NuBench: An Open Benchmark for Deep Learning-Based Event Reconstruction in Neutrino Telescopes

02/06/2026, IRN Neutrino Meeting

Iván Mozún Mateo

Overview

1. Neutrino telescopes and reconstruction
2. NuBench datasets
3. Simulation and processing
4. Deep Learning models
5. Results
6. Benchmarking conclusions

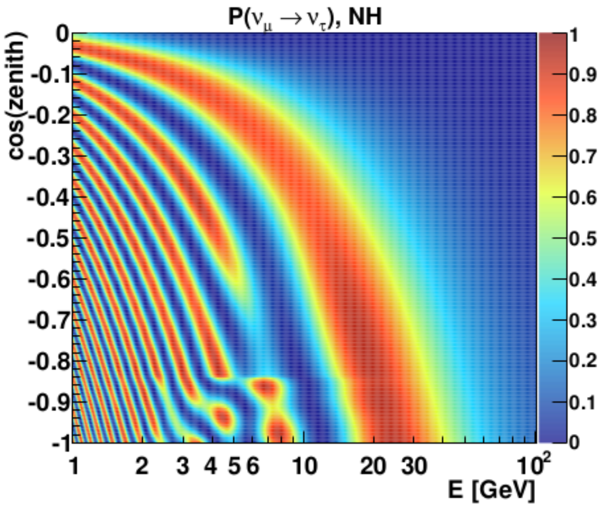
Neutrino telescopes

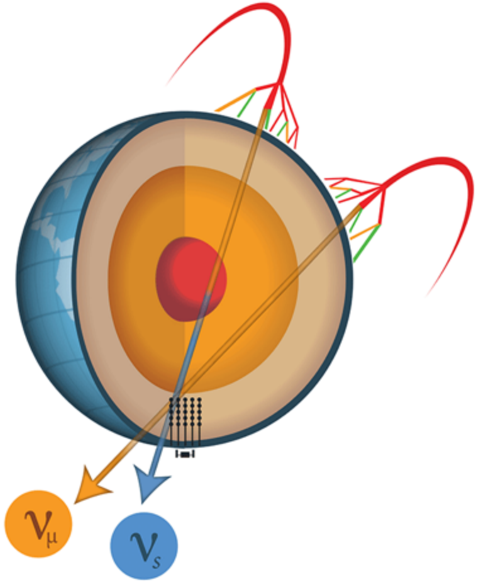
Big arrays of photo-sensors placed in large volumes of water or ice

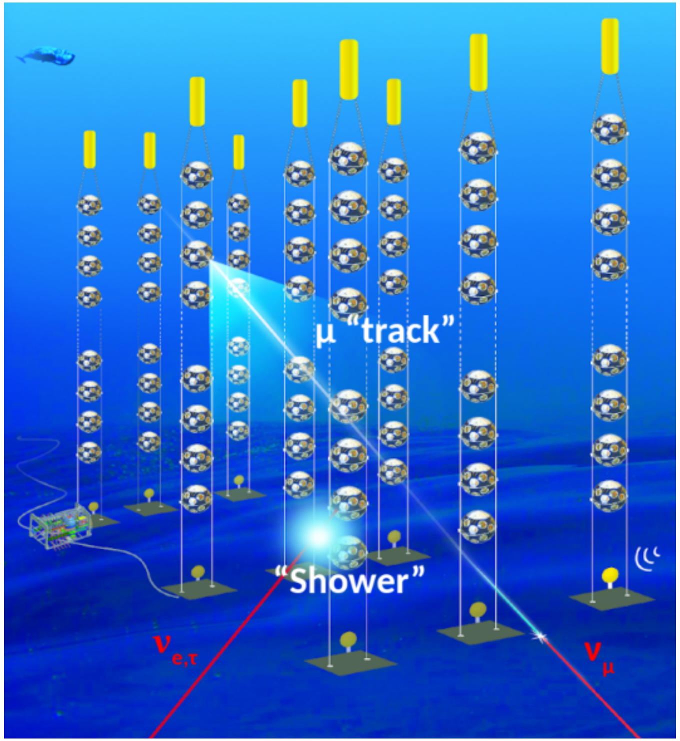
1) Neutrino interactions are very rare ⟶ Needs to maximize sensitivity

