. Colgate and R. H. White.
The Hydrodynamic Behavior of Supernovae Explosions.
The Astophysical Journal, 143:626, March 1966.

[76] S. Woosley and T. Janka.
The physics of core-collapse supernovae.
Nature Physics, 1(3):147–154, Dec 2005.

[77] T. Totani, K. Sato, H. E. Dalhed, and J. R. Wilson.
Future Detection of Supernova Neutrino Burst and Explosion Mechanism.
The Astrophysical Journal, 496(1):216–225, Mar 1998.

[78] J. Gava, J. Kneller, C. Volpe, and G. C. McLaughlin.
Dynamical Collective Calculation of Supernova Neutrino Signals.
Physical Review Letters, 103(7), Aug 2009.

[79] H. Minakata, H. Nunokawa, R. Tomàs, and J. W. F. Valle.
Parameter degeneracy in flavor-dependent reconstruction of supernova neutrino fluxes.
Journal of Cosmology and Astroparticle Physics, 2008(12):006, Dec 2008.

[80] J. F. Beacom, R. N. Boyd, and A. Mezzacappa.
Black hole formation in core-collapse supernovae and time-of-flight measurements of the neutrino masses.
Physical Review D, 63(7), Mar 2001.

[81] J. F. Cherry, J. Carlson, A. Friedland, G. M. Fuller, and A. Vlasenko.
Halo modification of a supernova neutronization neutrino burst.
Physical Review D, 87(8), Apr 2013.

[82] R. C. Schirato and G. M. Fuller.
Connection between supernova shocks, flavor transformation, and the neutrino signal.
arXiv:astro-ph/0205390 [astro-ph], 2002.

[83] F. Hanke, A. Marek, B. Müller, and H. Janka.
Is strong SASI activity the key to successful neutrino-driven supernova explosions?
The Astrophysical Journal, 755(2):138, Aug 2012.

[84] A. Friedland and A. Gruzinov.
Neutrino signatures of supernova turbulence.
arXiv:astro-ph/0607244, Jul 2006.

[85] B. Abi, R. Acciarri, M. Acero, G. Adamov, et al.
Supernova Neutrino Burst Detection with the Deep Underground Neutrino Experiment.
arXiv:2008.06647 [hep-ex], 08 2020.

[86] K. Scholberg.
Supernova signatures of neutrino mass ordering.
Journal of Physics G: Nuclear and Particle Physics, 45(1):014002, Dec 2017.

[87] L. Hudepohl.
Neutrinos from the Formation, Cooling and Black Hole Collapse of Neutron Stars.
PhD thesis, Technical University Munich, Max Planck Institue for Astrophysics, 2014.

[88] Scholberg K.
eventsperbin2d-argon.root histogram.
Private communication.

[89] K. Scholberg, J. B. Albert, and J. Vasel.
SNOwGLoBES: SuperNova Observatories with GLoBES, September 2021.

[90] S. Brice.
Proton Improvement Plan II: An 800 MeV Superconducting Linac to Support Megawatt Proton Beams at Fermilab.
Nuclear and Particle Physics Proceedings, 273-275:238–243, 04 2016.

[91] R. Acciarri, M. A. Acero, M. Adamowski, C. Adams, et al.
Long-Baseline Neutrino Facility (LBNF) and Deep Underground Neutrino Experiment (DUNE) Conceptual Design Report Volume 1: The LBNF and DUNE Projects.
arXiv:1601.05471 [physics.ins-det], 2016.

[92] A. Abed Abud et al.
Deep Underground Neutrino Experiment (DUNE) Near Detector Conceptual Design Report.
Instruments, 5(4):31, 2021.

[93] M. Adinolfi, F. Ambrosino, A. Andryakov, A. Antonelli, et al.
The tracking detector of the KLOE experiment.
Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, 488(1):51–73, 2002.

[94] S. Amerio, Salvatore Amoruso, M. Antonello, P. Aprili, et al.
Design, construction and tests of the ICARUS T600 detector.
Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, 527:329–410, 07 2004.

[95] M. Harańczyk.
The ICARUS detector. Past, present and future.
Journal of Physics: Conference Series, 798:012162, jan 2017.

