COPENHAGEN UNIVERSITY

MASTER THESIS

Inelasticity reconstruction for IceCube neutrino observatory upgrade

Reconstructing inelasticity for muon Neutrinos In monte carlo Data From The IceCube Neutrino Observatory Upgrade Using Graph Neural Networks and Transformers

Author:
Moust Oskar William Holmes

Supervisor:
D. Jason Koskinen

A thesis submitted in fulfillment of the requirements for the degree of Master in Computational Physics

at the

High Energy Particle Physics Department
Niels Bohr Institute

May 20, 2023

Declaration of Authorship

I, Moust Oskar William Holmes, declare that this thesis titled, "Inelasticity reconstruction for IceCube neutrino observatory upgrade" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made so clear.

Signed:

Date:

COPENHAGEN UNIVERSITY

Abstract

High Energy Particle Physics

Niels Bohr Institute

Master in Computational Physics

Inelasticity reconstruction for IceCube neutrino observatory upgrade

by Moust Oskar William Holmes

This thesis aims to enhance the inelasticity reconstruction for the IceCube Neutrino Observatory Upgrade. While features such as energy and direction of incoming particles have been reconstructed with reasonable precision, the inelasticity of the neutrino interaction has remained elusive, presenting challenges to conventional reconstruction methods. In response to this, I propose a new loss function based on the beta distribution, designed to reward the model more for accurate predictions it is confident in and penalize it less for challenging, incorrect predictions. I integrate this loss function into the Dynedge model from Graphnet, previously used for IceCube reconstruction, and a novel Transformer model, inspired by the runner-up in the Kaggle competition "IceCube - Neutrinos in Deep Ice", aimed at improving zenith angle reconstruction. My work has the potential to enhance the sensitivity of the IceCube detector, improving our understanding of neutrinos and their properties.

Acknowledgements

First and foremost, I would like to extend my sincerest gratitude to my supervisor, D. Jason Koskinen, for his unwavering guidance and support throughout this research journey.

Special thanks to Troels Petersen, whose invaluable advice was instrumental in shaping this work.

I am deeply thankful to the current and former members of the IceCube group at NBI: Marcus, Tom, Tetiana, Amalie, Linea, Katrine, Jorge, James, and Clotilde. Your warmth, talent, and welcoming nature have made my time in the group both productive and enjoyable. and also Kevin.

To my office mates: Peter, Morten, Simon, Andreas J., and Andreas H., I owe you many thanks. Our discussions and the memorable Machine Learning breakfasts have been highlights of my time here, even if my sleep schedule didn't always cooperate.

Lastly, I'd like to express my appreciation to the members of the GraphNet team: Rasmus, Aske, and Andreas S. The constructive developer meetings and your consistent assistance with GraphNet have been invaluable to my research.

Contents

List of most important Abbreviations

IceCube	IceCube Neutrino Observatory
GraphNeT	Graph Neural Networks for Neutrino Telescope Event Reconstruction
GNN	**G**raph **N**eural **N**etwork
FNN	**F**eed-forward **N**eural **N**etwork
QE	**Q**uasi-**E**lastic Scattering
RES	**Res**onance Production
DIS	**D**eep **I**nelastic **S**cattering
DOM	**D**igital **O**ptical **M**odule (The individual sensor in IceCube)
CC	**C**harged **C**urrent
NC	**N**eutral **C**urrent
EM	**E**lectro**m**agnetic
PMT	**P**hoto**m**ultiplier **T**ube
TPR	**T**rue**P**ositive **R**ate
TNR	**T**rue**N**egative **R**ate
FPR	**F**alse**P**ositive **R**ate
ICL	**I**ceCube**C**ount **L**ab
SLC	**S**oft**L**ocal **C**oincidence
HLC	**H**ard**L**ocal **C**oincidence
ROC Curve	**R**eceiver **O**perating **C**haracteristics Curve
FADC	**F**ast **A**nalog to **D**igital **C**onverter

Chapter 1

Introduction

The study of particles has continuously expanded our understanding of the universe and its underlying mechanics. The neutrino, despite being the most abundant, remains enigmatic and challenging for researchers. Neutrinos are elusive, both in terms of our understanding and their interaction with other matter. Their rarity in interaction is a double-edged sword: while it makes them challenging to study, it also renders them invaluable. Due to their minimal interactions, neutrinos can be traced back to their origins with relative certainty, as they're unlikely to have been deflected en route. However, detecting a neutrino is a monumental task. Despite their abundance, the probability of observing a neutrino interaction is minuscule. This challenge led to the creation of the IceCube neutrino observatory, which utilizes a cubic kilometer of Antarctic ice to detect neutrino interactions.

This thesis builds upon the foundation laid by a prior master's thesis at The Niels Bohr Institute, authored by Marc Jacquart and supervised by Frédéric Blanc and D. Jason Koskinen. Marc's work suggested that, under certain conditions, reconstructing inelasticity for neutrino events in IceCube could enhance its sensitivity to the neutrino mass ordering. This could potentially halve the data collection time required to achieve a sensitivity above 3σ. However, a critical component was missing: the successful reconstruction of inelasticity. This thesis aims to address that gap.

The primary focus here will shift from the theoretical underpinnings of Marc's sensitivity enhancement to the improvement of inelasticity reconstruction. An introduction to the standard model, neutrino physics phenomena, and the challenges of distinguishing between neutrinos and anti-neutrinos in IceCube will be provided. Additionally, the intricacies of the IceCube neutrino observatory, its detectors, upcoming upgrades, and its capability to detect neutrinos will be discussed.

On the technical front, a primer on machine learning will be presented, introducing a novel loss function termed "beta loss," which I developed. This cross-entropy-based function leverages the beta distribution to produce outputs ranging between 0 and 1. The GraphNet model, DynEdge, which has shown promise in reconstructing IceCube data, will be detailed. Furthermore, my Transformer-based model inspired by the runner-up entry in the recent Kaggle competition "IceCube - Neutrinos in Deep Ice" will be introduced.

This work culminates in the application of the beta loss function, used in conjunction with the DynEdge and my new Transformer model, to enhance inelasticity reconstruction. The ultimate goal is to demonstrate how this refined inelasticity can help distinguish between neutrinos and anti-neutrinos in IceCube.

Chapter 2

The Standard Model of Particle Physics

2.1 The Standard Model Of Particle Physics

The pursuit of understanding the universe's fundamental constituents has led to the development of the Standard Model of particle physics. This framework, while not exhaustive in its description of all known phenomena, provides a comprehensive picture of the elementary particles and their interactions. The Standard Model predicts the existence of a specific set of particles, grouped into fermions and bosons, which are further categorized based on their properties and roles in mediating forces. A visual representation of these particles and their attributes is shown in Figure 2.1.

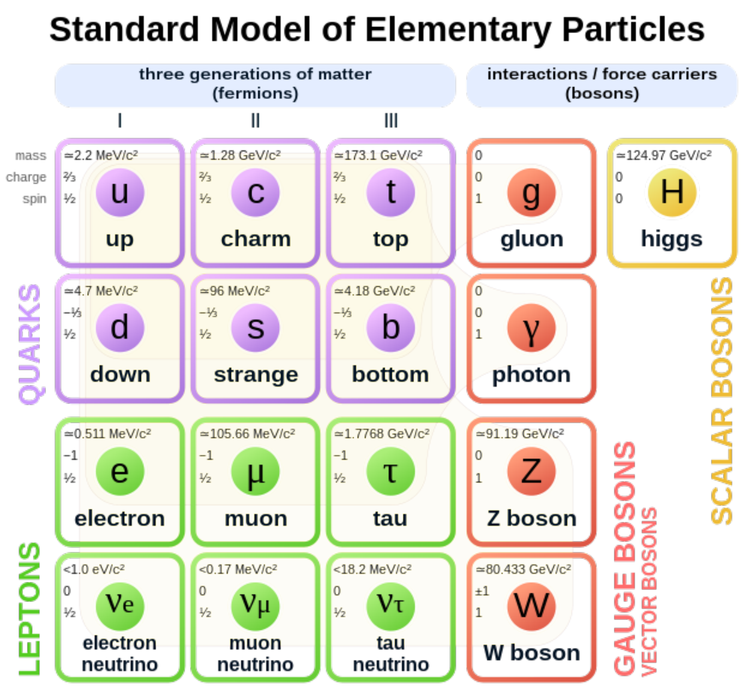


FIGURE 2.1: Overview of the Standard Model of particle physics taken from [1, 2] & . Includes all fundamental particles, their mass, electromagnetic charge and spin.

Fermions serve as the building blocks of matter and are further divided into quarks and leptons, each with their unique properties and interactions. The subsequent sections will delve deeper into the nature of these particles, exploring their characteristics and significance in the world of particle physics.

2.1.1 Fermions

Fermions: These are the fundamental particles of matter, and they come in three generations, each with increasing mass. All fermions possess a spin of 1/2 and have corresponding antiparticles.

- Quarks: Quarks are unique particles with electromagnetic charges of either -1/3 or 2/3. They also possess isospin, allowing them to interact via both the electromagnetic and the weak forces. Additionally, quarks have a property called colour charge, which means they interact via the strong force. This force grows with distance beyond a certain threshold, ensuring that quarks remain bound in colour-neutral combinations. Due to this phenomenon, known as colour confinement, quarks never appear unbound naturally.
- Hadrons: These are composite particles made up of quarks, held together by the strong force. Hadrons are further classified into two categories:
  - Baryons: These are particles made up of three quarks. The most well-known baryons are protons (composed of two up quarks and one down quark) and neutrons (composed of one up quark and two down quarks).
  - Mesons: These are particles made up of a quark and an antiquark. An example is the pion, which comes in three varieties: the positively charged π^+(made of an up quark and a down antiquark), the negatively charged π^-(made of a down quark and an up antiquark), and the neutral pi^0 (made of an up quark and an up antiquark or a down quark and a down antiquark). Pions are important in the context of neutrinos as it is one of the main ways of producing neutrinos. charged pion decays into a muon and an anti-neutrino or an anti-muon and a neutrino depending on the charge of the pion.

$$\pi^+ \rightarrow \mu^+ + \nu_\mu$$
$$\pi^- \rightarrow \mu^- + \bar{\nu_\mu}$$

- Leptons: Unlike quarks, leptons do not carry colour charge, so they don't interact via the strong force. The lepton family consists of the electron, muon, and tau particles, as well as their corresponding neutrinos. While the electron, muon, and tau have an electromagnetic charge of -1, neutrinos are neutral. This neutrality means that neutrinos only interact via the weak force, making them incredibly elusive and challenging to detect.
- Antiparticles: For every fermion, there's a corresponding antiparticle. These antiparticles have the same mass but opposite quantum numbers. For instance, the antiparticle of an electron (with a charge of -1) is the positron, which has a charge of +1. Similarly, neutrinos, which are neutral, have corresponding antineutrinos. While neutrinos and antineutrinos both lack electric charge, they differ in other quantum properties, like their lepton number. Neutrinos have a positive lepton number, while antineutrinos have a negative lepton number. When particles and antiparticles collide, they can annihilate each other, releasing energy. It's essential to note that while all fermions have antiparticles, not all particles, like certain bosons, possess antiparticle counterparts.

2.1.2 Bosons

Boson particles mediate the interactions between fermions. There are four vector bosons with spin 1 and a single scalar boson with spin 0.

- Gluon: The force mediator for the strong force, ensuring quarks remain bound within larger particles like protons and neutrons.
- Photon: This massless boson mediates the electromagnetic force, governing interactions between charged particles.
- W and Z Bosons: These bosons mediate the weak force. The W boson can have an electromagnetic charge of either -1 or +1, while the Z boson is neutral. Both W and Z bosons are massive, distinguishing them from the massless photons.
- Higgs Boson: This scalar boson is unique. It's associated with the Higgs field, which imparts mass to fermions, except for neutrinos. The process, involving spontaneous symmetry breaking, is intricate and has been a significant focus in particle physics, especially after the Higgs boson's discovery in 2012.

2.2 Neutrinos

Neutrinos are elementary particles that play a crucial role in the Standard Model of particle physics. They belong to the Lepton family and are unique due to their neutral charge. This means that they don't interact via the electromagnetic force, making them incredibly elusive. In the standard model, neutrinos were initially thought to be massless.

However, the observation of neutrino oscillations upended this assumption. Such oscillations can only occur if neutrinos possess mass, signifying a departure from the Standard Model's predictions and hinting at its limitations in fully grasping the nuances of neutrino physics.

The Standard Model posits the existence of three neutrino flavours, each linked to one of the three charged leptons: electron, muon, and tau. Yet, it falls short of explaining the observed mass differences among these neutrino types. This gap between theoretical prediction and empirical evidence points towards the potential of undiscovered physics beyond the current scope of the Standard Model.

2.2.1 Neutrino Interaction Types

Neutrinos, being charge-neutral and colourless, interact infrequently, exclusively through the weak force. There are two primary types of neutrino interactions:

- *Neutral Current (NC) Interaction*: Mediated by the neutral Z boson, the interaction is represented as $\nu_l + q \rightarrow \nu_l + q$, where l is a lepton (e.g., e, μ, τ) and q is a quark.
- *Charged Current (CC) Interaction*: Mediated by the charged $W^{\pm}$boson, the interaction is represented as $\nu_l + q_{-\frac{1}{3}} \rightarrow l + q_{-\frac{2}{3}}$. The change from $q_{-\frac{1}{3}}$ to a $q_{+\frac{2}{3}}$ is to not violate charge conservation.

Neutrinos, despite being elusive particles, interact with matter through the weak nuclear force. Their interactions can be broadly categorized into three main types, each dominant at different energy regimes:

- **Quasi-elastic Scattering (QE)**: This interaction is predominant at energies below 1 GeV. In this process, neutrinos scatter off an entire nucleon. When a neutrino interacts with a nucleon by exchanging a charged boson and emitting a charged lepton, it's termed "quasi-elastic scattering". If the neutrino scatters with the nucleon in a neutral current process, it's termed "elastic scattering". This interaction is characterized by the absence of pions in the final state and is the primary interaction type for neutrino energies below 1 GeV.
- **Resonance Production (RES)**: This interaction becomes significant for neutrino energies between 1 and 10 GeV. Here, the neutrino excites the nucleon to a resonance state, which subsequently decays into various mesonic final states. The resonance production is characterized by the presence of a single pion in the final state.
- **Deep Inelastic Scattering (DIS)**: This interaction type becomes dominant for neutrino energies above 10 GeV. In DIS, the neutrino scatters with an individual quark constituent of the nucleon. This leads to the nucleon disintegrating, producing a hadronic shower. The process is characterized by the exchange of a W or Z boson between the neutrino and a quark inside the nucleon. As the energy of the neutrino increases, the probability of DIS interactions also rises [3].

The contributions of each of these scattering processes, for both neutrinos and antineutrinos, are depicted in Figure 2.2. It's important to highlight that a high cross-section doesn't necessarily equate to a higher number of events detected by IceCube. Events are only recorded if they emit sufficient light to activate IceCube's triggers. While QE events typically have too low energy to be observed in IceCube, RES, despite having a cross-section comparable to DIS at around 10 GeV, isn't as efficient as DIS in emitting light. This is because DIS results in a more noticeable hadronic cascade.

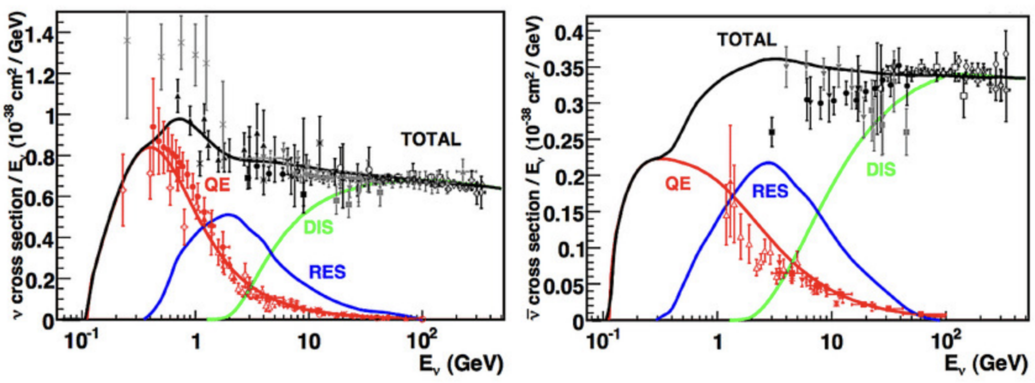


FIGURE 2.2: Neutrino (left) and antineutrino (right) cross-sections for the interaction type QE, RES, DIS and the total as a function of the neutrino energy. figure from [4].

2.2.2 Neutrino Oscillation

Neutrinos, intriguingly enigmatic particles within the standard model, have consistently defied initial predictions. For instance, the standard model originally posited that neutrinos were massless, only for subsequent discoveries to reveal they possess mass.

Neutrinos manifest in three distinct flavour states: ν_e, ν_μ, ν_τ, named after the charged leptons they produce during charged current interactions. However, the mass states of neutrinos, represented as (ν_1, ν_2, ν_3) do not align directly with these flavour states. This misalignment between flavour and mass states results in the fascinating phenomenon of neutrino flavour oscillation. As a neutrino travels through space, its flavour state oscillates among the three flavours.

The relationship between the flavour states and the mass states can be expressed using the Pontecorvo-Maki-Nakagawa-Sakata (PMNS) matrix, denoted as U and defined by three mixing angles θ_{23}, θ_{13} θ_{12} and a complex phase δ which is responsible for charge-parity (CP) Violation. [5].

$$\begin{pmatrix} \nu_e(x) \\ \nu_\mu(x) \\ \nu_\tau(x) \end{pmatrix} = U^* \begin{pmatrix} \nu_1(x) \\ \nu_2(x) \\ \nu_3(x) \end{pmatrix} \tag{2.1}$$

$$U = \begin{pmatrix} c_{12}c_{13} & s_{12}c_{13} & s_{13}e^{-i\delta_{CP}} \\ -s_{12}c_{23} - c_{12}s_{13}s_{23}e^{i\delta_{CP}} & c_{12}c_{23} - s_{12}s_{13}s_{23}e^{i\delta_{CP}} & c_{13}s_{23} \\ s_{12}s_{23} - c_{12}s_{13}c_{23}e^{i\delta_{CP}} & -c_{12}c_{23} - s_{12}s_{13}c_{23}e^{i\delta_{CP}} & c_{13}c_{23} \end{pmatrix} \tag{2.2}$$

where $c_{ij} = \cos\theta_{ij}$ and $s_{ij} = \sin\theta_{ij}$[5].

The PMNS matrix can be decomposed into a product of three matrices, each resembling a rotation matrix. This decomposition is essential because no single experiment can determine all components of the matrix. Thus, the matrix is divided into sections, each named after the type of experiment required to discern it. These sections are referred to as long-baseline/atmospheric, reactor, and solar experiments.

$$U = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_{23} & s_{23} \\ 0 & -s_{23} & c_{23} \end{pmatrix} \begin{pmatrix} c_{13} & 0 & s_{13}e^{-i\delta_{CP}} \\ 0 & 1 & 0 \\ -s_{13}e^{i\delta_{CP}} & 0 & c_{13} \end{pmatrix} \begin{pmatrix} c_{12} & s_{12} & 0 \\ -s_{12} & c_{12} & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.3}$$

To further understand the oscillation phenomenon, consider the simplified oscillation probability equation for two flavours. The full derivations can be found here[6]

$$p(\nu_\alpha \to \nu_\beta) = \sin^2(2\theta)\sin^2(\frac{\Delta m^2 L}{4E}) \tag{2.4}$$

Here, E represents the neutrino energy, and L denotes the oscillation length. This equation highlights that the oscillation probabilities in vacuum depend on the sine squared of the mass difference, implying that solely measuring the oscillation probabilities of neutrinos that have only propagated through a vacuum won't resolve the neutrino mass ordering problem since we won't be able to tell the sign of the mass difference.

2.2.3 Neutrino Mass Ordering

The phenomenon of neutrino oscillations provides compelling evidence that neutrinos possess non-zero masses. However, while experiments have been able to measure the differences in squared masses between the neutrino mass eigenstates, and the experiments Super-Kamiokande experiment[7] in Japan and the Sudbury Neutrino Observatory (SNO)[8] in Canada have been able to determine the sign of the Δm^2_{21} to positive the sign of the last sign of the mass difference of Δm^2_{31} still remains unknown. This leads to a fundamental question in neutrino physics: the problem of neutrino mass ordering.

There are two possible configurations for the ordering of the neutrino masses:

Normal Ordering (NO): In this configuration, ν_1 has the lightest mass, followed by ν_2 and then ν_3. Inverted Ordering (IO): Here, ν_3 has the lightest mass, followed by ν_1 and then ν_2. The determination of the neutrino mass ordering is crucial for understanding the nature of neutrinos and has implications for theories beyond the Standard Model of particle physics.

In the presence of matter, such as when neutrinos traverse the Earth, the oscillation probabilities can be further modified due to coherent forward scattering, known as the Mikheyev–Smirnov–Wolfenstein (MSW) effect. This effect can enhance or suppress the probability of a neutrino transitioning from one flavor to another which allows us to potentially distinguish between NO and IO if the oscillation patterns for neutrinos and anti-neutrinos can be discerned.