2) Detect Cherenkov radiation from outgoing charged particles

3) Sensitivity to oscillation parameters and squared mass differences





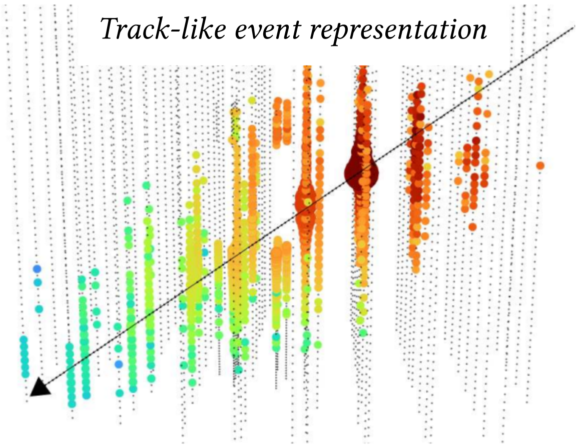


Reconstruction in neutrino telescopes

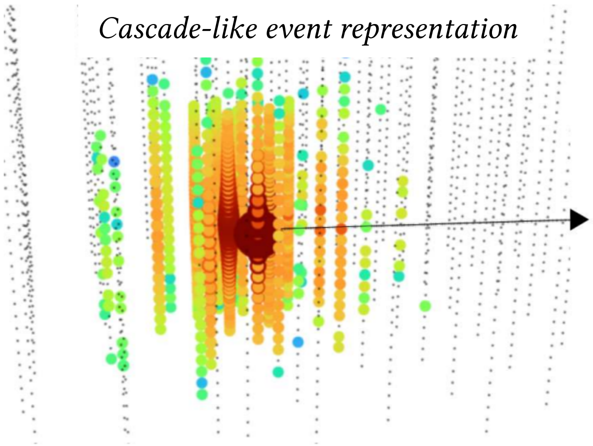
Likelihood-fit based methods
1) Define hypothesis of what's observed
2) Maximize the likelihood given the hypothesis

$$\mathcal{L}(x|\theta) = \prod_{i=1}^{d} A_i(h|\theta) \cdot \left(\prod_{j=1}^{h} p_i(t_j|\theta) \right)$$

Dependence on photons (h) and Optical Modules (d)
Main problem: **complexity** of neutrino interactions



Track-like reconstruction
- Fit to Cherenkov propagation
- Continuous energy loss
- Good directionality
- Problem: stochastic energy losses and containment



Cascade-like reconstruction
- Calorimeter approach
- Better energy measurement and event containment
- Problem: bias from invisible energy

NuBench goals and people

Jinst

PUBLISHED BY IOP PUBLISHING FOR SISSA MEDIALAB

RECEIVED: *November 20, 2025*
ACCEPTED: *April 2, 2026*
PUBLISHED: *May 5, 2026*

TECHNICAL REPORT

NuBench: An open benchmark for deep learning-based event reconstruction in neutrino telescopes

2026 JINST 21 T05001

Rasmus Ørsøe,[a,*] **Stephan Meighen-Berger**,[b,c,d] **Jeffrey Lazar**,[e] **Jorge Prado**,[f] **Iván Mozún-Mateo**,[g] **Aske Rosted**,[h] **Philip Weigel**[i] **and Arturo Llorente Anaya**[a]

[a]*Physik-department, Technische Universität München, D-85748 Garching, Germany*
[b]*School of Physics, The University of Melbourne, Victoria 3010, Australia*
[c]*Center for Cosmology and AstroParticle Physics (CCAPP), Ohio State University, Columbus, OH 43210, U.S.A.*
[d]*University of Iowa, 30 N Dubuque St, Iowa City, IA, 52242, U.S.A.*
[e]*Université Catholique de Louvain, Pl. de l'Université 1, 1348 Ottignies-Louvain-la-Neuve, Belgium*
[f]*IFIC — Instituto de Física Corpuscular (CSIC — Universitat de València), c/Catedrático José Beltrán, 2, 46980 Paterna, Valencia, Spain*
[g]*LPC CAEN, Normandie Univerity, ENSICAEN, UNICAEN, CNRS/IN2P3, 6 boulevard Maréchal Juin, Caen, 14050, France*
[h]*Department of Physics and The International Center for Hadron Astrophysics, Chiba University, Chiba 263-8522, Japan*
[i]*Department of Physics, Massachusetts Institute of Technology, Cambridge, MA 02139, U.S.A.*