[96] C. Adams, R. An, J. Anthony, J. Asaadi, et al.
Ionization electron signal processing in single phase LArTPCs. Part II. Data/simulation comparison and performance in MicroBooNE.
Journal of Instrumentation, 13(07):P07007–P07007, jul 2018.

[97] B. Abi, A. Abed Abud, R. Acciarri, M.A. Acero, et al.
First results on ProtoDUNE-SP liquid argon time projection chamber performance from a beam test at the CERN Neutrino Platform.
Journal of Instrumentation, 15(12):P12004–P12004, dec 2020.

[98] S. Palestini.
Fd2 vertical drift conceptual design report.
Technical report, CERN, 2022.

[99] J. Asaadi, M. Auger, R. Berner, A. Bross, et al.
A New Concept for Kilotonne Scale Liquid Argon Time Projection Chambers.
Instruments, 4(1), 2020.

[100] B. Abi, R. Acciarri, M.A. Acero, G. Adamov, et al.
Volume IV. The DUNE far detector single-phase technology.
Journal of Instrumentation, 15(08):T08010–T08010, aug 2020.

[101] L. Whitehead.
Dune far detector task force final report.

Technical report, University of Cambridge, 2018.

[102] A.A. Machado, E. Segreto, D. Warner, A. Fauth, et al.
The X-ARAPUCA: an improvement of the ARAPUCA device.
Journal of Instrumentation, 13(04):C04026–C04026, Apr 2018.

[103] C.E. Shannon.
Communication in the Presence of Noise.
Proceedings of the IRE, 37(1):10–21, 1949.

[104] W. Wu.
FELIX: the New Detector Interface for the ATLAS Experiment.
IEEE Transactions on Nuclear Science, 66(7):986–992, Jul 2019.

[105] S. Al Kharusi, S. Y. BenZvi, J. S. Bobowski, W. Bonivento, et al.
SNEWS 2.0: a next-generation supernova early warning system for multi-messenger astronomy.
New Journal of Physics, 23(3):031201, mar 2021.

[106] J. Klein.
Trigger rates and data volume estimates for dune.
Technical report, University of Pennsylvania, 2018.

[107] P. Abratenko, M. Alrashed, R. An, J. Anthony, et al.
The continuous readout stream of the MicroBooNE liquid argon time projection chamber for detection of supernova burst neutrinos.
Journal of Instrumentation, 16(02):P02008–P02008, feb 2021.

[108] M. Qiang, S. Muthukrishnan, and M. Sandler.
Frugal Streaming for Estimating Quantiles:One (or two) memory suffices.
arXiv:1407.1121 [cs.DB], 2014.

[109] K. Manolopoulos.
Upstream DAQ Tech Review TPG FW.
07 2020.

[110] *AMBA AXI and ACE Protocol Specification*, 2011.

[111] DUNE Collaboration, B Abi, R Acciarri, Mario Acero, et al.
The Single-Phase ProtoDUNE Technical Design Report.
arXiv:1706.07081 [physics.ins-det], 06 2017.

[112] T. Feist.
Vivado Design Suite.

Technical report, Xilinx, 06 2012.

[113] K. Biery, E. Flumerfelt, J. Freeman, W. Ketchum, et al.
artdaq: DAQ software development made simple.
Journal of Physics: Conference Series, 898:032013, oct 2017.

[114] T. Bezerra, A. Booth, A. Borkum, J. Hartnell, et al.
Dune supernovae triggering.
Technical report, University of Sussex, 2020.

[115] The GRAVITY Collaboration et al.
A geometric distance measurement to the Galactic center black hole with 0.3% uncertainty.
A&A, 625:L10, 2019.

[116] A. Mirizzi, G. G. Raffelt, and P. D. Serpico.
Earth matter effects in supernova neutrinos: optimal detector locations.
Journal of Cosmology and Astroparticle Physics, 2006(05):012–012, May 2006.

[117] G. Pietrzyński, D. Graczyk, A. Gallenne, W. Gieren, et al.
A distance to the Large Magellanic Cloud that is precise to one per cent.
Nature, 567(7747):200–203, Mar 2019.