At first glance, coherent forward scattering might seem inconsequential. This process involves an electron neutrino scattering off an electron, represented by the equation:

$$\nu_e + e \rightarrow \nu_e + e$$

The significance of this interaction lies in its impact on the neutrino's state. As a neutrino moves through space, it exists in a superposition of all flavour states, determined by their oscillation probabilities. However, since coherent forward scattering exclusively affects electron neutrinos, the interaction causes the neutrino's superposition to collapse, effectively "resetting" it to an electron neutrino state. This alteration influences the eventual flavour state probabilities when the neutrino is detected by instruments like IceCube.

In the case of *Normal Ordering*:

- Neutrinos travelling through matter experience enhanced oscillation probabilities, especially for the transition from muon neutrinos to electron neutrinos.
- Anti-neutrinos, on the other hand, might see suppressed oscillation probabilities for the same transitions.

For *Inverted Ordering*:

- The oscillation probabilities for neutrinos and anti-neutrinos are effectively swapped compared to the normal ordering scenario.

The differences in oscillation probabilities between neutrinos and anti-neutrinos, and between the two possible mass orderings, can be observed in experiments that detect neutrinos after they've traveled long distances through matter, such as the Earth. By comparing the observed oscillation probabilities with theoretical predictions for both orderings, experiments aim to determine the true mass ordering of neutrinos.

In Figure 2.3, the oscillation probabilities for neutrinos and anti-neutrinos are plotted for both normal and inverted mass orderings. The distinct patterns observed for each scenario provide a roadmap for experimental efforts to resolve the neutrino mass ordering problem.

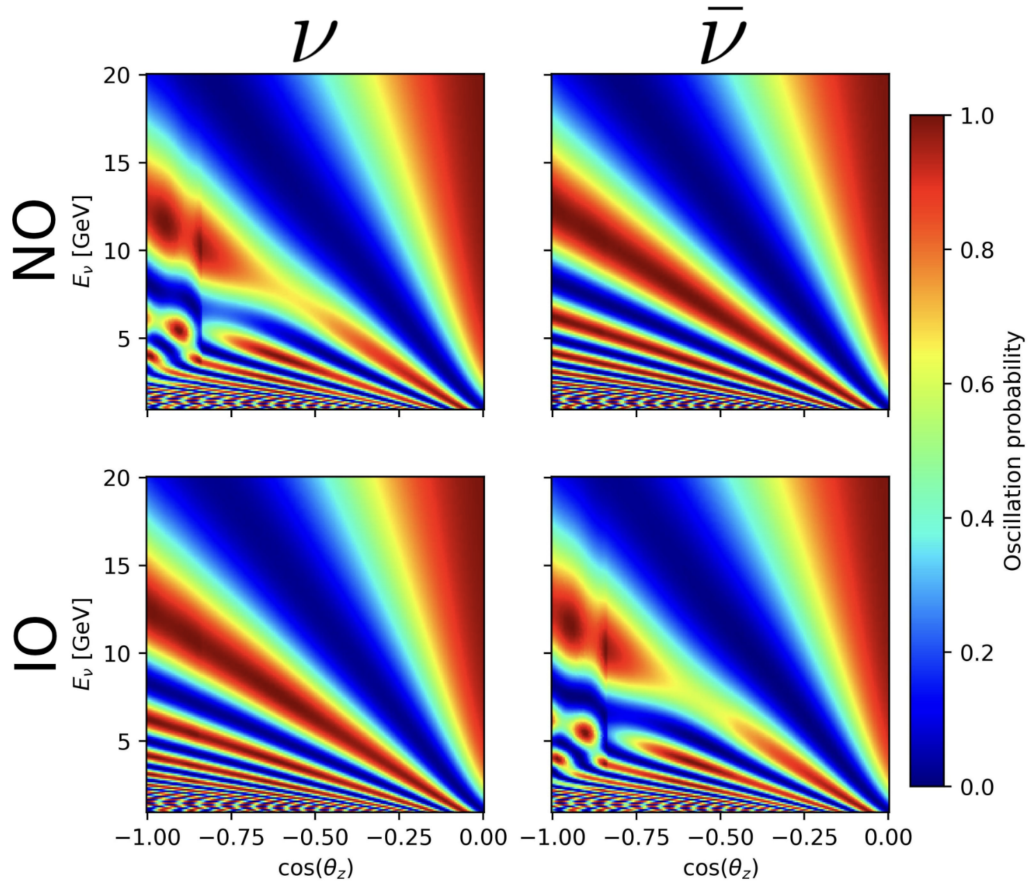


FIGURE 2.3: The matter effects can be seen in the neutrino normal ordering case and the anti-neutrino inverse ordering case for energies below 15 GeV and $\cos\theta_z$ below -0.8. kindly borrowed from [9].

2.3 Discerning Neutrinos from Anti-neutrinos using inelasticity

Distinguishing between neutrinos and anti-neutrinos has always been a challenge due to their charge-neutral nature. In the context of neutrino oscillations and their implications for the Standard Model, this distinction becomes even more crucial. The slight differences in the oscillation patterns between neutrinos and anti-neutrinos, especially in regions affected by the Mikheyev–Smirnov–Wolfenstein (MSW) effect, provide a potential pathway to determine the neutrino mass ordering.

Neutrinos and anti-neutrinos, while being mirror images of each other in many respects, exhibit certain differences when interacting with matter. These differences stem from the asymmetry in matter composition (more matter than anti-matter) and the intricacies of the weak force. As neutrinos traverse through matter, they experience coherent forward scattering with electrons, which can enhance or suppress oscillation probabilities. This effect, combined with the subtle differences in oscillation patterns between neutrinos and anti-neutrinos, allows for the possibility of distinguishing between the two particles.

2.3.1 Cerenkov radiation

Cherenkov radiation is a phenomenon that arises when charged particles travel through a dielectric medium at speeds exceeding the phase velocity of light in that medium [3]. This rapid motion of charged particles polarizes the stationary molecules of the medium. As these molecules return to their ground states, they emit energy in the form of light, radiating spherical waves. When the particle's velocity surpasses the phase speed of light in the medium, these spherical waves undergo constructive interference, resulting in the emission of a cone of light, with the particle's path defining the cone's axis, as illustrated in Figure 2.4. This process is analogous to the sonic boom created when a jet breaks the sound barrier, but in this case, it's for light instead of sound.

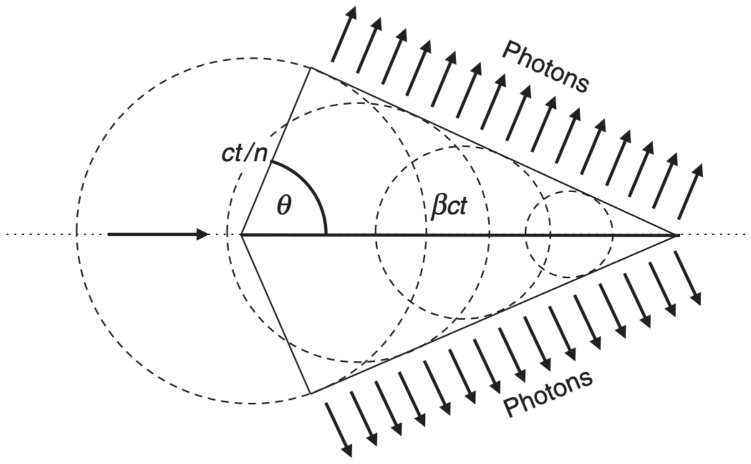


FIGURE 2.4: Cerenkov radiation [3]

The angle, θ, at which this radiation is emitted is dependent on the particle's speed, v_p, and the speed of light in the medium,$v_c \frac{c}{n}$, where c is the speed of light in a vacuum and n is the refractive index of the medium. The relationship between the angle of emission and the particle's speed is given by:

$$\cos\theta = \frac{1}{n\beta} \tag{2.5}$$

Where β represents the fraction of the speed of light at which the particle is travelling. Cherenkov radiation serves as the primary detection mechanism for particles in IceCube because ice, being a dielectric medium, enables the emission of this type of radiation. Notably, Cherenkov radiation manifests similarly for both positively and negatively charged particles. This symmetry in radiation patterns obscures the identification of particle charge, complicating the distinction between neutrinos and anti-neutrinos in IceCube. Consequently, alternative methods become essential to differentiate between these two types of particles.

2.3.2 Inelasticity

The central theme of this thesis revolves around the utilization of inelasticity as a pivotal kinematic variable in particle physics. Consequently, the subsequent sections will delve into how various physics properties interrelate with the capability to resolve inelasticity. A notable application of this is the differentiation between neutrinos and anti-neutrinos based on their distinct inelasticity distributions. Inelasticity is defined as [3]:

$$y := \frac{p_2 \times q}{p_2 \times p_1} \tag{2.6}$$

In the frame of reference where p_1, p_2 and p_3, are the four-momentum of the particles involved in the interaction. q represents the four-momentum transfer from the incident particle (often a lepton) to the target particle (often a nucleon). This is visualized in Figure 2.5:

$$\begin{aligned} p_1 &= (E_1, 0, 0, E_1) \\ p_2 &= (m_2, 0, 0, 0) \\ p_3 &= (E_3, E_3 \sin(\theta), 0, E_3 \cos(\theta)) \\ q &= (E_1 - E_3, p_1 - p_2) \end{aligned}$$

From the above definitions, we can derive:

$$y = \frac{m_2(E_1 - E_3)}{m_2 E_1} = 1 - \frac{E_3}{E_1} = 1 - \frac{E_{track}}{E_{total}} \tag{2.7}$$

The practical reconstruction of inelasticity often requires determining the energy of the initial and outgoing particles, denoted as E_{total} and E_{track} respectively, with both values established in the detector's reference frame. The term "track particle" pertains to the resulting particle post-interaction, the identity of which varies based on the nature of the interaction and the initial particle.

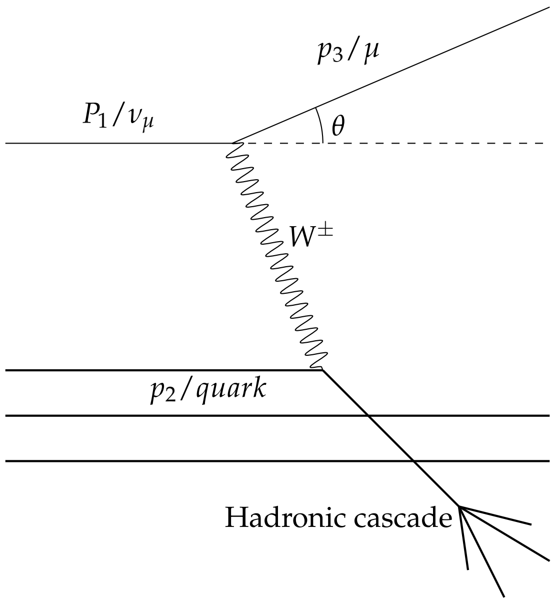


FIGURE 2.5: Deep inelastic scattering of a muon neutrino off a quark.

Being a Lorentz-invariant quantity, inelasticity remains unchanged across different reference frames. When considering a nucleon initially at rest, inelasticity represents the fraction of energy the neutrino transfers to the nucleon.

For Neutral Current (NC) interactions, inelasticity reconstruction presents a challenge as the track particle in this case is a neutrino, which leaves the detector without generating a discernible trace. The only remnant is the hadronic cascade (HC), produced from the energy deposited in the quark. Hence, determining inelasticity for NC interactions is not typically feasible.

Charged current interaction is a more promising candidate. These interactions yield a charged lepton as the track particle, the type of which corresponds to the initiating neutrino. However, not all charged leptons are equally detectable in IceCube due to their unique interaction signatures.

- *Electron Neutrino*: Produces an electron that scatters rapidly creating an electromagnetic cascade that rarely escapes the Hadronic cascade making it challenging to reconstruct.
- *Muon Neutrino*: Produces a muon, leaving a detectable track of Cherenkov radiation, making it the ideal candidate for inelasticity reconstruction.
- *Tau Neutrino*: Produces a tau that decays too swiftly for track detection. However, other kinds of signatures have been observed [10]

Given these considerations, this thesis will focus solely on charged current muon neutrinos when developing inelasticity reconstruction models.

While Muons travel significantly longer in ice than both the tau and the electrons they don't travel independently as ice like any material has a stopping power. In Figure 2.6 The stopping power for a μ^+ in copper is plotted which is approximately the same for ice once adjusted for density.

In the Antarctic ice, the stopping power for a muon translates to an average distance of roughly 4.5 meters per GeV. This characteristic influences the observable signature of certain interactions. For instance, in low-energy charge current muon neutrino interactions, the muon might lack the energy to traverse a distance sufficient to differentiate from the signature of the hadronic cascade. Conversely, in high-energy interactions, the muon may possess enough energy to exit the detector entirely, complicating the reconstruction process.

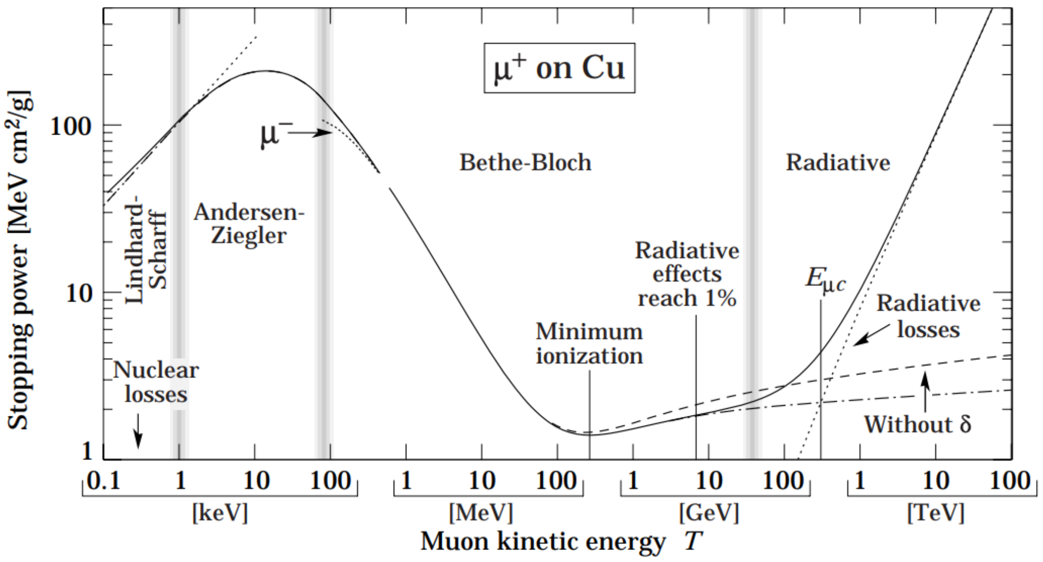


FIGURE 2.6: Stopping power for a μ^+ in copper as a function of energy of the muon.[11].

2.3.3 Neutrino and Anti-neutrino Inelasticity Distributions

In distinguishing between neutrinos and anti-neutrinos, one of the key features to consider is the difference in their inelasticity distributions during charged current (CC) interactions. This distinction arises from the nature of the weak force, which acts specifically on left-handed particles and right-handed antiparticles.

There are four possible interaction scenarios, which can be grouped into two main categories based on the total spin:

- *Spin-0 Interactions*
  - Neutrino-quark interaction
  - Anti-neutrino-anti-quark interaction
- *Spin-1 Interactions*
  - Neutrino-anti-quark interaction
  - Anti-neutrino-quark interaction

Instead of delving into the calculations detailing how spin correlates with the resulting inelasticity distribution, I'll provide a more qualitative explanation highlighting the differences in inelasticity distribution between neutrinos and anti-neutrinos.

Let's consider the scenario of a spin-zero interaction. In this context, the recoil angle, denoted as θ — which represents the angle between the incoming neutrino and the outgoing charged lepton (in our case, a muon) — is distributed uniformly across all possible angles. This uniform distribution can be justified by recognizing that inelasticity remains Lorentz invariant, implying it retains its value across all valid reference frames. For our analysis, we'll adopt the center-of-mass reference frame of the neutrino and the quark, where the total momentum is zero. Given that both momentum and angular momentum are zero (owing to the zero spin), there's no inherent parameter that could introduce a preferred direction. Consequently, the recoil angle must be isotropic and uniformly distributed in terms of $cos(\theta^)$. This behaviour can be seen in figure 2.7, in the leftmost and rightmost plots of the third row, corresponding to the neutrino-quark and anti-neutrino-anti-quark interactions, respectively.

In the case of spin-1 interactions, the total angular momentum isn't zero, which introduces the possibility of a favoured recoil angle. Explaining the existence of this preferred angle is slightly intricate. The rationale is that the spins in the final state are more inclined to align with their initial configuration prior to the interaction. This means that the spin of the outgoing muon, resulting from a muon neutrino interaction, is more likely to have a smaller recoil angle, leading to a non-uniform angle distribution.

This specific behaviour can be observed in figure 2.7. The two innermost plots in the third row represent the neutrino-anti-quark and anti-neutrino-quark interactions, respectively, and they clearly show the skewed recoil angle distribution.

The emphasis on the recoil angle distribution is crucial because it directly correlates with the inelasticity distribution. A smaller recoil angle means a larger portion of the neutrino's energy is transferred to the outgoing charged lepton. Given that inelasticity is defined as $1 - \frac{E_{track}}{E_{total}}$ a smaller recoil angle results in smaller inelasticity.

While these interactions are mirrored for neutrinos and anti-neutrinos, in that there is a spin-0 and a spin-1 interaction for both neutrinos and anti-neutrinos, the actual resulting inelasticity distributions we see differ. This discrepancy is attributed to the lower probability of encountering an anti-quark compared to a quark as the target in the Antarctic ice. Thus, the neutrino inelasticity distribution is primarily skewed with a minor contribution from the flat distribution. In contrast, the anti-neutrino inelasticity is predominantly flat with a slightly skewed component.

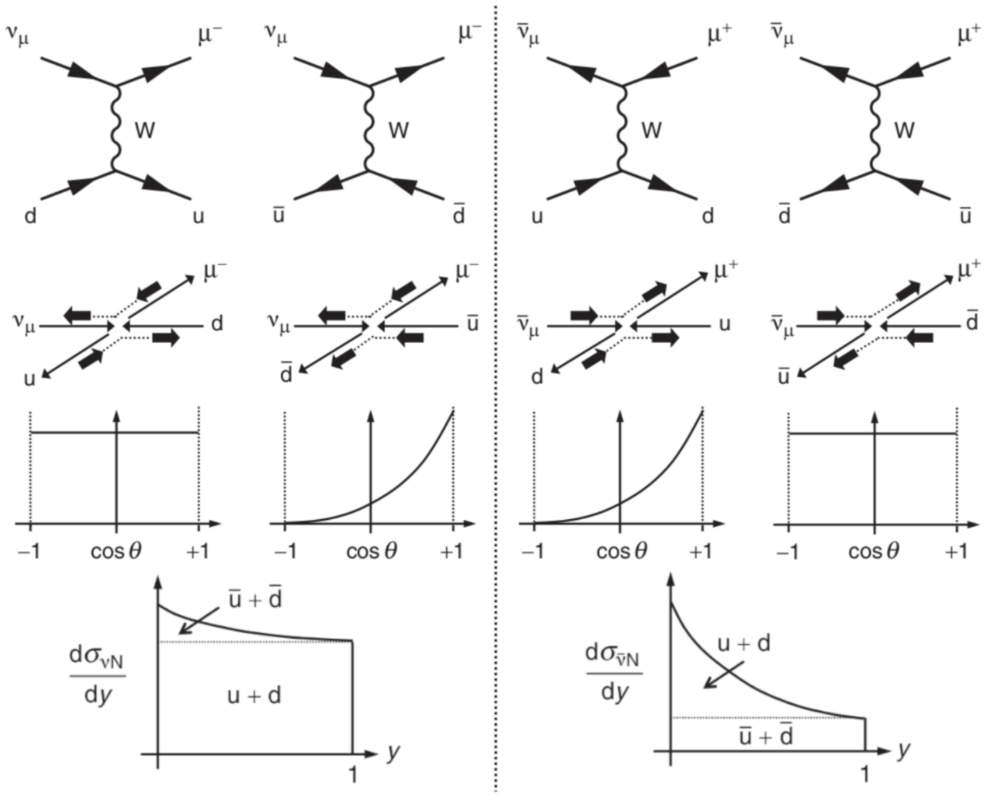


FIGURE 2.7: On the left we see the neutrino interaction with first a quark and then an anti-quark below is the resulting recoil angle distribution and at the bottom the resulting inelasticity distribution. This is mirrored on the right but with an anti-neutrino instead. [9].

2.4 Improving IceCube NMO sensitivity

Distinguishing between neutrinos and anti-neutrinos offers a significant advantage in the quest to determine the Neutrino Mass Ordering (NMO) within the context of the IceCube Upgrade:

Distinct Oscillation Probabilities: Neutrinos and anti-neutrinos possess different oscillation probabilities. This distinction in how ν transitions between various flavors, in comparison to $\bar{\nu}$, becomes particularly evident when considering the influence of matter effects during their transit through the Earth.

Matter Interactions and MSW Effect: As neutrinos traverse the Earth, they encounter interactions stemming from the weak force. These interactions differ for neutrinos and anti-neutrinos. Specifically, the electrons present in matter have differential effects on neutrinos and anti-neutrinos, leading to the Mikheyev–Smirnov–Wolfenstein (MSW) effect. This phenomenon induces variations in the oscillation patterns of neutrinos and anti-neutrinos, with certain patterns becoming more prominent depending on the mass ordering (NO or IO).

Enhanced Sensitivity to NMO: The distinct oscillation patterns for neutrinos and anti-neutrinos, once separated, provide two complementary datasets. This duality accentuates the sensitivity to discrepancies between the NO and IO scenarios, facilitating a clearer discernment of the authentic mass ordering.