E-mail: rasmus.orsoe@tum.de

ABSTRACT: Neutrino telescopes are large-scale detectors designed to observe Cherenkov radiation produced from neutrino interactions in water or ice. They exist to identify extraterrestrial neutrino sources and to probe fundamental questions pertaining to the elusive neutrino itself. A central challenge common across neutrino telescopes is to solve a series of inverse problems known as event reconstruction, which seeks to resolve properties of the incident neutrino, based on the detected Cherenkov light. In recent times, significant efforts have been made in adapting advances from deep learning research to event reconstruction, as such techniques provide several benefits over traditional methods. While a large degree of similarity in reconstruction needs and low-level data exists, cross-experimental collaboration has been hindered by a lack of diverse open-source datasets for comparing methods.

https://iopscience.iop.org/article/10.1088/1748-0221/21/05/T05001

What did we want to achieve?

We would like to write a paper that compares deep learning based reconstruction and classification algorithms on a series of physics tasks that are of general interest to the community (@GraphNeT workshop)

What is needed?

1) Neutrino telescope simulation: **PROMETHEUS**
2) Deep Learning models
3) Common software: **GraphNeT**
4) Benchmarking and discussion

NuBench team: Rasmus, Stephan, Jeffrey, Jorge, Aske, Phillip, Arturo and myself

NuBench datasets

7 datasets generated with PROMETHEUS corresponding to ~130M neutrino events
Inspiration taken from 6 real neutrino telescopes: different sparsity and size
Energies ranked from 10 GeV to 100 TeV

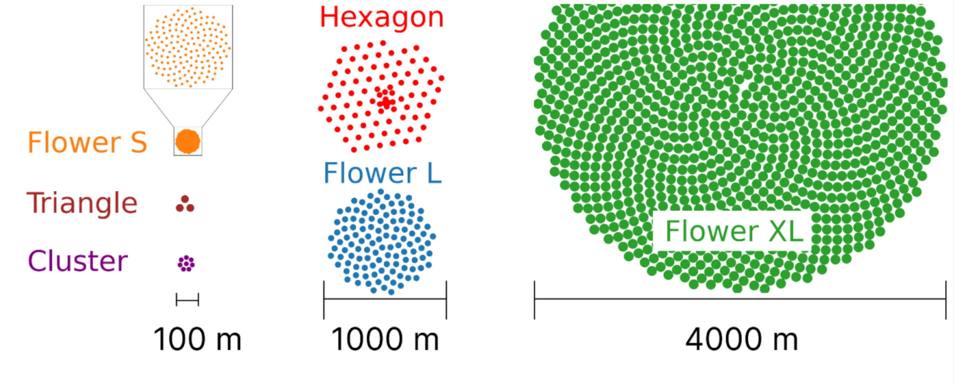


Figure 3. Top-down view of the 6 different detector geometries: `Flower S`, `Flower L`, `Flower XL`, `Triangle`, `Cluster`, and `Hexagon`. Approximate length scales are annotated for comparison.

Table 1. Overview of datasets processed for this work. A total of over 129.7 million events were distributed across 7 datasets with geometries similar to existing or proposed neutrino telescopes. Datasets marked with * are simulated in ice, whereas the remainder is simulated in water.

Dataset	**Events** (millions)	**Inspiration**	$\nu_\mu^{CC}/\nu_\mu^{NC}$ (%)	**Strings/DOMs**	**Energy Range** (GeV)
`Triangle`	23.1	P-ONE	35/65	3/60	$10–10^5$
`Cluster`	22.9	GVD	49/51	8/288	$10–10^5$
`Flower S`	20.5	ORCA	40/60	150/3300	$10–10^3$
`Flower L`	24.0	ARCA	35/65	115/2070	$10–10^5$
`Flower XL`	10.1	TRIDENT	88/12	1211/24220	$10–10^5$
`Hexagon`	20.5	IceCube	48/52	86/5160	$10–10^5$
`Hexagon Ice LE`*	8.6	IceCube	57/43	86/5160	$10–10^3$
Total:	129.7				