[118] P. Rodrigues.
Single phase tpc trigger primitive algorithms for tdr.
Technical report, University of Oxford, 2018.

[119] E.L. Snider and G. Petrillo.
LArSoft: toolkit for simulation, reconstruction and analysis of liquid argon TPC neutrino detectors.
Journal of Physics: Conference Series, 898:042057, oct 2017.

[120] S. Agostinelli, J. Allison, K. Amako, J. Apostolakis, et al.
Geant4 a simulation toolkit.
Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, 506(3):250–303, 2003.

[121] S. Gardiner.
Simulating low-energy neutrino interactions with MARLEY.
Computer Physics Communications, 269:108123, 2021.

[122] The MicroBooNE Collaboration.
Study of Reconstructed ^{39}Ar Beta Decays at the MicroBooNE Detector.
Technical report, FNAL, 2018.

[123] B. D. Flury.
Acceptance-Rejection Sampling Made Easy.
SIAM Review, 32(3):474–476, 1990.

[124] C. Grierson and J. Schiefelbein.
Machine Learning.
McGraw-Hill, 1997.

[125] D. Wolpert.
The Lack of A Priori Distinctions Between Learning Algorithms.
Neural Computation, 8, 03 1996.

[126] I. Goodfellow, Y. Bengio, and A. Courville.
Deep Learning.
MIT Press, 2016.
`http://www.deeplearningbook.org`.

[127] S. Kullback and R. Leibler.
On information and sufficiency.
The Annals of Mathematical Statistics, 22:79–86, 03 1951.

[128] H. Cramer.
Mathematical Methods Of Statistics.
Princeton University Press, 1946.

[129] S. Das Gupta.
Selected Papers of C. R. Rao.
New York: Wiley, 1994.

[130] D. Rumelhart, G. Hinton, and R. Williams.
Learning representations by back-propagating errors.
Nature, 323:533–536, 10 1986.

[131] D. P. Kingma and J. Ba.
Adam: A Method for Stochastic Optimization.
arXiv:1412.6980 [cs.LG], 2017.

[132] V. Nair and G. Hinton.
Rectified linear units improve restricted boltzmann machines.
In *Proceedings of the 27th International Conference on Machine Learning*, 2010.

[133] A. Krizhevsky, I. Sutskever, and G. E. Hinton.
ImageNet Classification with Deep Convolutional Neural Networks.

In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[134] K. He, X. Zhang, S. Ren, and J. Sun.
Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification.
arXiv:1502.01852 [cs.CV], 2015.

[135] A. L. Maas.
Rectifier Nonlinearities Improve Neural Network Acoustic Models.
In *Proceedings of the 30th International Conference on Machine Learning*, volume 28, 2013.

[136] Y. Zhou and R. Chellappa.
Computation of optical flow using a neural network.
IEEE 1988 International Conference on Neural Networks, pages 71–78 vol.2, 1988.

[137] S. Ioffe and C. Szegedy.
Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.
arXiv:1502.03167 [cs.LG], 2015.

[138] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry.
How Does Batch Normalization Help Optimization?
arXiv:1805.11604 [stat.ML], 2019.

[139] K. He, X. Zhang, S. Ren, and J. Sun.
Deep residual learning for image recognition.
arXiv:1512.03385 [cs.CV], abs/1512.03385, 2015.

[140] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger.
Densely Connected Convolutional Networks.
arXiv:1608.06993 [cs.CV], 2018.

[141] O. Ronneberger, P. Fischer, and T. Brox.
U-Net: Convolutional Networks for Biomedical Image Segmentation.
arXiv:1505.04597 [cs.CV], 2015.

[142] J. Redmon and A. Farhadi.
YOLOv3: An Incremental Improvement.
arXiv:1804.02767 [cs.CV], 2018.

[143] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, et al.
TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015.
Software available from tensorflow.org.

[144] F. Chollet.
Keras.
`https://github.com/fchollet/keras`, 2015.

[145] H. Anh, R. Chiriac, et al.
keras-yolo3.
`https://github.com/experiencor/keras-yolo3`, 2020.