Statistical Robustness: By achieving a separation, two data streams are acquired instead of a singular one. This amplifies the statistical strength of any analyses performed to ascertain the NMO. A more extensive dataset ensures that the conclusions derived are both reliable and robust.

In essence, the inherent and differential properties of neutrinos and anti-neutrinos, especially concerning their oscillatory behaviours and matter interactions, underscore the importance of their separation in enhancing the sensitivity to the Neutrino Mass Ordering within IceCube. This distinctiveness offers a more comprehensive and detailed insight into neutrino oscillations, aiding in the precise determination of whether the mass ordering aligns with the normal or inverted paradigm.

For a more exhaustive explanation, I recommend reading Marc Jacquart's thesis, which delves deeper into this topic[9].

Chapter 3

The IceCube Neutrino Observatory

3.1 The IceCube Neutrino Observatory

Overview

Situated at the South Pole, the IceCube Neutrino Observatory spans a vast area of 1 cubic kilometer. This monumental structure operates based on the Cherenkov detection principle. Constructed over a period from 2004 to 2011, it has been operational since 2011. The primary mission of IceCube is to detect astrophysical neutrinos and delve into the mysteries of their sources, as well as to conduct in-depth neutrino oscillation studies. An overview of the detector's geometry can be seen in Figure 3.1.

3.1.1 IceCube Array

The heart of the observatory is the IceCube array, which consists of a staggering 5484 Digital Optical Modules (DOMs) distributed across 86 vertical strings. These strings are strategically placed between depths of 1450m and 2450m. Each DOM is equipped with Photomultiplier Tubes (PMTs), designed to detect the Cherenkov photons emitted when neutrinos interact with the ice. The initial configuration of the observatory featured 78 strings arranged in a hexagonal formation, ensuring optimal coverage and detection capabilities.

3.1.2 DeepCore

DeepCore serves as an extension to the main IceCube array, specifically designed to detect lower-energy neutrinos, particularly those below 100 GeV. This sub-detector houses 480 DOMs spread across eight new strings, all of which are positioned below the "dust layer" to minimize background noise and enhance detection capabilities.

3.2 The detector

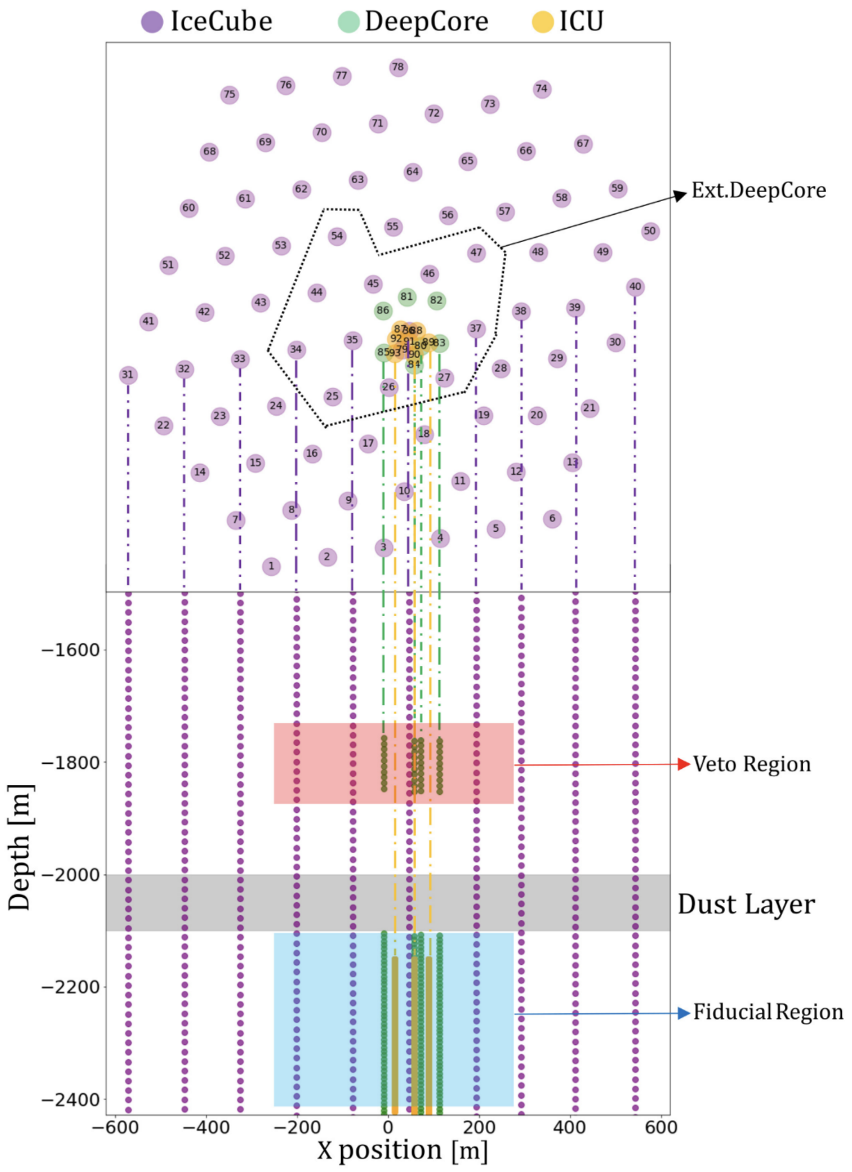


FIGURE 3.1: The detector with combined top view and a cross-section. Kindly borrowed from [12]

3.2.1 IceCube Upgrade

To further its mission and enhance its capabilities, an upgrade project for IceCube and DeepCore is underway. This ambitious project aims to deploy 700 new or upgraded sensors, arranged in a denser configuration to improve detection accuracy. The upgrade introduces several new DOM types, each with its unique features:

- PDOM: Resembling the original IceCube DOM, the PDOM has a single downward-facing PMT. However, it boasts enhanced electronics, improving its efficiency.
- D-Egg: This innovative design features two 8″ PMTs, with one facing upwards and the other downwards. This dual-direction approach significantly

improves the resolution for light coming from above, proving invaluable in detecting up-going neutrinos.

- mDOM: A marvel in design, the mDOM houses 24×3″ PMTs. This configuration is designed to capture light from all possible directions, substantially enhancing the detector's sensitivity.

These new DOMs, showcased in Figure 3.2, are not just upgrades in technology but are pivotal in enhancing the directional sensitivity of the observatory. This leads to a marked improvement in the rejection of atmospheric muons and background interference. The result is a more refined and precise neutrino detection mechanism, especially potent at lower energies.

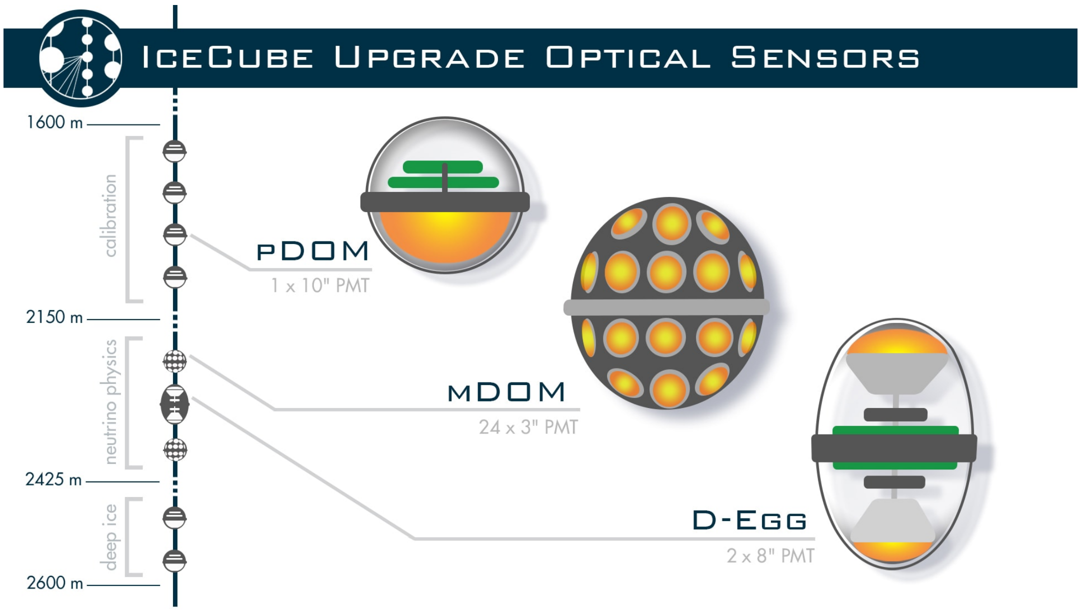


FIGURE 3.2: The new upgrade strings with the upgrade DOMS[13].

3.3 Data Collection and Processing in IceCube

3.3.1 From Cherenkov Radiation to Signal Reception

The journey from the occurrence of Cherenkov radiation to the signal received at the IceCube Counting Lab (ICL) is intricate. Each Digital Optical Module (DOM) is equipped with Photo Multiplier Tubes (PMTs), devices designed to convert photons into detectable currents.

3.3.2 Digitization at the DOM Level

Due to technical constraints, the raw PMT output is not directly transmitted from the DOMs to the surface. Instead, it undergoes digitization at the DOM level. Two types of digitizations are employed:

- Fast Analog-to-Digital-Converter (FADC): Offers data over longer timeframes but with less precision.
- Analog Transient Waveform Digitisers (ATWD): Provides shorter, more precise time windows but comes with a dead time after recording a pulse. To mitigate

this, two ATWDs are housed within a DOM, with one on standby while the other processes data.

The choice of digitization method hinges on specific criteria, including the discriminator threshold, which is the minimum voltage from the PMT that prompts a DOM to record waveforms. Another crucial factor is the Local Coincidence Criteria, a mechanism where each DOM connects to its neighbouring DOMs through a coincidence cable. When a DOM meets the discriminator threshold, it sends a signal through the cable, and if a neighbouring DOM also triggers within a $1 \pm \mu s$ timespan, certain actions are taken[14].

3.3.3 Data Transmission and Pulse Map Creation

Two scenarios can arise based on the aforementioned criteria:

- Soft Local Coincidence (SLC): Triggered when only the discriminator threshold is met. Here, only three bins around the peak FADC waveform are sent to the surface.
- Hard Local Coincidence (HLC): Activated when both the discriminator threshold and the local coincidence criteria are met. In this case, the complete waveform from both ATWDs (if applicable) and the FADC are transmitted to the surface.

Upon reaching the surface, the ICL is tasked with converting this waveform data into usable pulse maps. An algorithm estimates the number of photoelectrons that produced the signal and their arrival time, resulting in a pulse map comprising discrete DOM hits with associated charge and time.

3.3.4 Noise Cleaning Algorithms

To ensure data integrity, several noise cleaning algorithms are employed. Simple ones like Seeded-Radius-Timing examine SLC at specific radii. However, recent advancements have ushered in machine learning-based cleaning. The model, termed DynEdge, will be discussed in detail in section 4.4.1. Trained on simulated data, DynEdge offers an unparalleled capability to produce clean events.

3.3.5 Challenges in Noise Flagging

Correctly flagging a pulse as noise is non-trivial. A pulse may arise from a noise photon combined with a photon from an interaction. While individually they might not exceed the discriminator threshold, collectively they can. Determining such pulses as noise hits poses a challenge. However, for lower energy events, the focus of this thesis, the odds of this occurring are negligible.

3.4 Simulated Data and Its Role

3.4.1 Simulation Programs and Event Types

Simulated data isn't the output of a singular program. Instead, a suite of programs, each tailored for specific event types, produces the data. These include GENIE for neutrino events, MuonGun for muon events, and Vuvuzela for noise events.

3.4.2 Data Storage and Conversion

The resultant simulated events are stored in I3 files. However, for the convenience of training machine learning models, these I3 files are converted into SQLite databases, containing tables for the pulse map and truth labels, such as energy and inelasticity. The rationale behind this conversion is twofold: reduced access times and ease in selecting specific event types for training, as showcased in this thesis where the focus is on muon neutrino charge current events.

3.5 My Dataset and Its Significance

3.5.1 Dataset Composition

My dataset comprises approximately 2 million muon neutrino charge current events. The division is as follows: 80% for training, 10% for validation, and the remaining 10% for testing.

3.5.2 Comparison to Real-World Data Collection

Drawing from the work of Jorge, a previous master's student, post-filtering event rates for muon neutrinos stand at 3.75 mHz [12]. This translates to an annual count of 118k. Therefore, to accumulate real-world events equivalent to my test sample, approximately 1.68 years would be required. However, the actual time might be shorter, given that many of my neutrinos wouldn't pass through the myriad filters.

3.5.3 Physical Weights and Flux

In simulations, the zenith and azimuth angles are distributed isotropically. This means that there is no preferred direction in which these angles appear. Typically, the energy distribution adheres to the relationship $\frac{d\phi}{dE} \propto E^{-2}$, where ϕ represents the flux and E is the energy[9].

However, this simulated distribution differs from what we observe in the real world. As highlighted in section 2.2.3, the Earth influences neutrino oscillation patterns. Depending on the zenith angle—which correlates to the distance neutrinos have traversed through the Earth—and the energy of the neutrino, certain neutrino types become more or less probable.

While it might seem logical to simulate events based on the distribution expected from neutrino oscillations, our primary objective is to maintain unbiased models. Simulating based on expected oscillation patterns risks our models inadvertently learning these patterns. Such a scenario would lead to biased results that merely reflect our initial expectations. Ideally, a model should be ignorant of neutrino oscillations. This ensures that any oscillation patterns we observe post-reconstruction are genuine and not artifacts from the model's training data.

Moreover, there's another significant consideration. The oscillation patterns are subtly influenced by whether neutrinos follow a normal ordering (NO) or inverse ordering (IO). If our goal is to differentiate between NO and IO using our reconstructions, training on weighted data becomes unsuitable.

In simulations, the zenith and azimuth angles are distributed isotropically. This indicates that there is no favoured direction for these angles. The energy distribution typically adheres to the relationship $\frac{d\phi}{dE} \propto E^{-2}$, where ϕ denotes the flux and E denotes the energy.

However, the distribution we obtain from simulations does not align with real-world observations. As discussed in section 2.2.3, the Earth's properties influence neutrino oscillation patterns. The likelihood of certain neutrino types appearing varies based on both the zenith angle, which corresponds to the distance neutrinos have travelled through the Earth and the neutrino's energy.

One might assume that simulating events based on expected neutrino oscillation patterns is logical. However, our primary goal is to preserve unbiased models. If we were to simulate based on anticipated oscillation patterns, our models might inadvertently incorporate these patterns. Such an occurrence would produce biased outcomes, merely mirroring our initial predictions. For optimal results, our model should remain unaware of neutrino oscillations. This approach guarantees that any oscillation patterns detected after the reconstruction process are genuine and not merely remnants of the model's training data.

Additionally, it's crucial to note that oscillation patterns are slightly affected by whether neutrinos adhere to a normal ordering (NO) or an inverse ordering (IO). If our intention is to distinguish between NO and IO through our reconstructions, using weighted data for training becomes inappropriate.

3.5.4 Data Storage and Conversion

The resultant simulated events are stored in IceCube collaboration-specific I3 files. However, for the convenience of training machine learning models, these I3 files are converted into SQLite databases, containing tables for the pulse map and truth labels, such as energy and inelasticity. The rationale behind this conversion is twofold: reduced access times and ease in selecting specific event types for training, as showcased in this thesis where the focus is on muon neutrino charge current events.

Database Variables

The SQLite database contains a variety of variables detailing the characteristics of the events, as outlined in Table 3.1. While many of these variables offer similar or redundant information, they provide a comprehensive view of the event. It's worth noting that all variables, other than time and charge, could be reduced to a single variable without loss of information since the detector doesn't change from event to event. This variable could simply be a unique ID assigned to every single PMT in the detector. The reason for not doing this builds on the idea that the models are able to use the physical position of the DOM to ease the learning rather than having to learn every correlations between DOMs simply from the ID.

Variables	Description
`dom_time`	The time the DOM was hit relative to the first hit in the event
`charge`	The charge that was recorded by the PMT in the DOM, which is related to the number of photons it was hit by
`dom_x`	X position of the DOM
`dom_y`	Y position of the DOM
`dom_z`	Z position of the DOM
`string`	The number of the string on which the DOM is positioned
`dom_number`	Number of the DOM on the string
`dom_type`	Used to distinguish between pDOM's, upgrade pDOM's, mDOM's and D-EGG's.
`rde`	Relative DOM Efficiency: used to distinguish between the more sensitive DeepCore pDOM's and the rest of the pDOM's
`pmt_number`	Number the activated PMT on the DOM
`pmt_dir_x`	X direction of the PMT
`pmt_dir_y`	y direction of the PMT
`pmt_dir_z`	z direction of the PMT
`pmt_area`	The area of the PMT

TABLE 3.1

Chapter 4

Machine Learning

Machine learning, an integral subset of artificial intelligence, bestows upon computers the capability to learn from data, refine their algorithms, and consequently make decisions or predictions without being explicitly programmed for specific tasks. Central to this domain are algorithms and models tailored to identify patterns, derive insights, and generalize from pre-existing data.

Within the vast realm of machine learning, there exist diverse categories:

- Supervised Learning: Where data comes labelled, and the model learns to predict outcomes based on input features.
- Unsupervised Learning: Here, data isn't labelled. The model discerns underlying structures or patterns, such as clustering.
- Reinforcement Learning: In this paradigm, an agent learns by interacting with an environment, and receiving feedback in the form of rewards or penalties.

Despite the broad spectrum of machine learning categories, it's essential to note that the boundaries separating them aren't always clear-cut. Furthermore, no single algorithm or model architecture dominates the field. The variety of algorithms and models, ranging from linear regression and K-nearest Neighbors to support vector machines and neural networks, is astounding.

In this thesis, the spotlight will be on supervised learning, particularly harnessing the power of neural networks.

4.1 Neural Networks

Neural networks are computational models inspired by the way biological neural systems process information. They consist of layers of interconnected nodes or "neurons".

4.1.1 Architecture

A standard feed-forward neural network comprises three primary layers:

1. Input Layer
2. Hidden Layer(s)
3. Output Layer

Each connection (from neuron i to neuron j) has an associated weight w_{ij}. In addition to weights, each neuron has a bias term (b), which adjusts the output along with the weighted sum of inputs.

Mathematically, the output o_j of a neuron j in the next layer is given by:

$$o_j = f\left(\sum_i w_{ij} o_i + b_j\right)$$

Where f is the activation function, and the summation runs over all neurons i in the previous layer[15].

4.1.2 Activation Functions

Activation functions introduce non-linearity into the model, which allows the network to capture complex relationships in the data. Some commonly used activation functions include:

- **Sigmoid**:
$$f(x) = \frac{1}{1+e^{-x}}$$
- **ReLU (Rectified Linear Unit)**:
$$f(x) = \max(0, x)$$
- **Tanh**:
$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$
- **Softmax** (commonly used in the output layer of classification tasks):
$$f(x)_i = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

4.2 Loss Functions

Loss functions serve as the cornerstone in training machine learning models as they define the purpose of the model and which direction the output should be changed. Loss functions quantify the deviation of a model's predictions from the actual data. In supervised learning, particularly regression, the "error" denotes the difference between the true and predicted values.

4.2.1 Mean Squared Error(MSE)

MSE is a widely used loss function in regression tasks. It calculates the square of the difference between the actual and predicted values:

$$Loss_{MSE} = (y_{true} - y_{predicted})^2 \tag{4.1}$$

MSE is particularly robust, penalizing outlier predictions more severely than many other loss functions. It's a foundational loss function, especially in the context of linear regression.

4.2.2 Mean Absolute Error (MAE)

MAE loss is very similar to MSE loss but with an absolute instead of the square.

$$Loss_{MAE} = |y_{true} - y_{predicted}| \tag{4.2}$$

MAE is preferred over MSE when the model overly emphasizes outliers. However, its non-differentiability at 0 can lead to challenges during backpropagation.

4.2.3 LogCosh loss

LogCosh loss is an attempt to amalgamate the strengths of both MSE and MAE. For minor errors, it behaves like x^2 ensuring differentiability at zero. For larger errors, it mirrors $|x|$.

$$Loss_{Log-Cosh} = ln(cosh(y_{true} - y_{predicted})) \tag{4.3}$$

4.2.4 Cross Entropy Loss

Predominantly used in classification tasks, Cross Entropy Loss originates from information theory. It measures the average, number of bits required to discern if a sample belongs to distribution p or q.

I will split it up into two cases dependent on whether p and q are discrete or a continuous distribution **Discrete case**: is the more commonly used way of utilizing cross entropy as it can be used for classification tasks and is, in fact, the most commonly utilized loss function for classification tasks. Given:

- y: True labels (One-hot encoded vector meaning a vector of zeros with a one for the correct class).
- $p(y)$: Predicted probabilities (Softmax output from the model).

The cross-entropy loss for the discrete case is:

$$L(y, p(y)) = -\sum_i y_i log_2(p(y)_i) \tag{4.4}$$

where the sum runs over all classes i. y_i is the true label for the class $i()$ $p(y)_i$ is the predicted probability for class i[16].