NuBench datasets

7 datasets generated with **PROMETHEUS** corresponding to ~130M neutrino events
Inspiration taken from **6 real neutrino telescopes**: different sparsity and size
Energies ranked from 10 GeV to 100 TeV

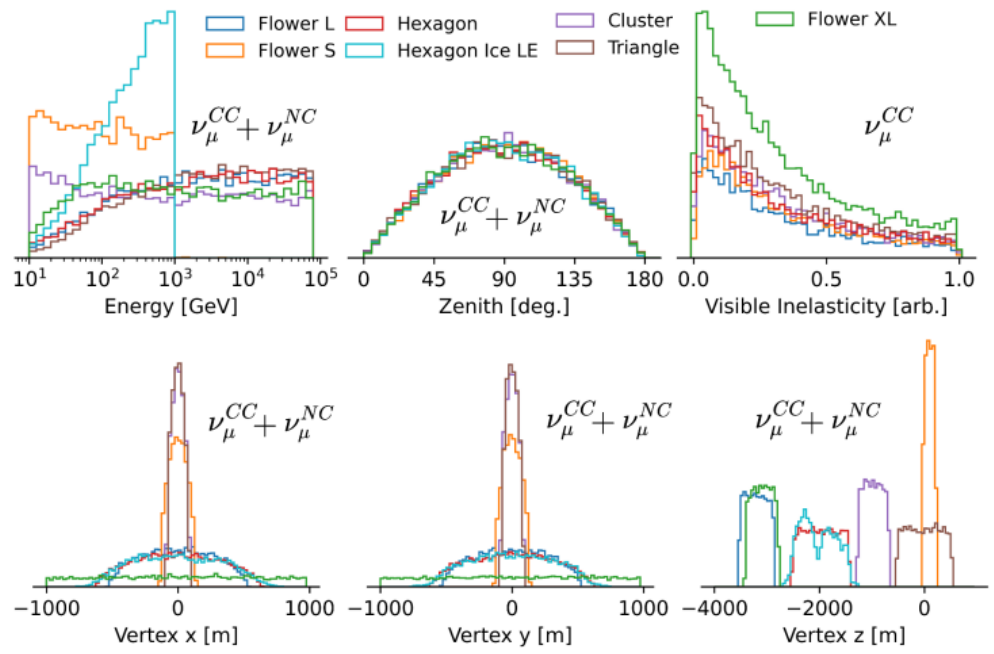


Figure 7. Distributions of neutrino energy, zenith, visible inelasticity, and interaction vertex in the seven NuBench datasets. Note that Hexagon Ice LE is omitted from the visible inelasticity distribution as that dataset does not cover the full numerical range.

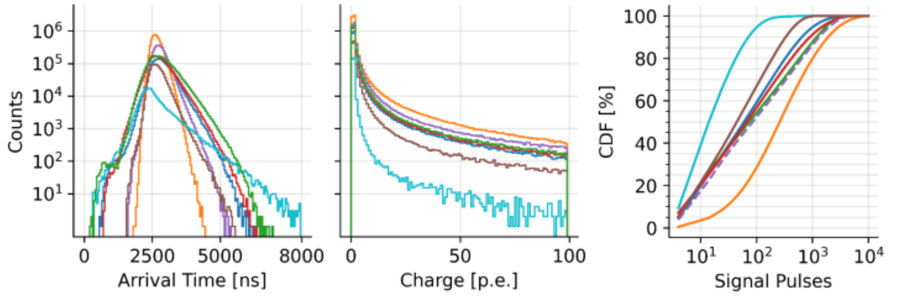
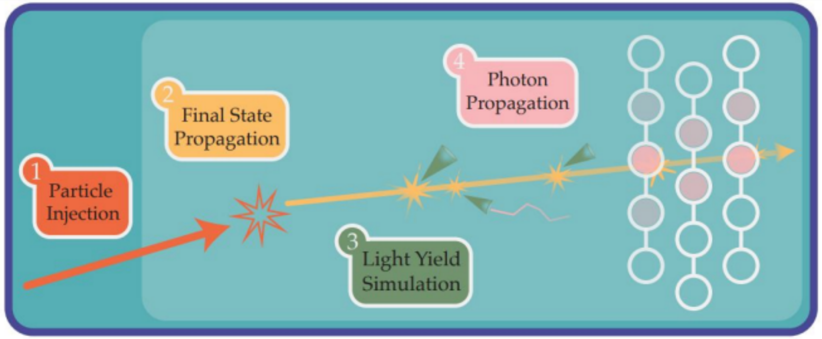