[146] B. Abi, R. Acciarri, M.A. Acero, G. Adamov, et al.
Neutrino interaction classification with a convolutional neural network in the DUNE far detector.
Physical Review D, 102(9), Nov 2020.

[147] MicroBooNE collaboration et al.
The Pandora multi-algorithm approach to automated pattern recognition of cosmic-ray muon and neutrino events in the MicroBooNE detector.
arXiv:1708.03135 [hep-ex], 2017.

[148] C. Adams, M. Alrashed, R. An, J. Anthony, et al.
Deep neural network for pixel-level electromagnetic particle identification in the MicroBooNE liquid argon time projection chamber.
Physical Review D, 99(9), May 2019.

[149] S. Ren, K. He, R. Girshick, and J. Sun.
Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.
arXiv:1506.01497 [cs.CV], 2016.

[150] MicroBooNE collaboration et al.
Convolutional Neural Networks Applied to Neutrino Events in a Liquid Argon Time Projection Chamber.
Journal of Instrumentation, 12, 11 2016.

[151] L. Breiman.
Bagging predictors.
Machine Learning, 24:123–140, 08 1996.

[152] B. Roe, H. Yang, Y. Liu, I. Stancu, and G. McGregor.
Boosted Decision Trees as an Alternative to Artificial Neural Networks for Particle Identification.
Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, 543(2-3), 2005.

[153] B. Roe, H. Yang, and J. Zhu.
Studies of Boosted Decision Trees for MiniBooNE Particle Identification.
arXiv:physics/0508045 [physics.data-an], 2005.

[154] A. Sherstinsky.
Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network.
arXiv:1808.03314 [cs.LG], 2018.

[155] Y.-J. Jwa, G. Di Guglielmo, L. P. Carloni, and G. Karagiorgi.
Accelerating Deep Neural Networks for Real-time Data Selection for High-resolution Imaging Particle Detectors.
In *2019 New York Scientific Data Summit (NYSDS)*, pages 1–10, 2019.

[156] A. Krizhevsky, I. Sutskever, and G. E. Hinton.
Imagenet classification with deep convolutional neural networks.
In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[157] R. Nane, V.-M. Sima, C. Pilato, J. Choi, et al.
A Survey and Evaluation of FPGA High-Level Synthesis Tools.
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 35(10):1591–1604, 2016.

[158] J. Redmon and A. Farhadi.
YOLO9000: Better, Faster, Stronger.
arXiv:1612.08242 [cs.CV], 2016.

[159] A. Bochkovskiy, C.Y. Wang, and H.Y.M. Liao.
Yolov4: Optimal speed and accuracy of object detection.
arXiv:2207.02696 [cs.CV], 2020.

[160] C. Ding, S. Wang, N. Liu, K. Xu, et al.
REQ-YOLO: A Resource-Aware, Efficient Quantization Framework for Object Detection on FPGAs.
arXiv:1909.13396 [cs.CV], 2019.

[161] H. Nakahara, M. Shimoda, and S. Sato.
A Demonstration of FPGA-Based You Only Look Once Version2 (YOLOv2).
In *2018 28th International Conference on Field Programmable Logic and Applications (FPL)*, pages 457–4571, 2018.

[162] F. Fahim, B. Hawks, C. Herwig, J. Hirschauer, et al.
hls4ml: An Open-Source Codesign Workflow to Empower Scientific Low-Power Machine Learning Devices.
arXiv:2103.05579 [cs.LG], 2021.

[163] D. H. Noronha, B. Salehpour, and S. J. E. Wilton.
LeFlow: Enabling Flexible FPGA High-Level Synthesis of Tensorflow Deep Neural Networks.
arXiv:1807.05317 [cs.LG], 2018.

[164] J. Pedoeem and R. Huang.
YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers.
arXiv:1811.05588 [cs.CV], 2018.

[165] F. Vilardell, I. Ribas, C. Jordi, E. L. Fitzpatrick, and E. F. Guinan.
The distance to the Andromeda galaxy from eclipsing binaries.
Astronomy and Astrophysics, 509:A70, Jan 2010.

[166] B. Graham, M. Engelcke, and L. van der Maaten.
3D Semantic Segmentation with Submanifold Sparse Convolutional Networks.
arXiv:1711.10275 [cs.CV], 2017.