Continuous case: is the slightly nicher application of cross-entropy loss. The continuous case is most often used when people want to make regression with uncertainty. just like the discrete case where the model tries to make the predicted distribution look as much as possible as the true distribution. A way of representing the true distribution In the continuous case is instead of having a vector that is zero everywhere except for the index corresponding to the actual true class we use a distribution that is zero everywhere except at the true value that we are trying to regress to. This function is of course the dirac delta function.

given:

- $p(x)$: true probability density function.
- $q(x)$: predicted probability density function.

The cross-entropy loss for the continuous case is:

$$H(q, p) = -\int p(x) log_2(q(x)) dx \tag{4.5}$$

Beta Loss

The beta loss function I'm proposing is a continuous case of cross entropy loss where Instead the model giving a discrete probability distribution is a continuous beta distribution defined by two parameters alpha and beta and defined in the range 0 to 1. The best way to describe a beta distribution is to describe the classic scenario where it would be used: determining the probability of a weighted coin turning up heads given you have seen n successes(heads) and m fails(tails). The beta distribution is the solution to that problem. It assigns a probability to all possible weightings parameterized an α and a β where α is the number of successes +1 and β is the number of fails + 1. Something the example fails to explain is that alpha and beta don't have to be a natural number, They just need to be larger than 0.

Following from equation 4.5 $p(x)$ is the "truth" distribution which in the continuous case is a dirac delta function centered at y_{true}. and q is simply the predicted beta distribution defined by the predicted α and β. The beta loss functions is:

$$\begin{aligned} H(\delta(x - y_{\text{true}}), \beta(\alpha, \beta)) &= -\int \delta(x - y_{\text{true}}) \log_2(\beta(\alpha, \beta))\, dx \\ &= -\log_2\left(\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\right) y_{\text{true}}^{\alpha-1}(1 - y_{\text{true}}^{\beta-1}) \end{aligned} \tag{4.6}$$

The loss is simply the negative log of the probability density function defined by the predicted α and β evaluated at y_{true}

4.2.5 Backpropagation

Backpropagation is an optimization algorithm used for minimizing the error in the neural network. It adjusts the weights and biases in the reverse order - from the output layer to the input layer.

For a given loss function L, the idea is to compute the gradient of L with respect to each weight w_{ij} and bias b_j. The chain rule of calculus plays a pivotal role:

$$\frac{\partial L}{\partial w_{ij}} = \frac{\partial L}{\partial o_j} \frac{\partial o_j}{\partial w_{ij}}$$

Where $\frac{\partial L}{\partial o_j}$ is the error of neuron j.

The weights and biases are then updated using these gradients.

4.3 Optimizers

4.3.1 Gradient Descent & Stochastic Gradient Descent (SGD)

Weights are adjusted to minimize the loss using the gradient:

$$w_{ij}^{\text{new}} = w_{ij}^{\text{old}} - \alpha \frac{\partial L}{\partial w_{ij}}$$

Where α is the learning rate. SGD introduces randomness by updating weights using a subset (batch) of the data[17].

4.3.2 Momentum

Momentum is a technique to prevent the optimizer from getting stuck in local minima and to speed up convergence. It does this by adding a fraction of the update from the previous time step to the current update.

$$v_{t+1} = \beta v_t + (1-\beta)\frac{\partial L}{\partial w_{ij}}$$

$$w_{ij}^{\text{new}} = w_{ij}^{\text{old}} - \alpha v_{t+1}$$

Where v is the velocity, and β is the momentum coefficient.

4.3.3 ADAM Optimizer

ADAM (Adaptive Moment Estimation) combines the ideas of Momentum and RMSProp. It maintains moving averages of the gradient and its square.

Given:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\frac{\partial L}{\partial w_{ij}}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)\left(\frac{\partial L}{\partial w_{ij}}\right)^2$$

The parameters are updated as:

$$w_{ij}^{\text{new}} = w_{ij}^{\text{old}} - \alpha \frac{m_t}{\sqrt{v_t} + \epsilon}$$

Where ϵ is a small number to prevent division by zero[18].

4.4 Graph Neural Networks

Neural networks, especially feedforward neural networks, excel in many applications that deal with fixed-size inputs. However, they encounter challenges when it comes to irregular-sized inputs. It's not feasible to alter the size of neural networks dynamically for each input.

Graph data is one example of irregularly shaped data. It encompasses a wide range of information, from representing social networks such as friend connections on Facebook to detailing molecular structures, and even capturing events in neutrino telescopes. Graph data comprises two primary elements: nodes and edges. Nodes depict entities like individuals, atoms, or detector hits. These nodes can hold specific details about the entity, such as age, gender, the type of atom, the timing of a detector hit, or its charge. On the other hand, edges outline the relationships between nodes. For instance, they can show whether two individuals are friends on a platform or the bond strength between two atoms. It's essential to understand that these edges can be directional. For example, while you might recognize Paul McCartney, he might not know you in return.

Graph Neural Networks (GNN) are designed specifically to handle such graph data.

To truly grasp the inspiration behind Graph Neural Networks (GNN) and Graph Convolution Neural Networks (GCNN), it's beneficial to revisit one of the most influential model architectures in machine learning: the Convolutional Neural Network (CNN)[19]. While I'll delve deeper into the most cited works later, it's worth noting that the CNN was primarily developed for image classification, specifically for a competition called ImageNet. In the context of images, when using traditional feedforward neural networks, the standard approach involves flattening the image into a large vector. Each index in this vector corresponds to a pixel value.

However, this method of flattening an image overlooks the inherent spatial structure and local patterns within the image. The pixel values in an image are not independent of each other; neighbouring pixel values often have meaningful relationships and define features like edges, textures, and patterns. Recognizing this inherent spatial structure was pivotal to the development of the CNN.

Core Concept of CNN

Convolutional Neural Networks (CNNs) are designed to automatically and adaptively learn spatial hierarchies of features from input images. They exploit the spatially-local correlation by enforcing a local connectivity pattern between neurons of adjacent layers. The layers of a CNN have neurons arranged in three dimensions: width, height, and depth. Here are the main components:

- **Convolutional Layer**: This layer applies a series of learnable filters to the input. Each filter is small spatially (along width and height), but extends through the full depth of the input volume. During the forward pass, each filter slides (or "convolves") across the width and height of the input volume to produce a 2-D activation map, capturing the spatial patterns in the data.
- **Pooling Layer**: These layers progressively reduce the spatial size of the representation, reducing the amount of parameters and computation in the network. Pooling layers operate independently on every depth slice of the input and resize it spatially.
- **Fully Connected Layer**: After several convolutional and pooling layers, CNNs often have one or more fully connected layers, where neurons connect to all activations in the previous layer, allowing the network to make predictions based on the entire image.

The main idea behind CNNs is that local spatial patterns in an image are crucial for tasks like object recognition. By using convolutional layers, the network can focus on local regions of the image in its early layers and then aggregate this information in deeper layers to identify larger patterns.

The tremendous success of CNNs in capturing spatial hierarchies in images served as an inspiration for graph neural networks. This success came from the realization that an image can be represented as a graph where pixel values serve as node features. In this representation, edges connect neighbouring pixels, highlighting the spatial relationships and dependencies between them. Using this perspective, Graph Convolutional Neural Networks (GCNNs) were born, aiming to extend this concept of convolutions to any graph, not just the implicit grid-like structure of an image.

One of the foundational concepts behind GCNNs and more broadly GNNs is "message passing". Nodes in the graph aggregate information from their neighbours, passing messages along the edges to iteratively update their representations. This process enables each node to gather information from its larger neighbourhood, much like how CNNs aggregate local spatial information from images.

Moreover, edge convolutions, a concept integral to GCNNs, allow the network to not just focus on node features but also consider the relationships (edges) between nodes. This leads to richer, more contextual representations of nodes in the graph, accounting for both their intrinsic properties and their position and role within the larger structure[20].

The innovation of edge convolutions and message passing in GNNs set the stage for more sophisticated models like the DynEdge, which we'll delve into in the next section.

4.4.1 DynEdge Model

The DynEdge Model is a product of the GraphNet team, which has developed an open-source Python framework, GraphNet. This framework is specifically designed to provide graph neural networks for reconstructing data from neutrino telescopes, such as IceCube[21].

The design of the DynEdge model draws inspiration from the model proposed in the paper "Dynamic Graph CNN for Learning on Point Clouds"[22]. This model is a graph convolutional neural network that employs edge convolution on point cloud data with the primary objective of segmentation. In their application, they worked with 3D models where the point cloud represented points on the surface of these models. The model's task was to predict which part of the 3D model a particular point belonged to, for instance, determining if a point on an aeroplane model was part of the wing or the fuselage.

The operational mechanism of this model involves connecting each point to its k nearest neighbours using edges, followed by performing edge convolution. Post convolution, each point is represented in a latent space, and new edges are added, connecting the k nearest neighbours in this space, thereby replacing the old edges. This process is iterated multiple times, depending on the desired model size. At the end of the model, the latent space representations of a point are concatenated and processed through a Feed-Forward neural network to produce the final prediction for each point.

While the DynEdge model operates on principles similar to those proposed by the "Dynamic Graph CNN for Learning on Point Clouds" paper, it introduces several modifications. The most significant distinction lies in handling the output post the convolution layers. While the original model from the paper predicts an output for each point, the DynEdge model often requires a single or fixed number of outputs for an entire event, as opposed to predictions for each DOM hit (an exception being noise prediction).

To achieve this, DynEdge incorporates a 'node aggregation' layer. In this layer, all concatenated latent space features undergo aggregation operations, such as calculating the mean, maximum, minimum, and sum across all latent features. This results in an output size that is independent of the number of DOM hits. This output is subsequently passed through an FNN to derive the final prediction. A detailed diagram of the DynEdge model is illustrated in Figure 4.5.

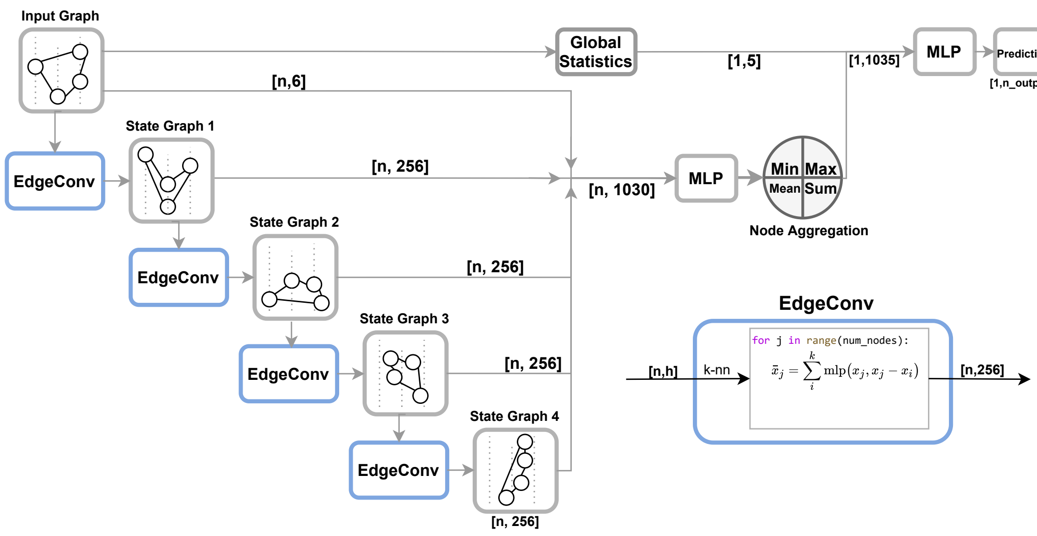


FIGURE 4.1: Diagram of the DynEdge model from[21].

4.5 The Transformer Architecture

The Transformer architecture has quickly risen to prominence in various machine learning domains. Its notoriety skyrocketed following the remarkable success of Generative Pre-trained Transformer models, notably ChatGPT. Primarily, Transformers have found their niche in the realm of natural language processing (NLP). Several reasons underpin the popularity of the Transformer architecture:

Scalability: A hallmark feature of the Transformer model is its ability to scale efficiently with increased model size and training data. While many machine learning architectures exhibit diminishing returns upon expanding the model and data size, the limits of Transformers in this regard remain uncharted. While there is speculation about OpenAI's GPT-4 reaching a plateau with its purported 1.7 trillion learnable parameters, definitive research on the topic is scarce. Despite OpenAI's somewhat secretive stance, the field of large language models is still nascent, and to date, there is no concrete evidence showing that performance plateaus with increased model and data size.

Versatility: While the primary application of Transformers remains in NLP, the architecture's flexibility has spurred interest across various research domains, including image recognition.

However, the Transformer architecture is not without its challenges. A significant drawback pertains to time complexity. Typically, architectures addressing sequence data, such as recurrent neural networks (RNN), have a time complexity of $O(n)$, indicating that the number of calculations required scales linearly with the sequence length. In contrast, Transformers have a time complexity of $O(n^2)$ due to their attention mechanisms. This quadratic scaling arises from the attention matrices' calculations, which involve two vectors of equivalent sequence length. While this might seem daunting, it's essential to understand that this high time complexity primarily becomes a concern for extremely long sequences. Only when sequences reach significant lengths does this time complexity overshadow other model-size-dependent computations, like backpropagation, in terms of the total number of required calculations.

In the subsequent sections, I will delve deeper into the Transformer architecture tailored for inelasticity reconstructions. The design of the Transformer model adopted in this research is largely influenced by the second-place finisher in the Kaggle competition titled "Neutrinos in Deep Ice." The primary distinction between our model and theirs lies in the embedding, which accounts for feature differences. This is attributable to the fact that my data pertains to the upgraded IceCube dataset, whereas the Kaggle competition focused on the current IceCube data.

To facilitate a comprehensive understanding, I will explain the foundational elements of my Transformer model. This includes an overview of the original Transformer model, the modified attention mechanism—Self-Attention with Relative Position Representations[23], and the classification token (CLS)[24].

4.5.1 Attention is all you need

First introduced in the paper "Attention is All You Need"[25], the Transformer model was designed for machine translation tasks. It takes a sequence of words in one language as its input and outputs a translated sequence of words in another language.

In natural language processing, the term 'token' is used instead of 'word'. This is because tokens are not necessarily whole words; many models split words into multiple tokens, enabling them to generate text with words they haven't encountered before. Similarly, in our context, it's more useful to think in terms of tokens rather than words, as the IceCube Observatory detects neutrinos rather than words.

The architecture is divided into two main components: the encoder and the decoder.

The decoder is responsible for the generative aspect of the model, allowing the output sequence length to vary from the input sequence length. However, when the entire input sequence is the target, such as in sentence classification tasks (e.g., determining whether a sentence violates social media platform guidelines), or token/word classification tasks, only the encoder is needed. In other words, if your model's output is a fixed size or a multiple of the input sequence length, the encoder part is sufficient. This thesis will therefore focus solely on the encoder part of the model.

In the original Transformer model, the encoder takes a sequence of tokens $x = (x_1, ..., x_n)$ as input and as the first part of the encoder passes the tokens through an embedding layer, where each token is projected from its original representation into a high-dimensional space $z = (z_1, ..., z_n)$ here z_1 to z_n represents the tokens in the sequence. The embedded tokens are a vector of size dz. The objective of this embedding in the context of NLP is to map semantically similar or related words close together in this high-dimensional space. This embedding is often pre-computed and contains no learnable parameters.

The encoder's next layer is the positional embedding. This layer is necessary because the rest of the Transformer architecture is agnostic to sequence order. In the context of NLP, two sentences with the same words but different orderings will yield the same result. For example, "The bear will cross the road." and "The road will bear the cross." would be identical to the Transformer. To break this invariance, a positional component is introduced and added to each token. Often called Fourier positional encoding, this component consists of adding sine and cosine waves of differing wavelengths depending on the token's position within the sequence.[25]

$$\begin{aligned} PE_{(pos,2i)} &= sin(\frac{pos}{10000^{\frac{2i}{dz}}}) \\ PE_{(pos,2i+1)} &= cos(\frac{pos}{10000^{\frac{2i}{dz}}}) \end{aligned} \tag{4.7}$$

Here is pos the position of the token in the sequence, i is an index that goes from 0 to $dz/2$ so that PE is a vector of size dz. The positional embeddings can be seen in Figure 4.2 where each row is the embedding for a given sequence position on the y-axis. it is worth noting that in the case of NLP position is a positive integer but the embedding also works for continuous variables which will be relevant later when making transformer models for IceCube data.

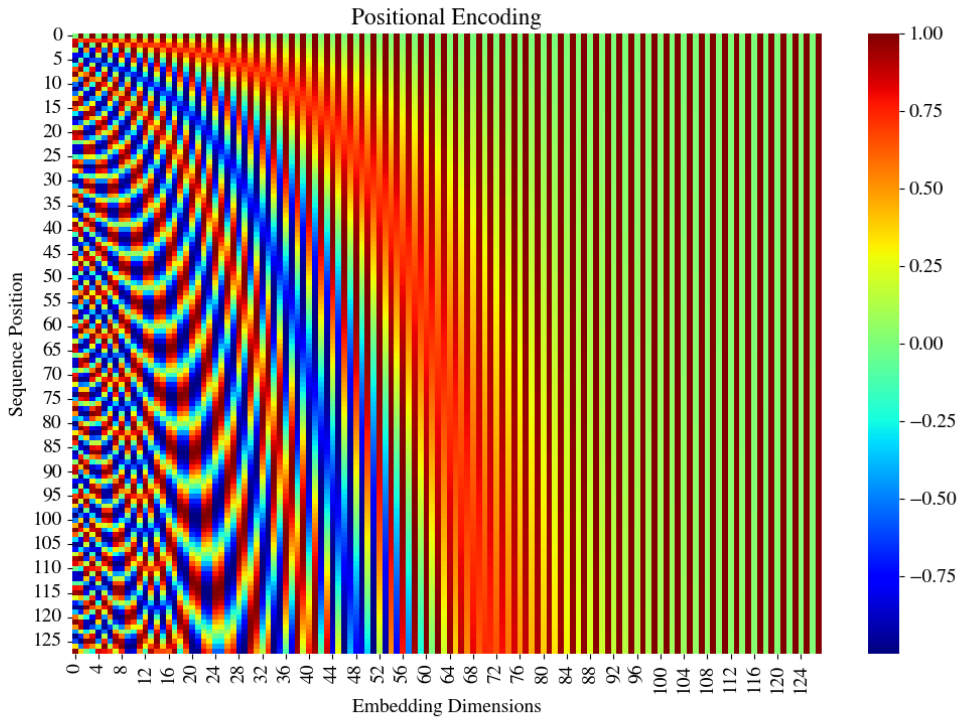


FIGURE 4.2: Positional embeddings plotted as a heatmap. plotted the first 128 position with dz = 128

Following the positional encoding, we arrive at the first encoder layer. The encoder layer is composed of several sub-layers: "multi-head attention", "add & norm", and "feed-forward".

The multi-head attention layer first computes the "key", "query", and "value" (denoted as K, Q, and V), by multiplying the embedded tokens by three learned weight matrices W^Q, W^K, W^V all of size dx by dz The attention weights are then calculated using K and Q, according to the equation:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V = \sum_{j=0}^{n} a_{ij}(x_j W^V) \tag{4.8}$$

where a_{ij} is the attention weights

$$a_{ij} = \frac{\exp e_{ij}}{\sum_{k=1}^{n} \exp e_{ik}} \tag{4.9}$$

and e_{ij}

$$e_{ij} = \frac{(x_i W^Q)(x_j W^K)^T}{\sqrt{d_z}} \tag{4.10}$$

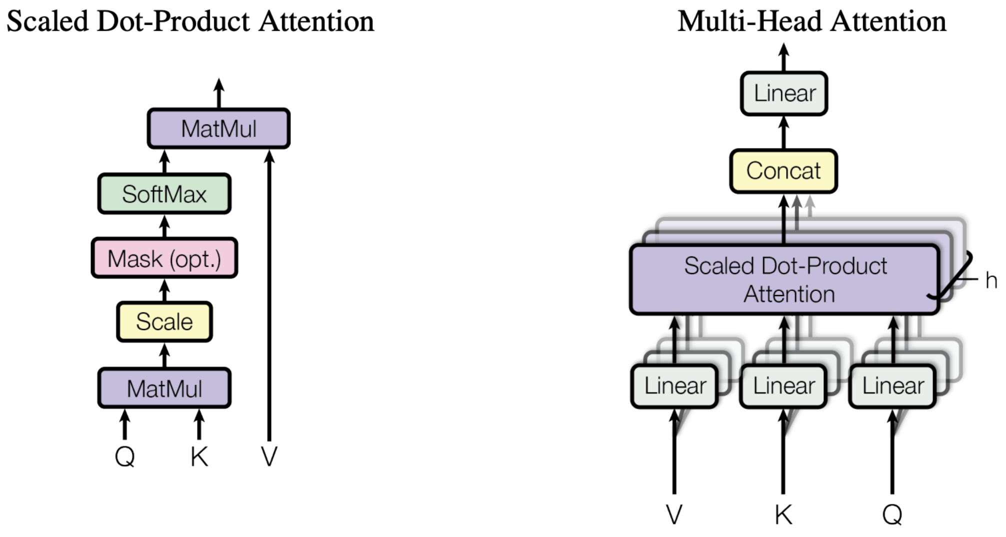


FIGURE 4.3: left: Scaled Dot-Product Attention Right: Multi-Head Attention.

This type of attention, known as dot-product (or scaled) attention, is commonly used in Transformer models due to its computational efficiency. This process is performed multiple times in parallel for each "head" using different weight matrices. Each self-attention head can learn different information, effectively highlighting tokens that share specific information.