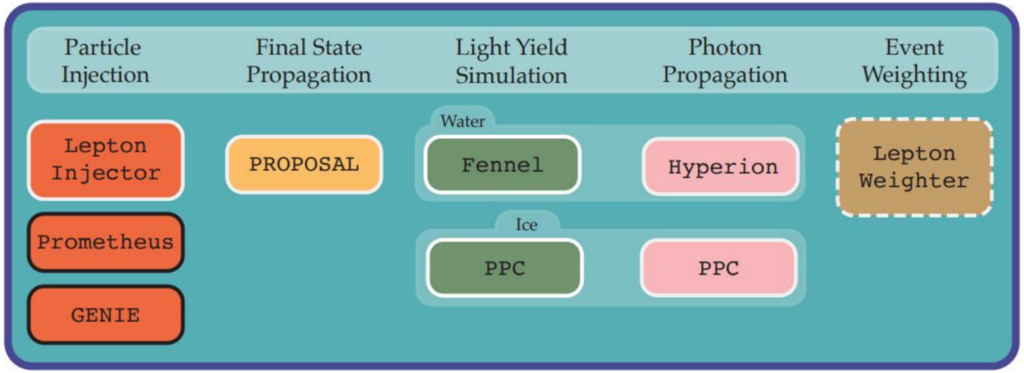
Figure 6. Quantification of pulse-level information in the test set of each dataset in table 1. Left: distribution of pulse arrival time. Middle: distribution of pulse charge. Right: the cumulative percentage of events with respect to the number of signal pulses.

Use low-level information, i.e. individual pulses with position, time and charge information, to reconstruct neutrino energies, direction, interaction vertices, inelasticy and event topology (track or cascade).

Simulation and detector response

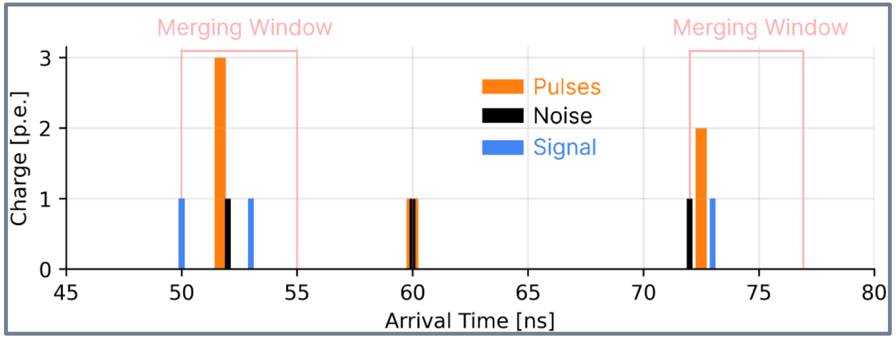
Datasets generated using open software [PROMETHEUS](#)
From neutrinos and their interactions to light propagation and detector response





Detector response

- Individual photon arrival times are linearly shifted into a trigger window of at least 5 μs around the mean arrival time
- Noise is artificially injected around the trigger window
- Pulse: photon-noise merging based on TTS