[167] L. Dominé and K. Terao.
Scalable deep convolutional neural networks for sparse, locally dense liquid argon time projection chamber data.
Physical Review D, 102(1), Jul 2020.

Acronyms

A

ADC analogue to digital converted. 30, 35–40, 43–45, 56, 64, 66–68, 79, 101–104, 108, 137, 141

ANN artificial neural network. 82, 83, 86–88, 100, 142

APA anode plane assembly. 27–30, 32, 34, 36, 39, 40, 44, 64–66, 70, 102, 106, 107, 110, 131, 136, 137, 141

ASIC application specific integrated circuit. 30

AXI4-Stream advanced extensible interface 4 stream. 39

B

BDT boosted decision tree. 103, 104

BE back-end. 30–33

BR BoardReader. 46, 48

BRAM block random access memory. 39

C

CC charged current. 7, 8, 17, 22, 23

CCM control, configuration and monitoring. 30, 31

CE cold electronics. 28–30, 49

CERN European Council for Nuclear Research. 39

CM Cryo Mezzanine. 30, 31

CNN convolutional neural network. 3, 83, 88, 100, 102–105

CPA cathode plane assembly. 27, 65, 69

CPU central processing unit. 2, 33, 35, 63, 64, 78, 80, 102

CR central router. 38, 39, 46

CVN convolutional visual network. 102

D

DAG directed acyclic graph. 83, 85

DAQ data acquisition. 1–3, 27, 30–35, 38, 43, 49, 63, 102–105, 135, 137–139

DDR4 double data rate fourth generation. 32

DFO data flow orchestrator. 33

DMA direct memory access. 48

dpRAM dual ported random access memory. 38, 39

DS data selection. 30–33, 38, 39

DUNE Deep Underground Neutrino Experiment. 1, 2, 5, 6, 10, 12, 22, 23, 25–27, 30, 31, 34, 35, 38–40, 44, 58, 63–67, 78–80, 101–104, 135, 139, 140

E

EB event builder. 33

ES elastic scattering. 7, 8, 17

ETI external trigger interface. 33

F

FD far detector. 1, 2, 25–27, 30, 34, 40, 58, 64, 69, 76, 79, 80, 123, 124, 139, 140

FE front-end. 30, 32

FELIX front-end link exchange. 32, 38–40, 46, 48, 101

FEMB front-end motherboard. 30, 38

FFT fast Fourier transform. 142

FIR finite impulse response. 36, 39, 43–46, 48, 56, 64, 67, 106

fn false negative. 100, 120, 144, 145

FNAL Fermi National Accelerator Laboratory. 1, 25, 31

fp false positive. 100, 120, 124, 129–131, 144, 145

FPGA field programmable gate array. 1, 2, 32, 33, 35, 38–40, 43, 58, 63, 64, 67, 79, 80, 104–106, 137–140

FPR false positive rate. 34–36, 123–130, 133

FPS frames per second. 105, 136

G

GEANT4 GEometry ANd Tracking 4. 64–67, 107

GPU graphics processing unit. 33, 101, 102, 105, 136, 137

H

HF hit-finder. 39, 57

HLF high level filter. 31–33, 101–105, 137, 138

HLS high level synthesis. 104

HLS4ML High Level Synthesis for Machine Learning. 105

I

IOU intersection over union. 97–100, 109, 112, 116, 122, 123, 133, 138, 144, 145

K

K2K KEK to Kamioka. 8, 9

KL Kullback-Leibler. 82

KLOE $\mathrm{K_L}^0$ LOng Experiment. 26

L

LAr liquid argon. 1, 6, 12–16, 22, 23, 26, 27, 30, 34, 40, 65, 66, 69

LArTPC liquid argon time projection chamber. 2, 12, 13, 25–27, 40, 64, 102, 103, 137, 139