The output from all the different heads is then concatenated and passed through a linear layer to maintain the same output shape as the input shape. The inputs from before the multi-head attention are also concatenated with the output from the multi-head attention, thus preserving information extracted in the first layers for later layers. This practice of concatenating the input with the output of a layer is known as a residual connection[26]. The combination of the residual connection and layer normalization is referred to as an "add & norm" layer.

After the "add & norm" layer, the output is passed through a layer normalization which scales all values to between 0 and 1 [27], followed by a short fully connected feed-forward layer and another "add & norm". This completes one encoder layer. To construct the desired size model, this process is repeated for the desired number of encoder layers.

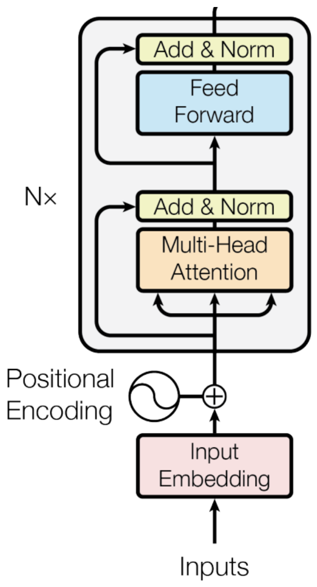


FIGURE 4.4: Overview of the regular transformer encoder layer

4.5.2 Self-Attention with Relative Position Representations

As mentioned the original Transformer was primarily designed for natural language processing (NLP). However, with its rising popularity, its applications have expanded to other fields, such as image processing. As the Transformer evolved, researchers sought more flexible ways to represent data structure beyond just positional encoding. One such method we're incorporating into our model is "Self-Attention with Relative Position Representations."[23]. Essentially, the method alters the way attention is computed to incorporate a learnable pairwise connection between all input tokens. This introduces edge representations between two tokens for both the value and key, denoted as a_{ij}^V and a_{ij}^K respectively, where i and j indicate the edge between tokens x_i and x_j.

The attention calculation is then modified as:

$$Attention(Q, K, V) = \sum_{j=0}^{n} a_{ij}(x_j W^V + a_{ij}^V) \tag{4.11}$$

$$a_{ij} = \frac{\exp e_{ij}}{\sum_{k=1}^{n} \exp e_{ik}} \tag{4.12}$$

$$e_{ij} = \frac{(x_i W^Q)(x_j W^K + a_{ij}^K)^T}{\sqrt{d_z}} \tag{4.13}$$

This modification adds a learnable position-dependent bias to the calculation. The original paper proposed a method to cap the length, but I chose not to implement it, as our main focus is on low-energy events where the number of DOM hits isn't problematic. However, other analyses might consider this aspect.

4.5.3 Classification Token

The classification token(CLS) is a technique to condense the encoder layer's output from a size dependent on input tokens to a fixed size. A token of only zeros is appended to the sequence, and in the end, only this classification token is processed through a feed-forward neural network to obtain the output[24]. It is worth noting that the CLS token might clash with Self-Attention with Relative Position Representations as the CLS token doesn't have any meaningful position. Therefore in the model, I will switch from using the encoder layer with the relative position representation to the regular encoder layer after appending the CLS token

4.6 My Transformer Model

Our model derives inspiration from the second-place solution in the Kaggle "competition IceCube - Neutrinos in Deep Ice"[28], titled "2nd place solution: Neutrino direction prediction with transformers."[29] My model differs mainly in the embedding layer and the target. While their use case focused on reconstructing the Zenith angle, my target is inelasticity. My modified embedding is necessitated due to the difference in the provided features, and the fact that we're using upgrade data, which has a variety of DOM types.

The embedding takes inspiration from the original transformer positional embedding. For continuous variables like position, time, and charge, we utilize a Fourier embedding. Discrete variables, like the 3 DOM types and 2 DOM efficiencies, are processed through a standard embedding layer, which maps them to five different vectors (standard IceCube DOM, high quantum efficiency DOM, Upgrade pDOM, dEgg, and mDOM). These embeddings are concatenated and passed through a linear layer to reduce the embedding size.

The structure of the Transformer model is as follows:

- Our modified embedding layer.
- Four encoder layers with Relative Position Representations.
- Appending of the CLS token to the input.
- Twelve regular encoder layers.
- Processing the CLS token through an MLP, consisting of a linear layer, an activation layer, and another linear layer.
- The output then goes through an activation function to achieve the desired range.

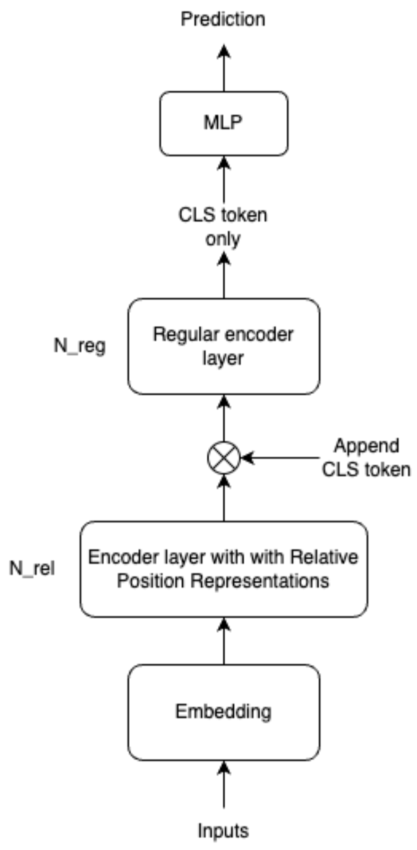


FIGURE 4.5: Overview of my transformer model

For the beta loss, the MLP outputs two values, which are processed via a softplus activation layer to determine the values for alpha and beta in the beta distribution. We opted for softplus over the standard ReLU as ReLU yields 0 for all values below 0, and the beta distribution isn't defined when alpha or beta is 0.

4.6.1 Transformer Training and Data Loading Considerations

When training a transformer model, it's vital to consider the nuances of data loading. For computational efficiency, events are batched together during training. However, it's essential that all events within a batch have the same length. To achieve this uniformity, I pad the events with tokens, ensuring that each event matches the length of the longest event in the batch.

Fortunately, the transformer's attention mechanism is adept at handling such padding. By setting the attention scores of all padding tokens to zero—achieved by masking these tokens—the model effectively disregards them during output calculations. This ensures that padding does not adversely influence the model's predictions.

However, there's an inherent challenge with padding: it significantly amplifies the number of required calculations. As mentioned earlier the time complexity of a transformer is $O(n^2)$ so adding unnecessary padding becomes costly especially due to the distribution of the number of DOM hits in events for IceCube data with mostly shorter events but a few really long ones. Even though the attention scores for padding tokens are set to zero post-multiplication, there's no efficient way to bypass these calculations initially.

To optimize training time, a strategy to minimize the number of padding tokens is needed. I employed a technique known as "sequence bucketing." Instead of randomly batching events, I sorted them into "buckets" based on their lengths. When forming a training batch, events are selected from the same bucket, ensuring similar lengths and thus minimal padding.

This bucketing approach also offers the flexibility to vary batch sizes based on event lengths, optimizing GPU memory usage during training. However, this technique of variable batch sizes introduces a subtle challenge. Loss is often calculated as the average loss across the batch. If the label under consideration correlates with event lengths—a common occurrence with IceCube data, such as during energy reconstruction—this can skew the gradient calculations. In such scenarios, shorter events effectively have reduced weight in gradient calculations. While this bias can be rectified by adjusting the loss based on batch size, I chose not to implement this corrective measure and simply had the same batch size for all buckets.

Chapter 5

Inelasticity Reconstruction

In the upcoming section, I will detail my journey of understanding inelasticity. This exploration began with several preliminary approaches that, while insightful, did not yield the desired results. It was the discovery of the Beta Loss method that proved transformative. Armed with the Beta Loss reconstruction, I proceeded to assess the precision with which I could differentiate between neutrinos and anti-neutrinos.

5.1 Initial DynEdge reconstruction attempts

In my initial attempts to understand inelasticity, I centred my research on the premise that if we can accurately measure both energy and its corresponding track, then we should also be able to calculate inelasticity using these two parameters following equation 2.7.

All the energy reconstructions were done on log10 of the energy or energy track. This transformation is favoured primarily because it aligns the model's error tendencies with our inherent understanding of energy measurements.

In this logarithmic domain, an absolute error translates to a relative error in the linear space. For clarity, consider two energy scenarios:

- Distinguishing between particles with energies of 5 GeV and 30 GeV.
- Distinguishing between particles with energies of 300 GeV and 325 GeV.

While the absolute energy differences in both scenarios are 25 GeV, the relative difference in the first is significantly more pronounced than in the second. This distinction is more apparent in the logarithmic domain. For instance, the difference between 5 and 30 GeV particles is 0.78, whereas the difference between 325 and 300 GeV particles in log10 space is merely 0.03.

Training in the log space ensures that the model inherently differentiates these scenarios, prioritizing relative energy differences over absolute ones. This is intuitive since distinguishing between particles with energies close in relative terms (like 5 GeV vs. 30 GeV) is inherently more intricate than those close in absolute but distant in relative terms (like 300 GeV vs. 325 GeV).

My initial explorations predominantly hinged on the DynEdge model, given that the Transformer model was conceived much later in my research journey.

In numerous preliminary trials, I instituted specific conditions, imposing cuts on energy and the interaction vertex. My rationale was to ascertain if these restrictions could potentially enhance the reconstruction process. Specifically, the conditions were set with energy ranging between 15 GeV to 300 GeV, and the interaction vertex was confined within a cylinder of radius 120 m of the centre of the detector.

5.1.1 Reconstruction with LogCoshloss

After an initial review of the existing work, I chose to base my initial reconstruction attempt on Marc's thesis, whose results are shown in Figure 5.1. My goal was to reproduce his reconstructions for energy, energy track, and inelasticity using the LogCoshloss function. However, it's crucial to note that my data differed from Marc's; it underwent GraphNet cleaning to remove noise.

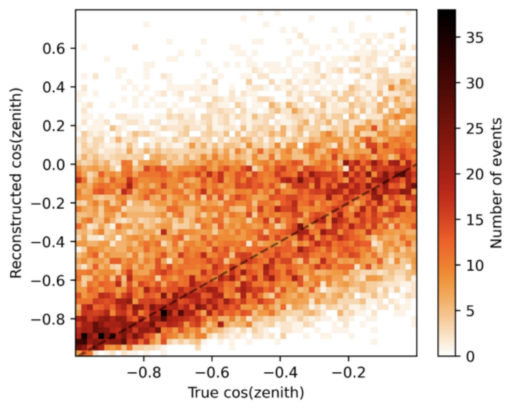


Figure 20: $\cos\theta_z$ reconstruction

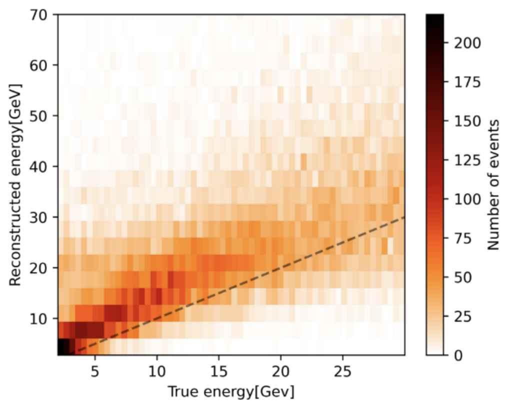
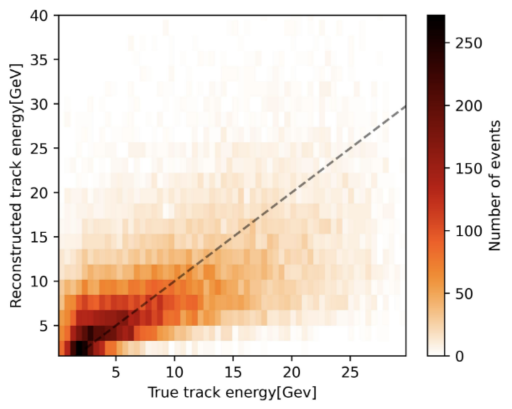


Figure 21: E_ν reconstruction.



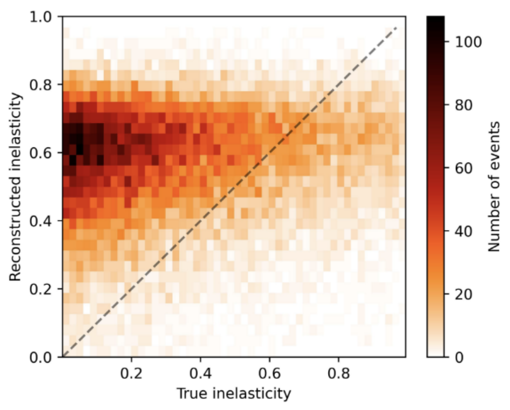


FIGURE 5.1: Reconstruction of Zenith, Energy, Energy track and Inelasticity from Marc's thesis[9]

We can see in figure 5.2 that I am able to reconstruct both the energy and the energy track with reasonable precision and bearing resemblance to Marc's reconstruction in figure 5.1. However, there's a subtle difference in our inelasticity reconstructions. While both display a prominent peak, indicating the model's tendency to guess a singular value for most events, the peaks' centres differ. Marc's peak centres around 0.6, whereas mine is around 0.36, aligning with the inelasticity distribution's mean. Interestingly, Marc's inelasticity appears as if it's been horizontally flipped. Nevertheless, the overarching conclusion remains consistent: the reconstruction fails for most events, predicting the mean inelasticity. A closer inspection might reveal a faint correlation, suggesting partial reconstruction for a few events.

Upon closer inspection of the inelasticity reconstruction, one can discern a subtle correlation, despite the dominant peak. there at least seem to be more events reconstructed near (1,1) than (0,1) and more at (0,0) than (1,0) which would indicate that there were a few events it was partially able to reconstruct.

I also reconstructed the energy and energy track of events, in addition to inelasticity, as they can be utilized in equation 2.7 to have an alternative inelasticity reconstruct method.

This Reconstruction can be seen in the left plot in figure 5.3 However, this alternative reconstruction proved less effective than the direct LogCosh loss method.

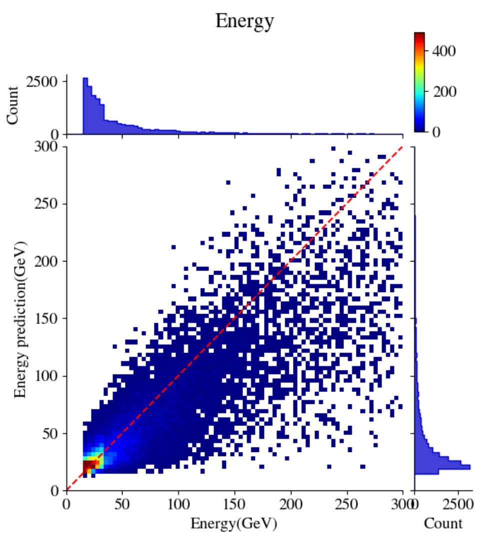
5.1.2 Reconstruction with energy, energy track and inelasticity in one loss

Building on my earlier attempts, I hypothesized that a combined loss function targeting energy, energy track, and inelasticity simultaneously could offer improved reconstructions. The idea was to leverage the relative ease of reconstructing energy and energy track to guide the more challenging inelasticity reconstruction.

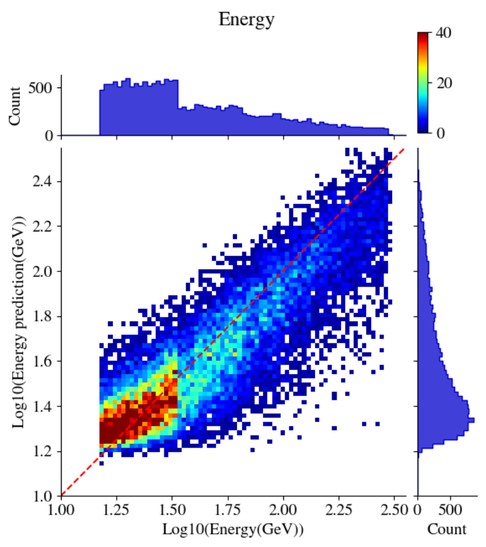
The error passed to the loss function was then the Euclidean distance between the truth vectors and the prediction vector. I scaled the log10 energies by 4 to ensure inelasticity was the predominant factor in the loss function.

$$Loss = LogCosh\left(\sqrt{\left(\frac{E_{tot,pred} - E_{tot,true}}{4}\right)^2 + \left(\frac{E_{track,pred} - E_{track,true}}{4}\right)^2 + (y_{pred} - y_{true})^2}\right)$$

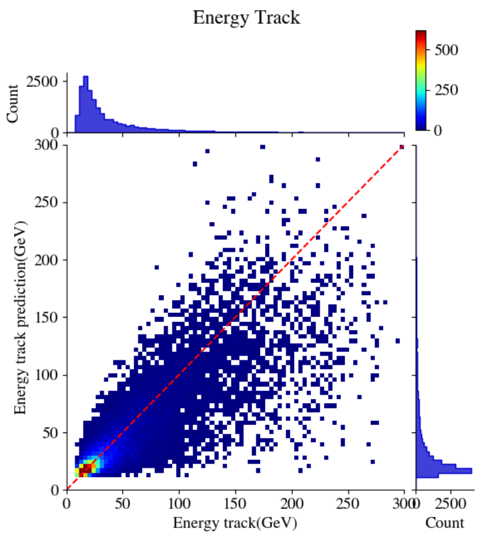
This approach ultimately wasn't fruitful. The inelasticity prediction is still predominantly centred around the mean inelasticity, with some variance as can be seen in figure 5.5. Moreover, the predictions for energy and energy track were consistently inaccurate as seen in figure5.4. My interpretation suggests that the model's higher inelasticity weighting in the loss function led it to deliberately predict slightly incorrect energy and energy track values, ensuring the inelasticity always matched the mean inelasticity.



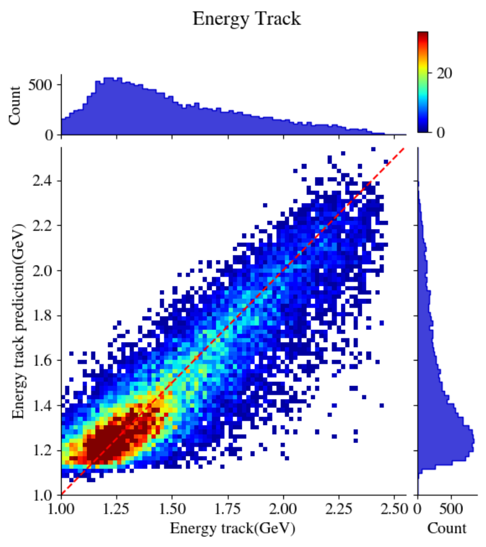
(A) Energy reconstruction using LogCosh DynEdge with loss



(B) log_{10}(Energy) reconstruction using DynEdge with LogCosh loss



(C) Energy reconstruction using LogCosh loss



(D) log_{10}(Energy track) reconstruction using DynEdge with LogCosh loss

FIGURE 5.2: Energy and energy track reconstruction using DynEdge with LogCosh loss

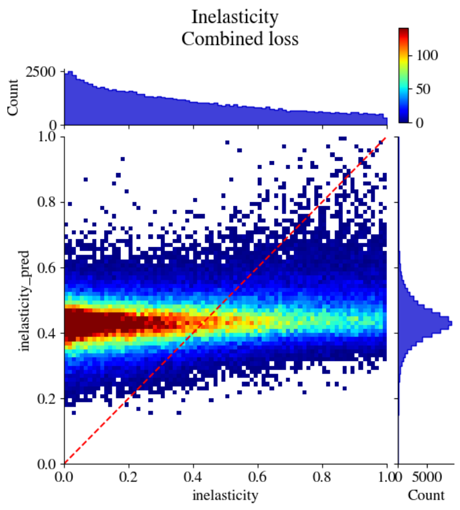
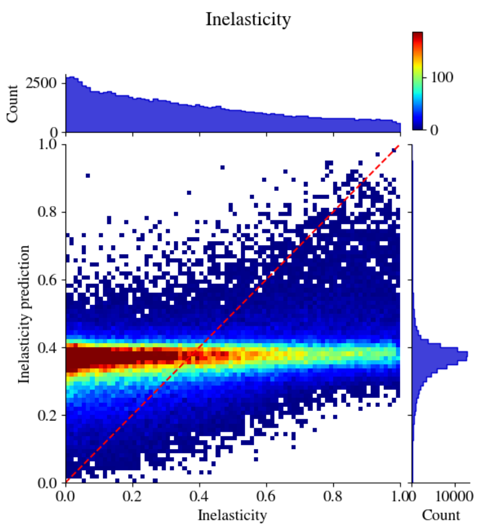
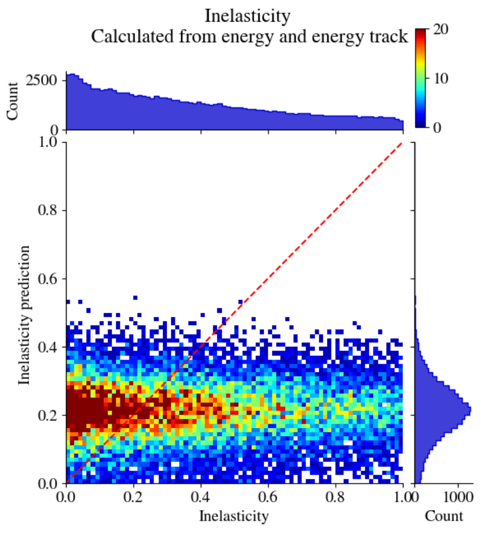


FIGURE 5.5: Inelasticity reconstructed using combined loss



(A) Inelasticity reconstruction using DynEdge with LogCosh loss



(B) Inelasticity reconstructed using the reconstructed Energies and track energies reconstruction using DynEdge with LogCosh loss

FIGURE 5.3: Inelasticity reconstructed using LogCosh loss

5.2 Beta Loss reconstruction

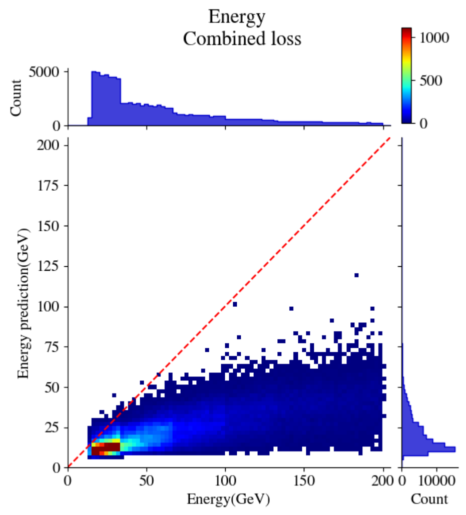
The Beta loss is implemented for both the DynEdge model and the Transformer model. The DynEdge model was trained for 120 epochs which took 6 hours and reached a minimum validation loss of -0.2665. The Transformer model trained for 24 epochs and took just over 3 hours and reached a minimum validation loss of -0.2532. It should be noted that I have had a lot more time to find the best DynEdge hyperparameters than I had for the Transformer model.