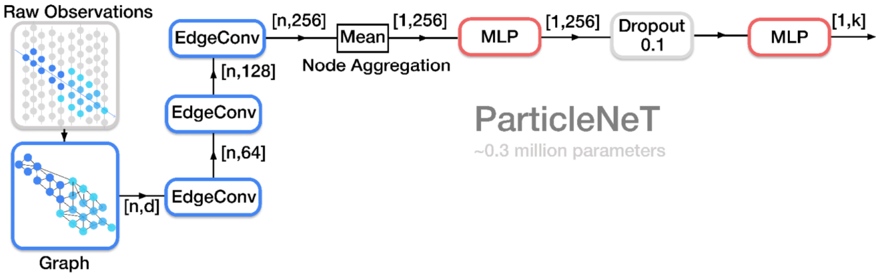


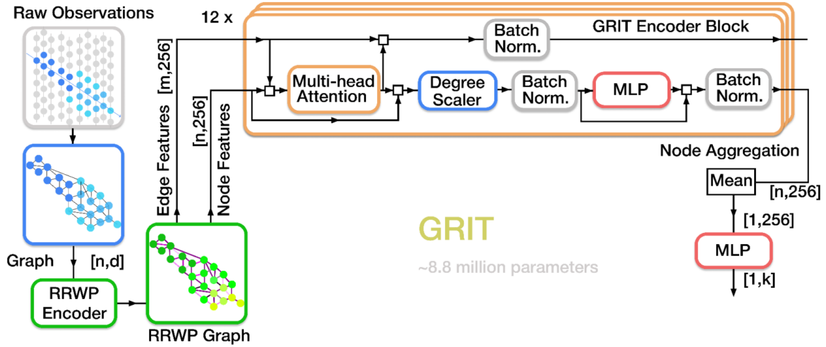
Considerations:

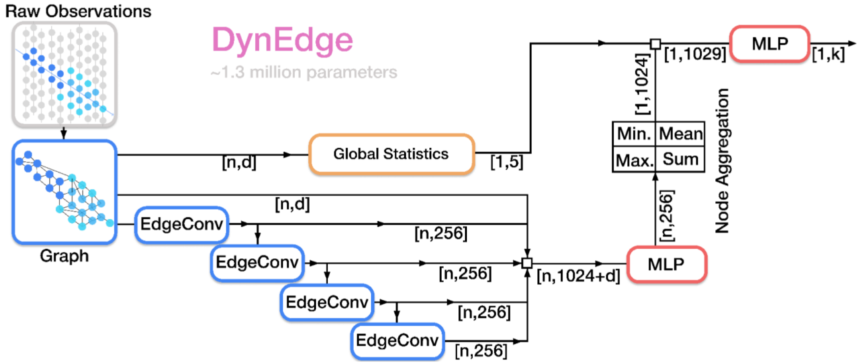
1) No directional information from multi-PMTs OMs
2) Effective OMs with increased photo-detection rate
3) Ice/Water propagation: different OMs angular acceptance

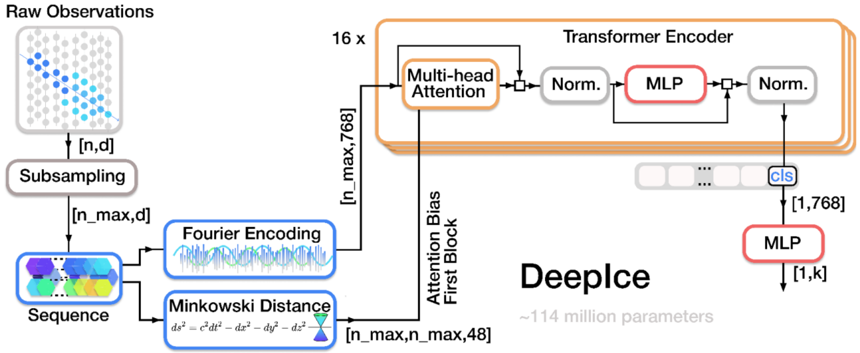
Deep Learning Models

4 different deep learning models considered: ParticleNeT, DynEdge, GRIT and DeepIce
2 GNNs, 1 transformer and 1 hybrid GNN-transformer
From 300k up to 114M parameters









Results

Our goals and questions (we wanted to compare *apples to apples)*
1) Is there any model that performs better across all tasks?
2) Is there any model that performs better across all detectors?
3) What is the best DL architecture? GNNs or transformers?

Benchmarking strategy
- Inject the different DL models into GraphNeT
- Define the telescope geometries
- Use GraphNeT for training and evaluation

Why GraphNeT?
Open software for deep learning in neutrino telescopes
Several models already available
Inputs: detector geometry, dataset, input and model definition

GraphNeT
Deep Learning for Neutrino Telescopes

Usage	Development
JOSS 10.21105/joss.04971	Build passing
DOI 10.5281/zenodo.6720188	Code quality passing
License Apache 2.0	code style black
python | 3.9 | 3.10 | 3.11	maintainability B
version v1.0.0	test coverage 63%

🚀 About

GraphNeT is an open-source Python framework aimed at providing high quality, user friendly, end-to-end functionality to perform reconstruction tasks at neutrino telescopes using deep learning (DL). GraphNeT makes it fast and easy to train complex models that can provide event reconstruction with state-of-the-art performance, for arbitrary detector configurations, with inference times that are orders of magnitude faster than traditional reconstruction techniques.