LBNF Long Baseline Neutrino Facility. 25

LINAC linear accelerator. 25

LMC Large Magellanic Cloud. 18, 63, 78–80, 130, 133, 135, 137, 138, 140

LSB least significant bit. 49, 58

LSTM long short-term memory. 104

M

MARLEY Model of Argon Reaction Low Energy Yields. 65, 67, 69, 70

MC Monte Carlo. 74, 79, 130

MCR main communications room. 30, 31

MGT multi-gigabit transceiver. 32, 38, 39, 48

MicroBooNE Micro Booster Neutrino Experiment. 34

MIP minimum-ionising particle. 14, 30

ML machine learning. 3, 81, 82, 88, 101, 102, 104, 108, 133, 135, 140

MLE maximum likelihood estimation. 82, 85, 100

MLT module level trigger. 31, 33, 102, 103

MPD Multi-Purpose Detector. 25, 26

MSB most significant bit. 49, 57

MSE mean squared error. 82, 84, 85, 96

N

NC neutral current. 7, 8, 22, 23

ND near detector. 25–27

NMS non-maximal suppression. 99, 100, 120, 123, 124, 126, 131, 132, 136

P

PCIe Peripheral Component Interconnect Express. 32

PIP-II Proton Improvement Plan II. 25

PMNS Pontecorvo-Maki-Nakagawa-Sakata. 8

ProtoDUNE-SP ProtoDUNE single phase. 26, 38–40, 58

PS pedestal subtraction. 36, 39, 43–46, 48

R

R-CNN region-based neural network. 105

RAM random-access memory. 32, 33, 137

ReLU rectified linear unit. 86, 87, 90, 92, 93

RNN recurrent neural network. 104

ROI region of interest. 102, 104

RPN region proposal network. 105

RU readout unit. 32, 102, 104, 105, 137, 138

S

SADC summed ADC. 50, 107

SAND System for on-Axis Neutrino Detection. 26

SGD stochastic gradient descent. 86, 95, 113, 142, 143

SiPM silicon photomultiplier. 27

SK Super-Kamiokande. 8, 9

SLR super-logic region. 48

SM Standard Model. 1, 5, 6

SN supernova. 2, 6, 18–23, 33, 67, 74, 76, 77, 79, 80, 102, 104, 107, 108, 120–122, 136, 140

SNB supernova burst. 1–3, 6, 22, 23, 32–35, 37, 63, 64, 74, 76–80, 101–105, 108, 120, 129, 130, 133, 135, 137–140

SNe supernovae. 2, 18, 21, 23, 80

SNEWS supernova early warning system. 33

SNO Sudbury Neutrino Observatory. 7, 9

SNOwGLoBES SuperNova Observatories with GLoBES. 22, 23

SP single phase. 26, 27, 39, 40, 58, 139, 140

SPV summed pixel value. 99, 120, 123, 124, 126, 128, 129, 132, 136, 137

SSCN submanifold sparse convolutional network. 137

SSD solid-state drive. 32

SSM standard solar model. 7

SURF Sanford Underground Research Facility. 1, 25, 30

T

TC trigger cluster. 33–35, 37, 63, 64, 70, 71, 73, 74, 76–79, 101, 104, 130, 140–142

tn true negative. 100

TOT ticks over threshold. 37, 55, 69, 70

TP trigger primitive. 32–40, 43, 46–58, 63, 64, 67–74, 79, 80, 100–102, 104–108, 137–142

tp true positive. 100, 120, 131, 144, 145

TPC time projection chamber. 15, 25, 30, 32, 34, 35, 38–40, 102, 104, 108, 110, 137, 140

TPG trigger primitive generation. 1, 33, 35, 38–40, 43, 46–49, 53, 55, 58, 63, 64, 67, 68, 78, 102, 105, 106, 137–140

U

UD upstream data acquisition. 30–34, 38, 43, 48, 104, 137

V

VHDL Very High-Speed Integrated Circuits Hardware Description Language. 44

W

WAN wide area network. 31

WIB warm interface board. 30, 32, 38–40, 48

Y

YOLO you-only-look-once. 105, 135

YOLOv2 You-Only-Look-Once version 2. 105, 137

YOLOv3 You-Only-Look-Once version 3. 1–3, 82, 92, 95, 99–101, 105–107, 110, 113, 120, 129, 133, 135–138, 140

YOLOv4 You-Only-Look-Once version 4. 105