The event selection for training both models differs slightly from previous experiments. No constraints were placed on energy or interaction/vertex position. The sole criterion was the number of DOM hits in the event, specifically between 5 and 300, post noise-cleaning. This decision was twofold:

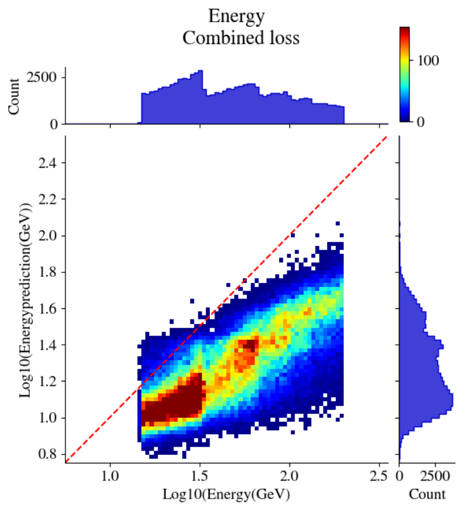
- Events with fewer than 5 hits likely lack sufficient information for inelasticity reconstruction.
- The upper limit is imposed due to the Transformer model's time complexity of $O(n^2)$ as described in section 4.6.1 Given the distribution of event lengths and our focus on low-energy events, a maximum length of 300 DOM hits was deemed appropriate.

5.2.1 DynEdge and Transformer Beta Loss reconstruction

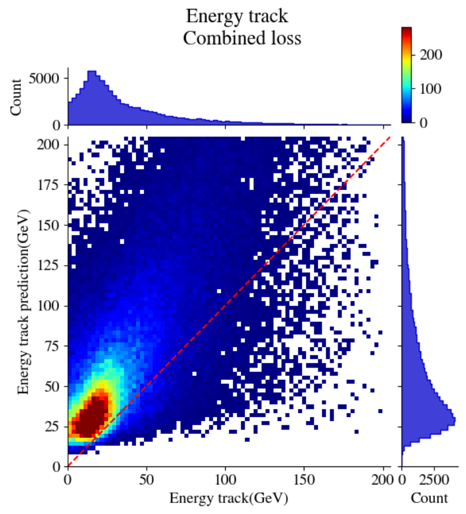
The idea behind using my Beta loss is that I wanted to try to reconstruct inelasticity with estimated uncertainties. the problem is that most methods that implement uncertainties do so with Gaussian uncertainties. The problem is that inelasticity is bounded in the range 0 to 1 which the Gaussian isn't. The closest thing to a Gaussian bounded in the range 0 to 1 is a beta distribution



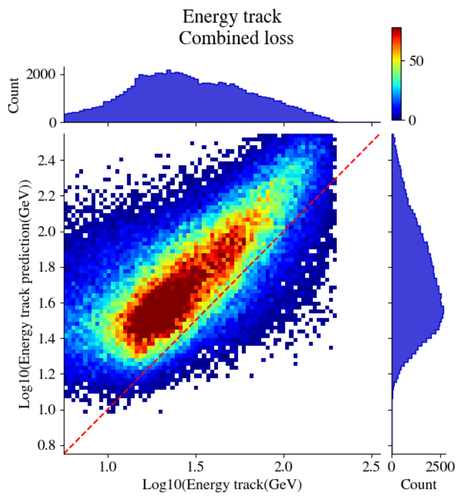
(A) Energy reconstructed using combined loss



(B) log_{10}(Energy) reconstructed using combined loss



(C) Energy track reconstructed using combined loss



(D) log_{10}(Energy track) reconstructed using combined loss

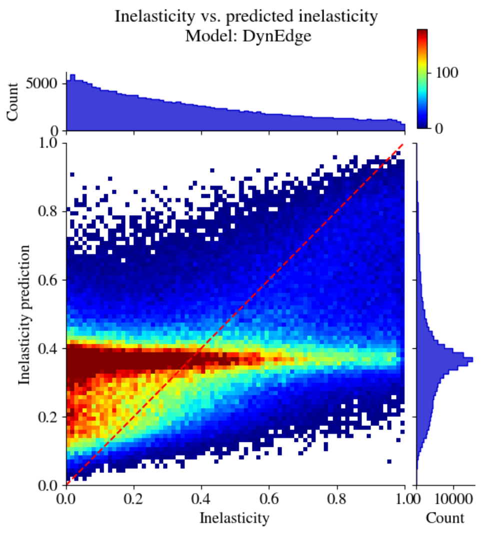
FIGURE 5.4: Energy and energy track reconstructions using combined loss

When visualizing the Beta loss reconstruction, a decision must be made regarding the predicted inelasticity since the output is a distribution. The most intuitive approach, which I adopted, is to select the distribution's mean. An alternative could be the mode, akin to classification strategies. However, a comparison between the mean and mode against true inelasticity revealed the mean as consistently superior.

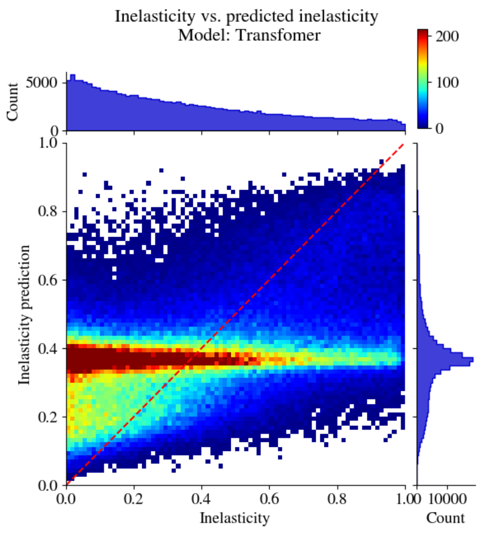
Upon initial observation, the Beta loss reconstruction appears largely unchanged, with a prominent peak at the mean inelasticity. However, a closer look reveals a subset of events predicted to have a lower inelasticity, around 0.2, which indeed aligns with actual low inelasticity events. This is evident in Figure 5.6 for both models, though more pronounced in DynEdge reconstructions.

Given that we're predicting a distribution, rather than just a single value, we can filter events reconstructed with higher uncertainty. One method is to compute the distribution's variance, but this inherently favours events near 0 or 1. A more straightforward metric is the sum of α and β, which reflects the number of trials in a Beta distribution. I opted for a threshold where $\alpha + \beta > 6$. Examples of beta distributions where $\alpha + \beta = 6$ are illustrated in Figure 5.8. For comparison, the median prediction is plotted together with the highest $\alpha + \beta$ to illustrate the range of prediction in Figure 5.9. note the similarities with the median prediction that the overall inelasticity distribution.

Applying this filter significantly refines the reconstructed events, as seen in Figure 5.7. The dominant peak at the mean inelasticity vanishes, suggesting the model's awareness of events with insufficient information. However, a significant portion of events, 84% for DynEdge and 89% for the Transformer, fall below this threshold, indicating challenges in inelasticity reconstruction. Notably, the model hesitates to predict means above 0.9 or below 0.1, Here should it be noted that some of the transform figures have been moved to appendix A as many are almost identical the the DynEdge counterpart but with ever so slightly worse performance.

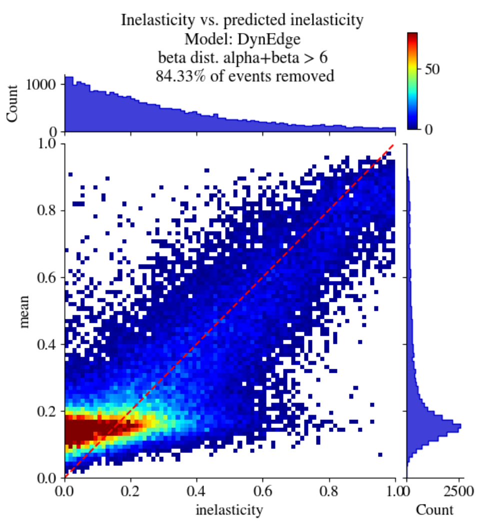


(A) Mean of the DynEdge predicted beta distribution vs true Inelasticity

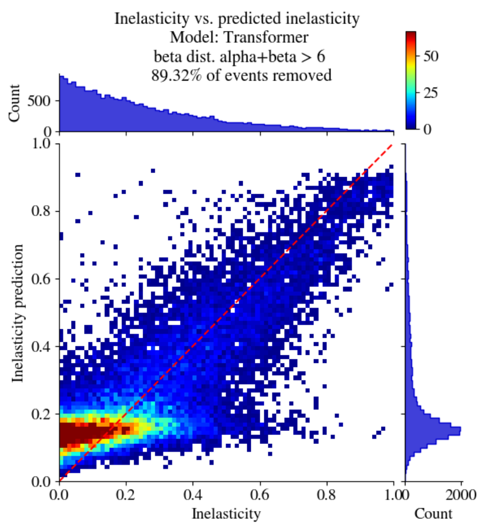


(B) Mean of the Transformer predicted beta distribution vs true Inelasticity

FIGURE 5.6: inelasticity reconstruction using Beta Loss



(A) DynEdge Inelasticity Reconstruction for events where $\alpha + \beta > 6$



(B) Transformer Inelasticity Reconstruction for events where $\alpha + \beta > 6$

FIGURE 5.7: Inelasticity Reconstruction for events where $\alpha + \beta > 6$

Cuts on alpha beta sum

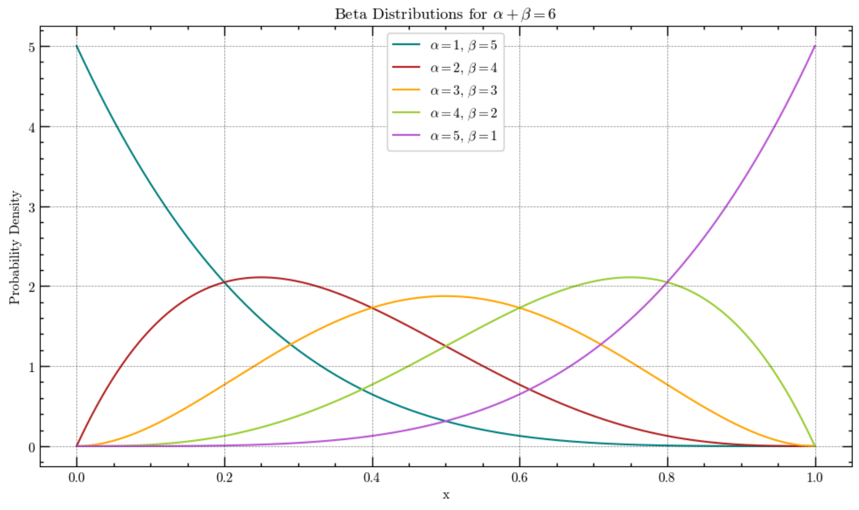


FIGURE 5.8: Different beta distribution where $\alpha + \beta = 6$

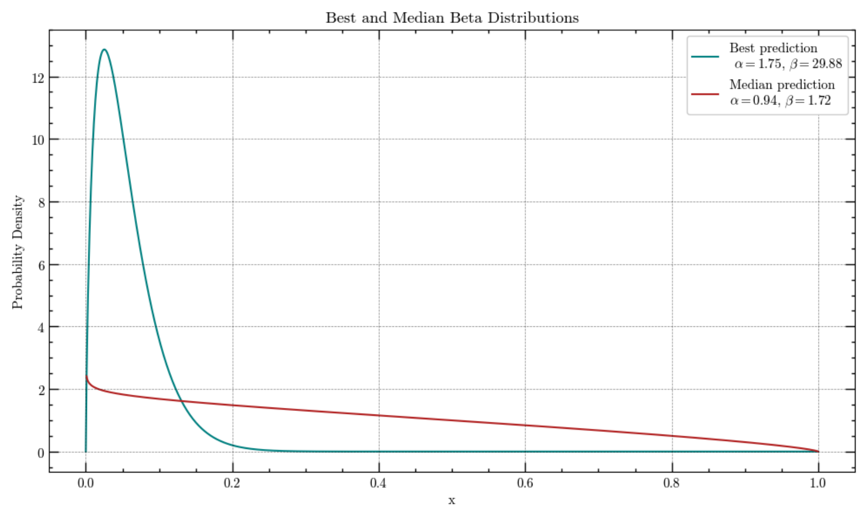
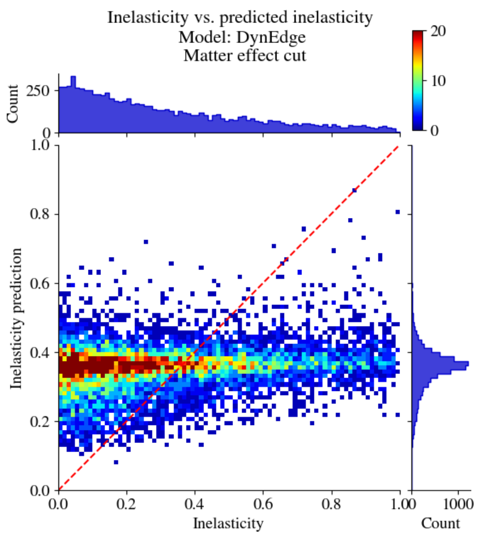


FIGURE 5.9: beta distribution for the prediction with the highest $\alpha + \beta$ and the median prediction

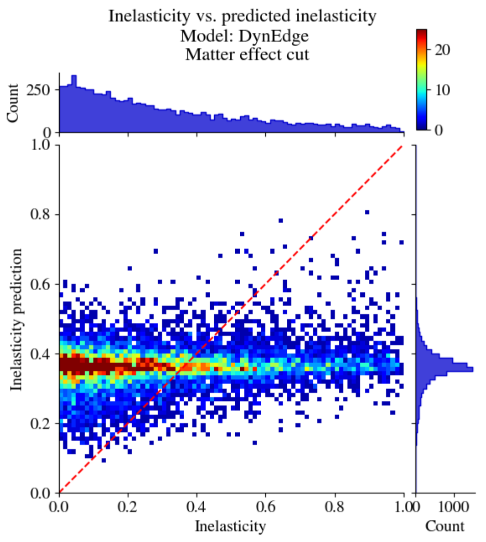
5.2.2 Reconstruction in matter effect range

As previously highlighted, our primary interest in reconstructing inelasticity stems from the need to differentiate between neutrinos and anti-neutrinos within the energy and zenith range where matter effects are evident. These matter effects are illustrated in Figure 2.3, with the energy range being 1-20 GeV and a zenith range of cos(zenith) below -0.75.

The reconstructed events within this range are depicted in Figure 5.10. At first glance, the results appear less than ideal. The cluster of events reconstructed at 0.2 is not as pronounced as when the matter cut isn't applied. However, upon applying the same criterion of $\alpha + \beta > 6$, a subset of events meets this threshold and exhibits reasonable reconstruction. It's worth noting that while 16% (DynEdge) and 11% (Transformer) of events met this criterion without the matter cut, only 1.37% and 0.71% do so within the matter effect range.

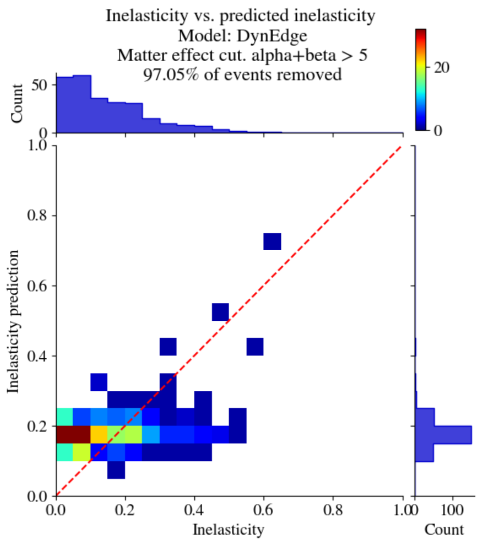


(A) DynEdge inelasticity reconstruction for events where Energy < 20 Gev and cos(z) <-0.75

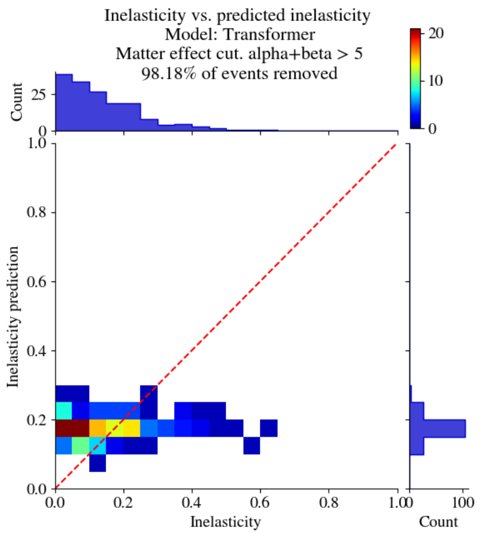


(B) Transformer inelasticity reconstruction for events where Energy < 20 Gev and cos(z) <-0.75

FIGURE 5.10: inelasticity reconstruction for events where Energy < 20 Gev and cos(z) <-0.75



(A) DynEdge inelasticity reconstruction for events where Energy < 20 Gev and cos(z) <-0.75 and $\alpha + \beta > 6$



(B) Transformer inelasticity reconstruction for events where Energy < 20 Gev and cos(z) <-0.75 and $\alpha + \beta > 6$

FIGURE 5.11: Inelasticity reconstruction for events where Energy < 20 Gev and cos(z) <-0.75 abd $\alpha + \beta > 6$

5.3 Neutrino anti-neutrino classification using inelasticity

Distinguishing between neutrinos and anti-neutrinos is pivotal for understanding their properties and interactions. In this section, I explore how the reconstructed inelasticity can be employed as a discriminating feature for such classification.

To differentiate neutrinos from anti-neutrinos based on inelasticity, a straightforward approach is to set a threshold. Events with inelasticity above this threshold are classified as neutrinos, and those below as anti-neutrinos. The actual inelasticity distributions for both are shown in Figure 5.12.

Figure 5.13 presents the predicted inelasticity from both DynEdge and the Transformer. The distinction isn't as clear-cut as in the actual inelasticity distribution. When we apply the energy and cos(zenith) cuts to isolate events influenced by matter effects, the distinction nearly vanishes, as seen in Figure 5.16.

To visualize the threshold selection, I've plotted a Receiver Operating Characteristics (ROC) curve in Figure 5.14. This curve juxtaposes the True Positive Rate (TPR) against the False Positive Rate (FPR). Notably, even with the challenges in distinguishing reconstructed inelasticity distributions within the matter effect range, there's a slight capability to differentiate between neutrinos and anti-neutrinos using inelasticity.

5.3.1 Metric for Comparing Cuts and Selections: Best Balanced Accuracy

To effectively compare the various cuts and selections, it's pivotal to employ a metric known as the "best balanced accuracy."

Balanced accuracy is formally defined as:

$$Acc_{balanced} = \frac{TPR + TNR}{2} = \frac{TPR + (1 - FPR)}{2} \tag{5.1}$$

Here, TNR represents the true negative rate.

"best balanced accuracy" is calculated by finding the maximum balanced accuracy value across all combinations of TPR and FPR showcased in the ROC curves.

Opting for balanced accuracy over conventional accuracy offers the advantage of robustness against dataset imbalances. Such imbalances might arise when one category significantly outweighs another in representation.

While our dataset might seem fairly balanced when referencing Figure 5.12, the introduction of cuts based on $\alpha + \beta$ could unintentionally skew the ratio of neutrinos to anti-neutrinos, or vice versa. Had I solely relied on the standard accuracy metric for neutrino classification, there might have been a misconception that the cut was enhancing classification by eliminating uncertain predictions. In reality, the cut could have amplified the classification accuracy for neutrinos but diminished it for anti-neutrinos.

For illustrative purposes, Figure 5.14 includes the ROC curve derived from an idealized scenario where inelasticity is reconstructed flawlessly, devoid of any uncertainties. Naturally, this is a theoretical construct, but it's worth noting that even under such optimal conditions, the best balanced accuracy peaks at just 62.23%.

However, Figure 5.14 doesn't capture the entire picture. It doesn't consider the potential benefits of applying stricter $\alpha + \beta$ cuts to enhance the distinction between neutrinos and anti-neutrinos. In Figure 5.15, the ROC curve for neutrino vs. anti-neutrino separation using DynEdge's inelasticity reconstruction is plotted with increasingly stringent $\alpha + \beta$ cuts. While stricter cuts do enhance separation, the benefits diminish. For instance, raising the cut from 5 to 10 only marginally improved separation with the best balanced accuracy going from 57.5% to 58.5% but resulted in a loss of 18% points of the data. This presents a trade-off between improved separation and data loss, necessitating a balanced approach.

Cut Value	**Balanced Accuracy**	**Events Excluded**
no cut	54.92%	0%
2.7	56.43%	47.51%
5	57.5%	78.21%
10	58.5%	96.25%

TABLE 5.1: Effects of $\alpha + \beta$ Cuts on Accuracy and Event Exclusion

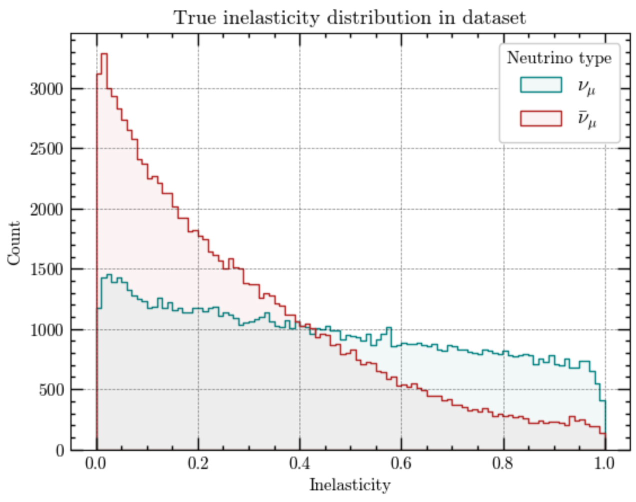
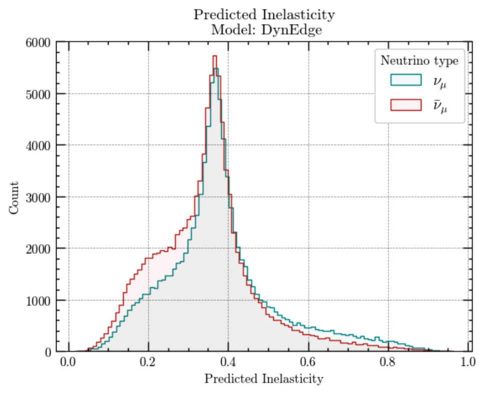
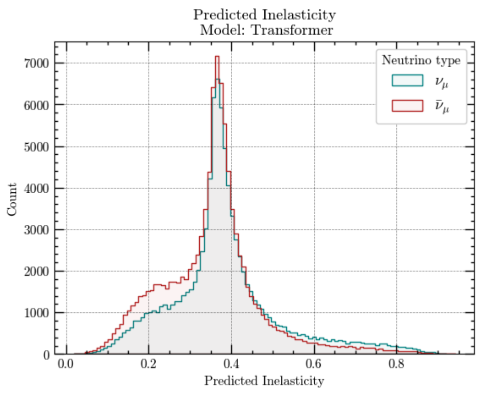


FIGURE 5.12: True Inelasticity Distributions in data for neutrinos and anti-neutrinos



(A) DynEdge reconstructed Inelasticity Distributions for neutrinos and anti-neutrinos



(B) Transformer reconstructed Inelasticity Distributions for neutrinos and anti-neutrinos

FIGURE 5.13: reconstructed Inelasticity Distributions for neutrinos and anti-neutrinos

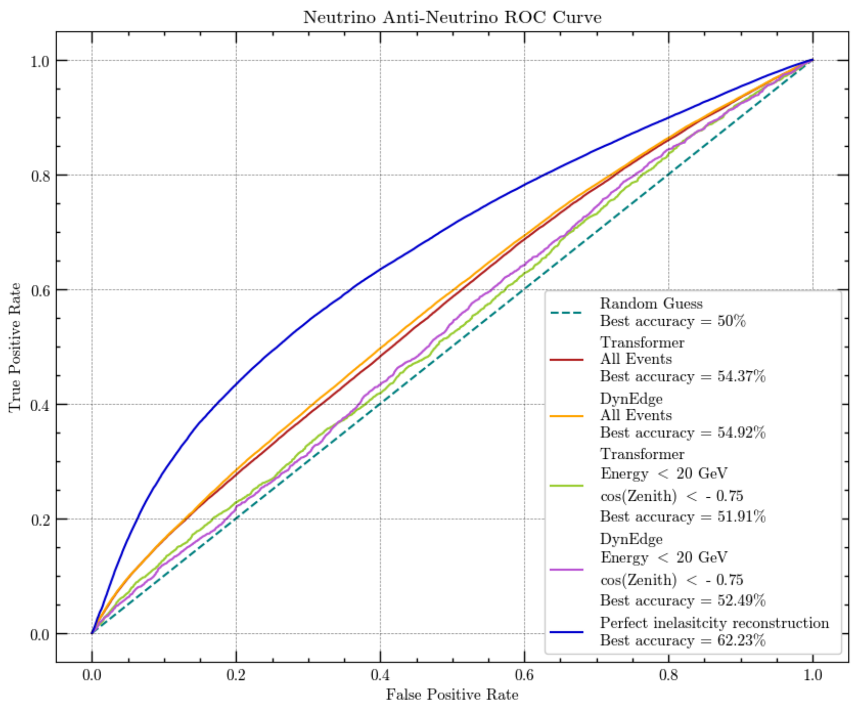


FIGURE 5.14: ROC curve for neutrino anti-neutrino using DynEdge and Transformer inelasticity reconstruction for all event and for events in matter effect region. The ROC curve for the perfect inelasticity reconstruction is also plotted for comparison

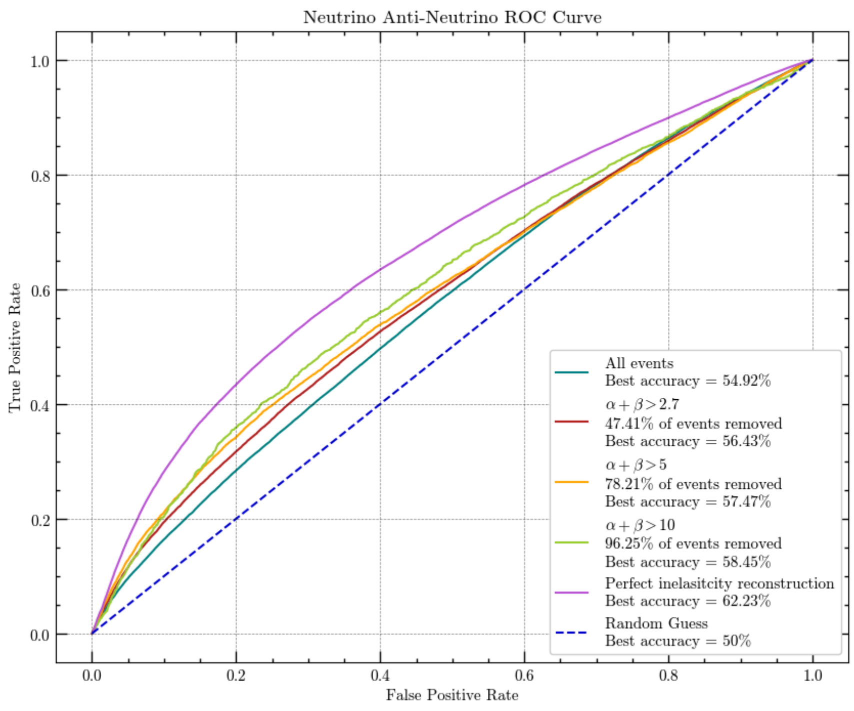


FIGURE 5.15: ROC curve for neutrino anti-neutrino using DynEdge inelasticity reconstruction for all events, all events with alpha + beta > 5 and all events with alpha + beta > 10

5.3.2 Neutrino vs. Anti-neutrino Classification in the Matter Effect Region

Upon analyzing Figure 5.16, it becomes evident that differentiating between neutrinos and anti-neutrinos in the matter effect region is a more intricate task compared to the analysis involving the entire dataset. Specifically, when juxtaposed with Figure 5.13, the inelasticity distributions for neutrinos and anti-neutrinos do not present any marked distinctions.

This observation is further corroborated by Figure 5.14. The following table summarizes the balanced accuracy achieved by DynEdge and the Transformer model:

	DynEdge	**Transformer**
Matter Effect Region	52.49%	51.91%
Entire Dataset	54.92%	53.37%

TABLE 5.2: Balanced Accuracy Comparisons

Moreover, for the comprehensive dataset, accuracy can be enhanced by making cuts based on $\alpha + \beta$. The repercussions of such cuts in the matter effect region can be visualized in Figure 5.17. By implementing a cut of $\alpha + \beta$ at 2.7, the best balanced accuracy climbs to 55.06%, albeit at the cost of excluding 68.22% of the events.

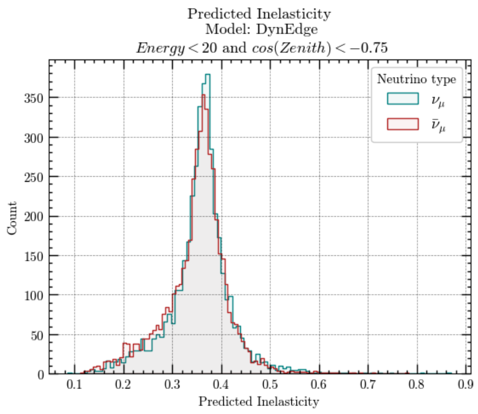
Cut Value	**Balanced Accuracy**	**Events Excluded**
No cut	52.49%	0%
2.7	55.06%	68.22%
4	54.45%	93.49%

TABLE 5.3: Effects of $\alpha + \beta$ Cuts on Accuracy and Event Exclusion in the matter effect region

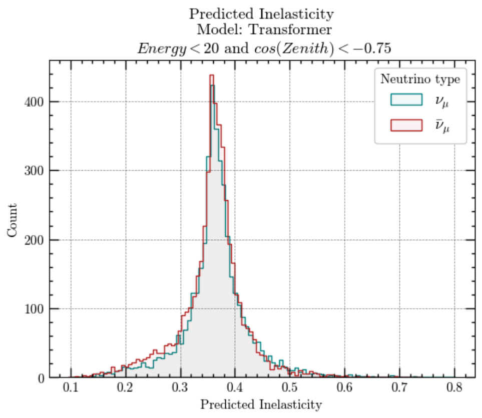
Strikingly, intensifying the cuts on $\alpha + \beta$ appears to degrade accuracy while drastically reducing event counts. A plausible rationale is the heightened difficulty associated with reconstructing high inelasticity events compared to their low inelasticity counterparts. Intuitively, the capability to reconstruct inelasticity might vary based on its actual value. While low inelasticity events are predominantly influenced by the track, high inelasticity events are cascade-dominated. A possible challenge in reconstructing high inelasticity events is the cascade's potential to mask a low-energy muon. Consequently, intensifying cuts on $\alpha + \beta$ may disproportionately eliminate high inelasticity neutrinos, diminishing accuracy, especially since the extremes of inelasticity provide the most distinction between neutrinos and anti-neutrinos.

Furthermore, even with the application of cuts in the matter effect region, accuracy levels don't reach the benchmarks set by the whole dataset. As depicted in Figure 5.18, increasing the energy threshold correspondingly elevates balanced accuracy. The cut-off for $\alpha + \beta$ at 5 doesn't extend below 20 GeV due to the scarcity of events, leading to erratic accuracy fluctuations as single classifications gain disproportionate importance. This phenomenon might explain the accuracy spike for the $\alpha + \beta$ cut at 2.7, observed around 10 GeV.

In contrast, Figure 5.19 portrays a scenario akin to Figure 5.18, with accuracy calculations for events surpassing the energy threshold. As lower energy events are excluded, the overall accuracy experiences an upswing. However, noteworthy aspects emerge: upon eliminating events below 60 GeV, lenient cuts on $\alpha + \beta$ cease to enhance accuracy, and a cut of 5 actually diminishes it. Another intriguing trend is the peak accuracy for the entire dataset (with no cuts) at 140 GeV, which subsequently declines. This dip might be attributed to the increased likelihood of muons escaping the detector at elevated energies, complicating inelasticity reconstruction. Yet, this declining accuracy trend seems to be counteracted by applying cuts on $\alpha + \beta$.



(A) DynEdge reconstructed Inelasticity Distributions for neutrinos and anti-neutrinos in the matter effect region



(B) Transformer reconstructed Inelasticity Distributions for neutrinos and anti-neutrinos in the matter effect region

FIGURE 5.16: Reconstructed Inelasticity Distributions for neutrinos and anti-neutrinos where $\alpha + \beta > 5$

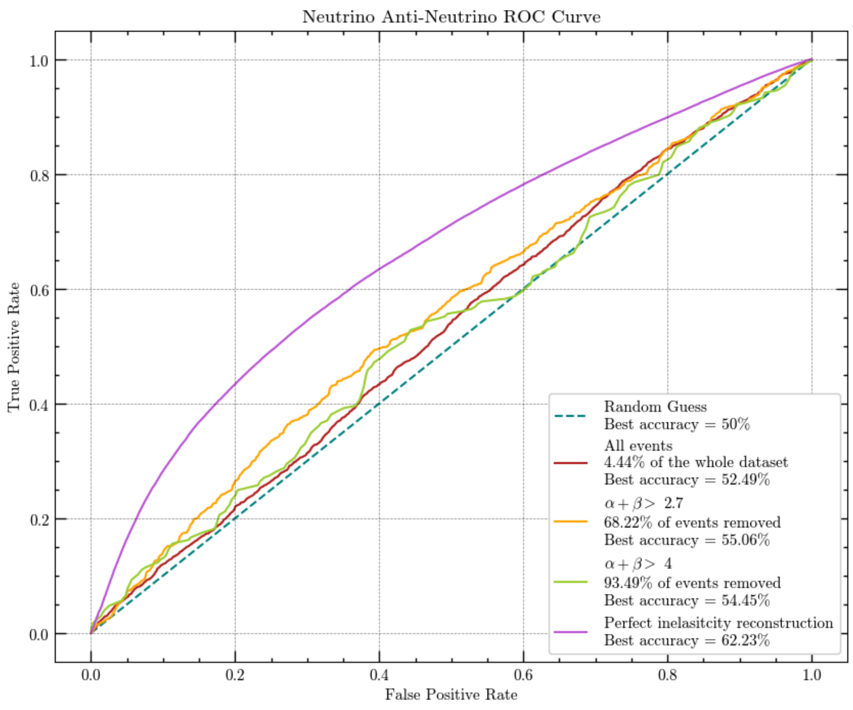


FIGURE 5.17: ROC curve for neutrino anti-neutrino using DynEdge inelasticity reconstruction in the matter effect region with alpha + beta cuts

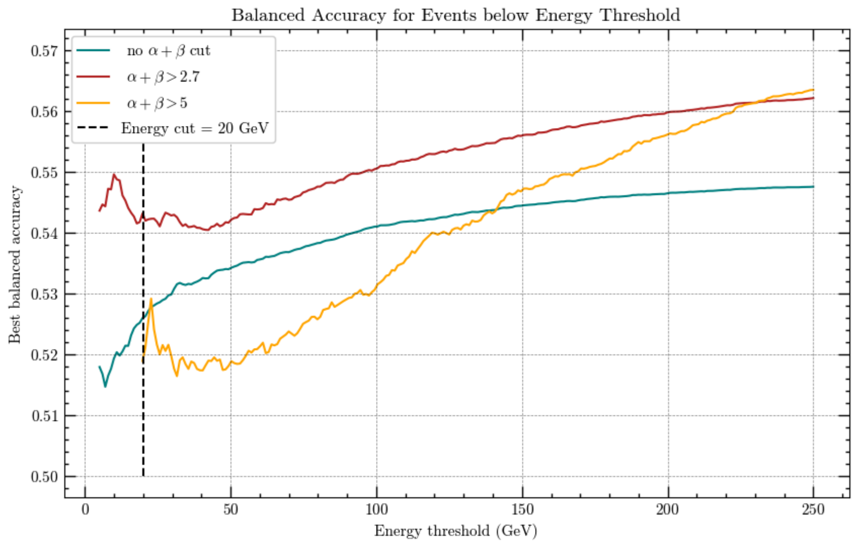


FIGURE 5.18: Best accuracy calculated from True Positive Rate and False Positive Rate for events **below** the Energy threshold showing that as energy decreases so does the accuracy of predicting neutrino or anti-neutrino

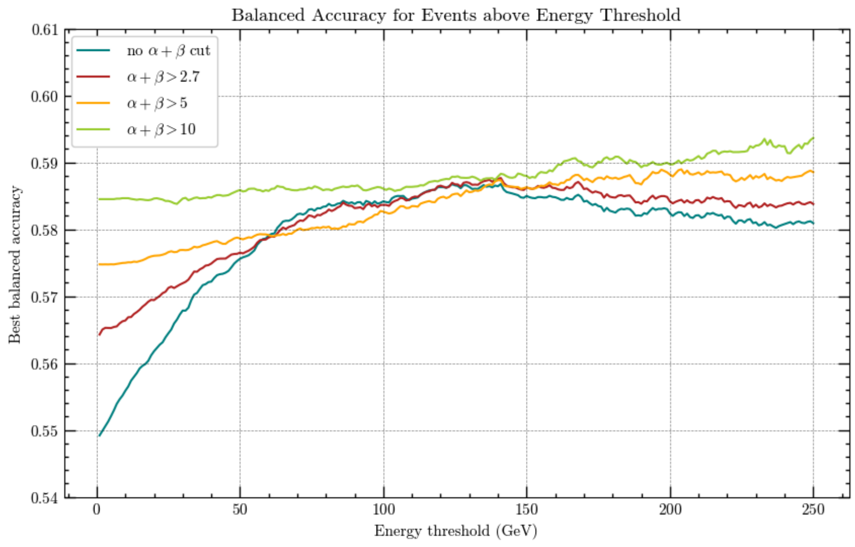


FIGURE 5.19: Best accuracy calculated from True Positive Rate and False Positive Rate for events **above** the Energy threshold showing that as energy decreases so does the accuracy of predicting neutrino or anti-neutrino

Chapter 6

Future Work

6.1 Further NMO analysis

A natural progression of my thesis would be to reevaluate Marc Jaquart's NMO sensitivity calculations. Specifically, one should consider the impact of my inelasticity reconstructions on the new QUESO event selection.

In undertaking such a study, a primary objective would be to determine the optimal inelasticity threshold. This threshold should serve two purposes: maximizing statistical power and ensuring a balanced trade-off between filtering out poorly reconstructed events and retaining a sufficient number of events for meaningful analysis.

Furthermore, this analysis should incorporate a muon neutrino and muon anti-neutrino charge current events classifier. The goal here is to curate a dataset that closely mirrors what one might expect from real-world data. The reason for this is to examine what contamination from other interaction types would affect the sensitivity as this thesis only has dealt with a pure muon neutrino and muon anti-neutrino charge current events. It is plausible to hypothesize a correlation between the precision of inelasticity reconstruction and the confidence with which a classifier can identify an event as a muon neutrino CC event which could result in an unbalanced contamination for high and low inelasticity events.

one could fear that the higher inelasticity events which have less of a track-like signature are more likely to be misidentified which would further erode the accuracy of the neutrino anti-neutrino classification.

Another aspect to consider in this analysis is whether there's a relationship between poorer inelasticity reconstructions and less accurate energy and zenith reconstructions. Some events might have inadequately reconstructed inelasticity, making it challenging to differentiate between a neutrino or an anti-neutrino, but their energy and zenith could still be reconstructed with enough precision for relevance in an NMO analysis.

6.2 Transformer Improvements

6.2.1 Performance

In evaluating the performance of the Transformer model against DynEdge, it became evident that the Transformer slightly underperformed in inelasticity reconstruction. Given that a model of similar calibre secured second place in the Kaggle competition, my expectations for the Transformer model were understandably higher. A closer examination revealed several potential reasons for this deviation from expectation.

Arguably, the most dominant factor is data availability. While the Kaggle dataset was expansive, my training dataset was inherently more constrained. The original dataset I utilized served broader purposes, encompassing a multitude of particles and even pure noise events. Although this sounds extensive, the dataset dwindles substantially when narrowed down to exclusively muon neutrinos and charge current events. For context, my dataset, containing 200,000 entries, pales in comparison to the Kaggle dataset's 13 million. The Kaggle dataset's vastness allowed it to represent a diverse array of events and interactions. In contrast, my dataset was limited to a singular interaction type, focusing solely on neutrinos and anti-neutrinos, which, as most are aware, exhibit striking similarities. To put it into perspective, my training data constituted a mere 1.5% of the Kaggle dataset.

This disparity in data volume might be the linchpin explaining why the Transformer model didn't outshine DynEdge as it did in the Kaggle context. This hypothesis gains traction when we consider my experiments related to the Transformer's size. The Kaggle model authors offered three size presets for their model: tiny, small, and big. Their trials indicated a direct correlation between model size and performance, with the 'big' variant being the most effective. Contrarily, my experiments identified the 'tiny' model as the top performer, with performance deteriorating as the model size expanded. This divergence further underscores the potential role of dataset size in influencing outcomes. It is a well-established fact in the machine-learning community that Transformer architectures exhibit an insatiable appetite for data.

6.2.2 Graph Features Extractor

The runner-up in the Kaggle competition incorporated an option in their model to use a DynEdge-based learnable embedding in addition to the other embeddings detailed in the transformer section (4.6). In my approach, I chose not to implement this feature to maintain a clear distinction between the Transformer and GNN comparison. However, the claim from the competition runner-up suggests that the DynEdge-based learnable embedding enhances their results. Thus, exploring this avenue or other graph-based embeddings could prove beneficial for further refinement.

6.2.3 Learnable Noise Attention Bias

The training parameters used by the Kaggle competition's runner-up for their Transformer model differ slightly from those I employed. A notable divergence is in the noise cleaning techniques. While they utilized simpler methods, such as SRT, I incorporated the more advanced DynEdge noise cleaning. Specifically, DynEdge cleaning involves a straightforward cut on the prediction value from DynEdge for all DOM hits with a value exceeding 0.7.

A factor to consider when applying this kind of noise cleaning cut is the optimal value threshold. The ideal cut-off might be dependent on the event's size. For instance, smaller events might benefit from a more lenient cut to ensure the preservation of the few actual hits. Conversely, larger events might necessitate stricter cuts. Post-cut, all hits are treated uniformly, even though some hits have a higher likelihood of being noise.

Transformer models, with their inherent architecture, are exceptionally equipped to address these noise event challenges. Leveraging the attention mechanism, I suggest introducing a "noise attention bias". This bias would employ the noise classification score within the attention mechanism, thereby reducing attention to hits more likely to be noise. It may be necessary to subject this noise classification score to a transformation before its incorporation. Another challenge this could address is the requirement for padding tokens. Instead of adding these tokens to align with the length of the longest event in a batch, adjusting the cut-off for the noise classification score for each event could standardize event lengths within the batch.

6.3 Improvements in Inelasticity Reconstruction

6.3.1 Data Selection

One of the pivotal enhancements for inelasticity reconstruction lies in identifying distinguishable metrics between well-reconstructed and poorly-reconstructed events. By omitting events with a high likelihood of poor reconstruction, models can prioritize events where accurate inelasticity reconstruction is feasible. While it's crucial for models to discern between high-quality and subpar events, the current balance may be skewed. Currently, over 50% of the events are of lower quality. A more balanced distribution might situate poor-quality events at around 15% to 20%.

6.3.2 Change of Parametrization

Occasionally, machine learning models can achieve improved performance by merely altering data representation, a process termed 'feature engineering'. This transformation isn't exclusive to input features but extends to target labels as well. One such strategy involves reparametrizing the beta distribution, transitioning from parameters α and β to the mean $\frac{\alpha}{\alpha+\beta}$ and $\alpha + \beta$. This modified representation might be more intuitive for the model to interpret.

6.3.3 $\alpha + \beta$ cut

Future considerations should evaluate whether adjusting the cut on $\alpha + \beta$ optimizes the accuracy of neutrino and anti-neutrino classification. While this method is effective in eliminating uncertain predictions, it might not be the ultimate strategy for enhancing accuracy. As observed, high inelasticity events might inherently pose greater reconstruction challenges than their low inelasticity counterparts. Thus, a tailored cut might yield better results.

6.4 Considerations on Binned Inelasticity

It's worthwhile to question if the beta distribution aptly captures the inherent uncertainties during inelasticity reconstruction. Although the beta distribution seems competent for a significant majority of events, there are reservations about how well it describes the uncertainty of some events, especially for events the model is more certain about. This sentiment is echoed by Figure 5.7, where models seem reluctant to predict inelasticities nearing 0 or 1.

An alternative approach might involve a discrete distribution, where inelasticities are categorized into bins. Subsequently, a conventional cross-entropy classification loss function can be applied. This method, while offering the flexibility to emulate any distribution, is bound by the precision of inelasticity binning. Yet, given the beta distribution's nuances observed in Figure 5.9, a bin width of 0.05 (translating to 20 bins) might suffice.

It's noteworthy that a similar approach was employed for zenith reconstruction in the Kaggle competition, yielding promising results.

6.5 Direct Neutrino vs. Anti-Neutrino Classification

An alternative to the two-step process of inelasticity reconstruction followed by neutrino and anti-neutrino classification based on inelasticity is a direct classification of events.

Direct classification might offer models a more targeted approach, zeroing in on the nuances distinguishing neutrinos from anti-neutrinos. Currently, models weigh all inelasticities equally. However, some inelasticities, especially those closer to 0.5 and the threshold, might be more critical as a small improvement might push them to the "correct" side of the threshold compared to those at the extremes.

Conversely, if inelasticity is the sole discerning factor between neutrino and anti-neutrino events, it should theoretically serve as a superior training label compared to a binary neutrino vs. anti-neutrino classification. The latter, devoid of additional information, might only introduce noise. If no unique features differentiate neutrinos from anti-neutrinos, the inelasticity-centric method would likely be the most effective.

Chapter 7

Conclusion

In this research, I introduced a novel transformer-based model, drawing inspiration from the second-place finisher in the Kaggle competition titled "Neutrinos in Deep Ice".

A distinctive feature of my approach is the utilization of a cross-entropy-based loss function. This function employs the beta distribution to make predictions on continuous variables ranging from 0 to 1, such as inelasticity.

By combining this new loss function with both my transformer model and the DynEdge model, I successfully reconstructed the inelasticity for ν_μ and $\bar{\nu}_\mu$charged current events using simulated data for the IceCube upgrade. This achievement enabled me to partially distinguish between neutrinos and anti-neutrinos. In terms of quantitative results, the best balanced accuracy I achieved for the entire dataset was 54.92%, approaching the maximum of 62.23%. Notably, by applying cuts on the predicted parameters of the beta distributions, I enhanced the balanced accuracy to 56.43%. This enhancement, however, came at the expense of excluding 47.41% of the events. A stricter cut further improved accuracy to 58.45%, but this necessitated the exclusion of a substantial 96.25% of the events.

An essential aspect of neutrino analysis revolves around the Neutrino mass ordering problem. This problem primarily concerns low-energy neutrinos, specifically those with 20 GeV or less. These neutrinos have traversed a significant portion of the Earth, exhibiting slight oscillation pattern variations between neutrinos and anti-neutrinos due to coherent forward scattering. These nuances offer potential solutions to the neutrino mass ordering problem. For events in my dataset that met these criteria, the best balanced accuracy stood at 52.49%. This accuracy has the potential for enhancement to 55.06% with the application of cuts, albeit with a trade-off of losing 68.22% of the events.

While these results are promising, additional research is essential to assess their impact on the Neutrino Mass Ordering sensitivity for the IceCube upgrade.

Regarding the broader applications of my transformer model, it holds considerable promise for future analyses. The Kaggle competition underscored the potential of transformer models as a robust tool for reconstructing IceCube data. My model, supplemented with proposed enhancements like the Learnable Noise Attention Bias, the graph feature extractor, and hyperparameter tweaking, has garnered interest for potential use in subsequent analyses.

Appendix A

Transformer neutrino anti-neutrino classification using inelasticity results

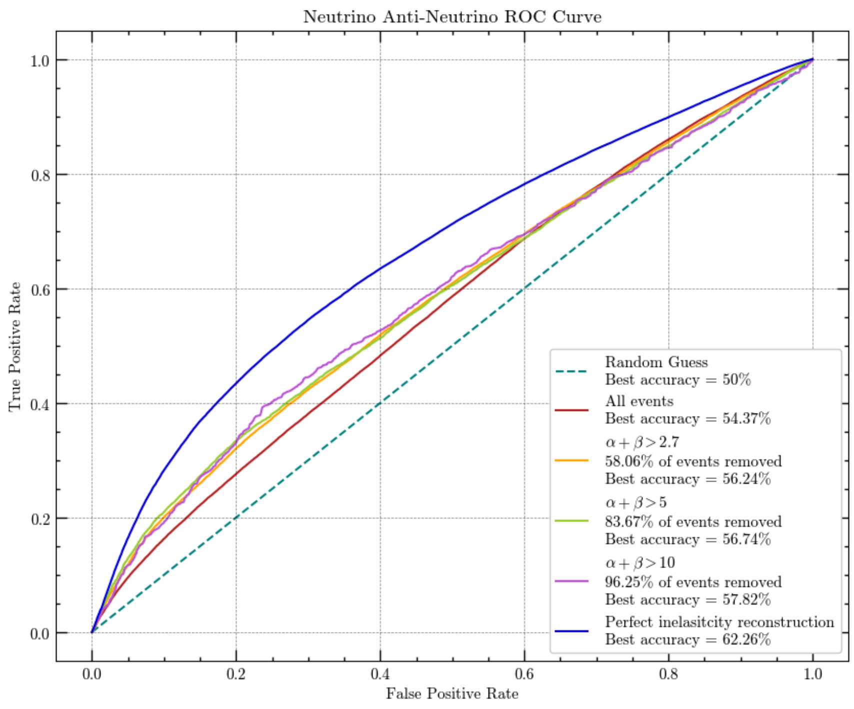


FIGURE A.1: ROC curve for neutrino anti-neutrino using DynEdge inelasticity reconstruction for all events, all events with alpha + beta > 5 and all events with alpha + beta > 10

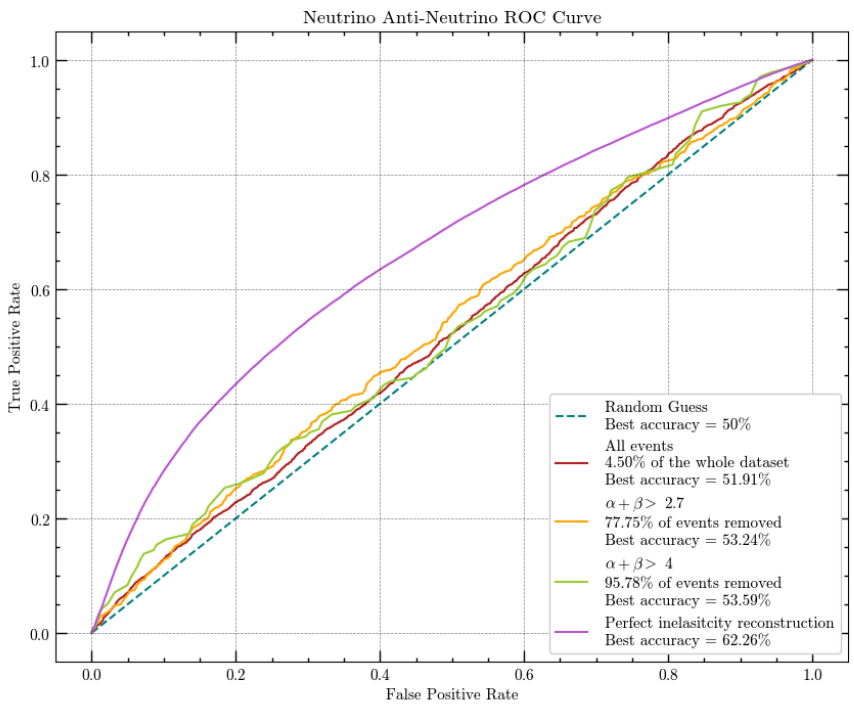


FIGURE A.2: ROC curve for neutrino anti-neutrino using DynEdge inelasticity reconstruction in the matter effect region with alpha + beta cuts

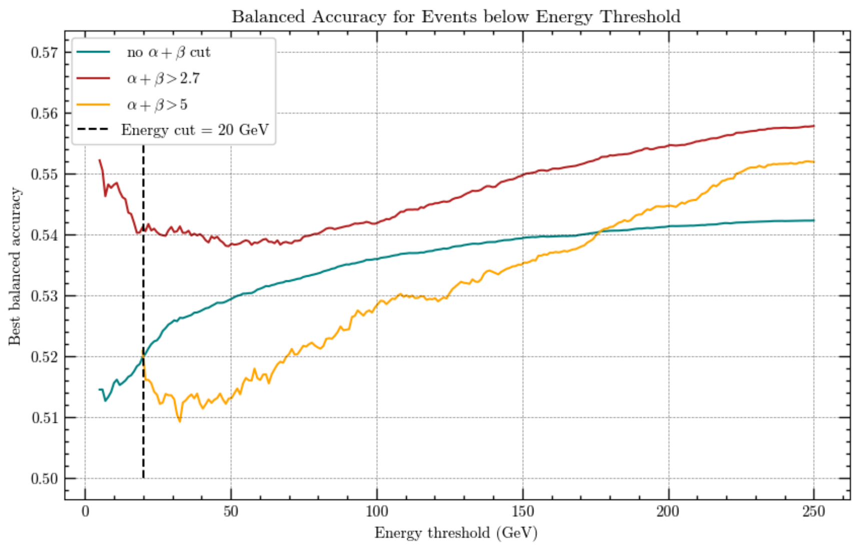


FIGURE A.3: Best accuracy calculated from True Positive Rate and False Positive Rate for events **below** the Energy threshold showing that as energy decreases so does the accuracy of predicting neutrino or anti-neutrino

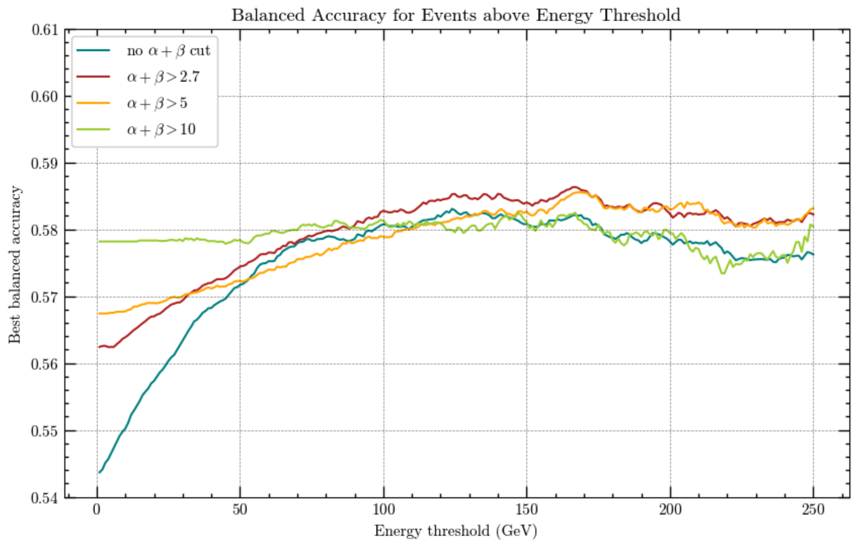


FIGURE A.4: Best accuracy calculated from True Positive Rate and False Positive Rate for events **above** the Energy threshold showing that as energy decreases so does the accuracy of predicting neutrino or anti-neutrino

Bibliography

[1] R. L. Workman et al. "Review of Particle Physics". In: *PTEP* 2022 (2022), p. 083C01. DOI: 10.1093/ptep/ptac097 (cit. on p. 2).

[2] Cush MissMJ. *Standard Model of Elementary Particles*. URL: https://en.wikipedia.org/wiki/File:Standard_Model_of_Elementary_Particles.svg (cit. on p. 2).

[3] Mark Thomson. *Modern Particle Physics*. Cambridge University Press, 2013. DOI: 10.1017/CBO9781139525367 (cit. on pp. 5, 9, 10).

[4] P. Kyberd et al. *nuSTORM: Neutrinos from STORed Muons*. 2012. arXiv: 1206.0294 [hep-ex] (cit. on p. 5).

[5] C. Giganti, S. Lavignac, and M. Zito. "Neutrino oscillations: The rise of the PMNS paradigm". In: *Progress in Particle and Nuclear Physics* 98 (2018), pp. 1–54. DOI: 10.1016/j.ppnp.2017.10.001. URL: https://doi.org/10.1016%2Fj.ppnp.2017.10.001 (cit. on p. 6).

[6] C. Giganti, S. Lavignac, and M. Zito. "Neutrino oscillations: The rise of the PMNS paradigm". In: *Progress in Particle and Nuclear Physics* 98 (2018), pp. 1–54. ISSN: 0146-6410. DOI: https://doi.org/10.1016/j.ppnp.2017.10.001. URL: https://www.sciencedirect.com/science/article/pii/S014664101730087X (cit. on p. 6).

[7] Y. Fukuda et al. "Evidence for Oscillation of Atmospheric Neutrinos". In: *Phys. Rev. Lett.* 81 (8 1998), pp. 1562–1567. DOI: 10.1103/PhysRevLett.81.1562. URL: https://link.aps.org/doi/10.1103/PhysRevLett.81.1562 (cit. on p. 7).

[8] Q. R. Ahmad et al. "Direct Evidence for Neutrino Flavor Transformation from Neutral-Current Interactions in the Sudbury Neutrino Observatory". In: *Phys. Rev. Lett.* 89 (1 2002), p. 011301. DOI: 10.1103/PhysRevLett.89.011301. URL: https://link.aps.org/doi/10.1103/PhysRevLett.89.011301 (cit. on p. 7).

[9] Marc Jacquart. "Neutrino mass ordering sensitivity of the IceCube Upgrade, enhanced using inelasticity". MA thesis. Niels Bohr Institute, 2022 (cit. on pp. 8, 14, 15, 20, 39).

[10] D F Cowen and (forthe IceCube Collaboration). "Tau Neutrinos in IceCube". In: *Journal of Physics: Conference Series* 60.1 (2007), p. 227. DOI: 10.1088/1742-6596/60/1/048. URL: https://dx.doi.org/10.1088/1742-6596/60/1/048 (cit. on p. 11).

[11] DONALD E. GROOM, NIKOLAI V. MOKHOV, and SERGEI I. STRIGANOV. "MUON STOPPING POWER AND RANGE TABLES 10 MeV–100 TeV". In: *Atomic Data and Nuclear Data Tables* 78.2 (2001), pp. 183–356. ISSN: 0092-640X. DOI: https://doi.org/10.1006/adnd.2001.0861. URL: https://www.sciencedirect.com/science/article/pii/S0092640X01908617 (cit. on p. 12).

[12] Jorge Prado González. "Event Selection for the IceCube Upgrade". MA thesis. Niels Bohr Institute, 2023 (cit. on pp. 17, 20).

[13] I. Collaboration. *New sensor designs shown at string depths*. Accessed: 4/09/2023. 2023. URL: https://icecube.wisc.edu/gallery/nsf-approves-funding-for-icecube-upgrade/ (cit. on p. 18).

[14] Etienne Bourbeau. "Measurement of Tau Neutrino Appearance in 8 Years of IceCube Data". PhD thesis. The Niels Bohr Institute, Faculty of Science, University of Copenhagen, 2021 (cit. on p. 19).

[15] Larry Hardesty. *Explained: Neural networks Ballyhooed artificial-intelligence technique known as "deep learning" revives 70-year-old idea.* April 2017 (accessed 27/02 2023). URL: `https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414` (cit. on p. 24).

[16] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006 (cit. on p. 25).

[17] Pankaj Mehta et al. "A high-bias, low-variance introduction to Machine Learning for physicists". In: *Physics Reports* 810 (2019), pp. 1–124. DOI: `10.1016/j.physrep.2019.03.001`. URL: `https://doi.org/10.1016%2Fj.physrep.2019.03.001` (cit. on p. 27).

[18] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. DOI: `10.48550/ARXIV.1412.6980`. URL: `https://arxiv.org/abs/1412.6980` (cit. on p. 27).

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105 (cit. on p. 28).

[20] William L. Hamilton. "Graph Representation Learning". In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* 14.3 (), pp. 47–54 (cit. on p. 29).

[21] R. Abbasi et al. "Graph Neural Networks for low-energy event classification & reconstruction in IceCube". In: *Journal of Instrumentation* 17.11 (2022), P11003. DOI: `10.1088/1748-0221/17/11/p11003`. URL: `https://doi.org/10.1088%2F1748-0221%2F17%2F11%2Fp11003` (cit. on pp. 29, 30).

[22] Yue Wang et al. *Dynamic Graph CNN for Learning on Point Clouds*. 2018. DOI: `10.48550/ARXIV.1801.07829`. URL: `https://arxiv.org/abs/1801.07829` (cit. on p. 29).

[23] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. "Self-Attention with Relative Position Representations". In: *CoRR* abs/1803.02155 (2018). arXiv: `1803.02155`. URL: `http://arxiv.org/abs/1803.02155` (cit. on pp. 31, 34).

[24] Zhiliang Peng et al. *BEiT v2: Masked Image Modeling with Vector-Quantized Visual Tokenizers*. 2022. arXiv: `2208.06366 [cs.CV]` (cit. on pp. 31, 35).

[25] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017) (cit. on pp. 31, 32).

[26] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: `1512.03385 [cs.CV]` (cit. on p. 34).

[27] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. *Layer Normalization*. 2016. arXiv: `1607.06450 [stat.ML]` (cit. on p. 34).

[28] Philipp Eller Rasmus Ørsøe Sohier Dane Ashley Chow Lukas Heinrich. *IceCube - Neutrinos in Deep Ice*. 2023. URL: `https://kaggle.com/competitions/icecube-neutrinos-in-deep-ice` (cit. on p. 35).

[29] Maxim Shugaev and drHB. *Title of the Discussion*. Accessed: 01-09-2023. 2023. URL: `https://www.kaggle.com/competitions/icecube-neutrinos-in-deep-ice/discussion/402882` (cit. on p. 35).