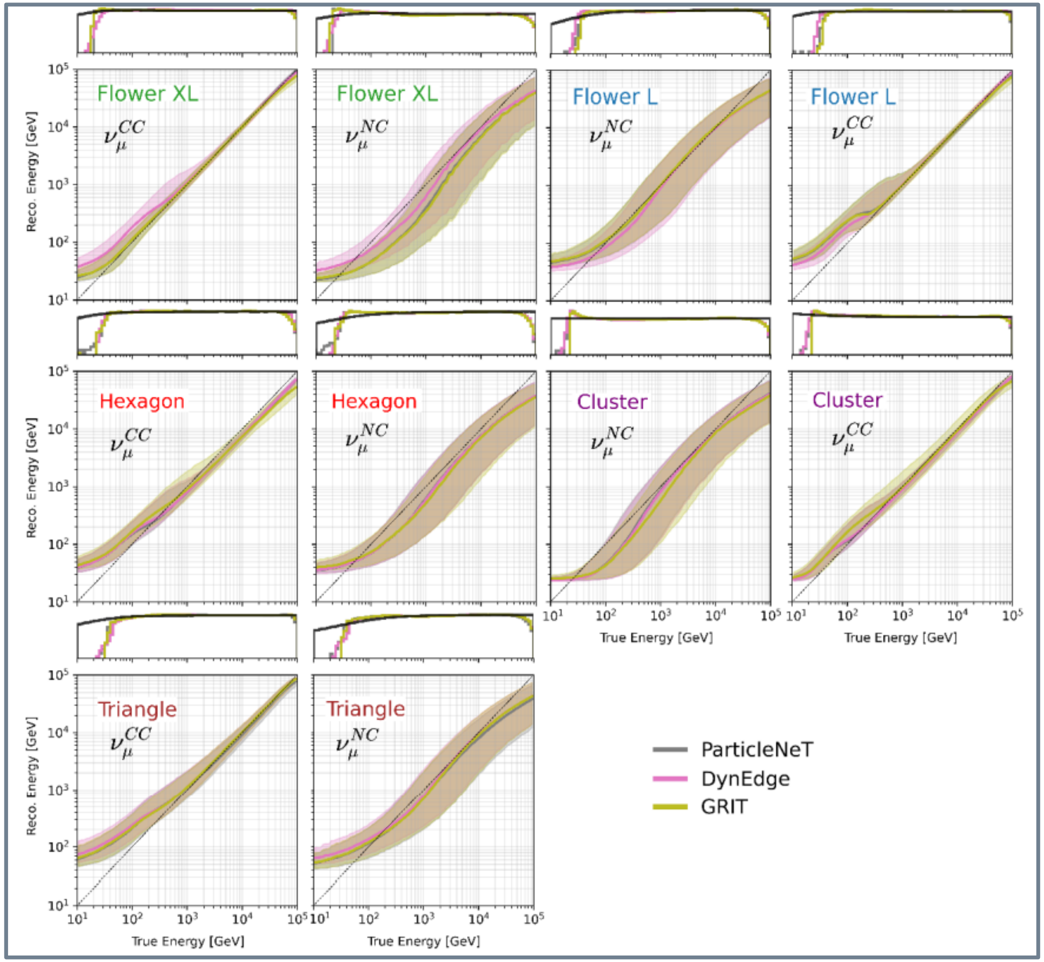
Energy reconstruction

Why is it useful to reconstruct the energy?
1) Characterization of cosmic rays and neutrino fluxes.
2) Measurement of neutrino oscillations.

Benchmarking metric: *energy resolution (true vs reco)*

Findings:
1) Better resolution for CC-interactions
2) More spreading of NC-interactions

Difficulties: non-visible energy from non-charged particles in NC-interactions, lack of resolution at low energies, saturation at high energy.



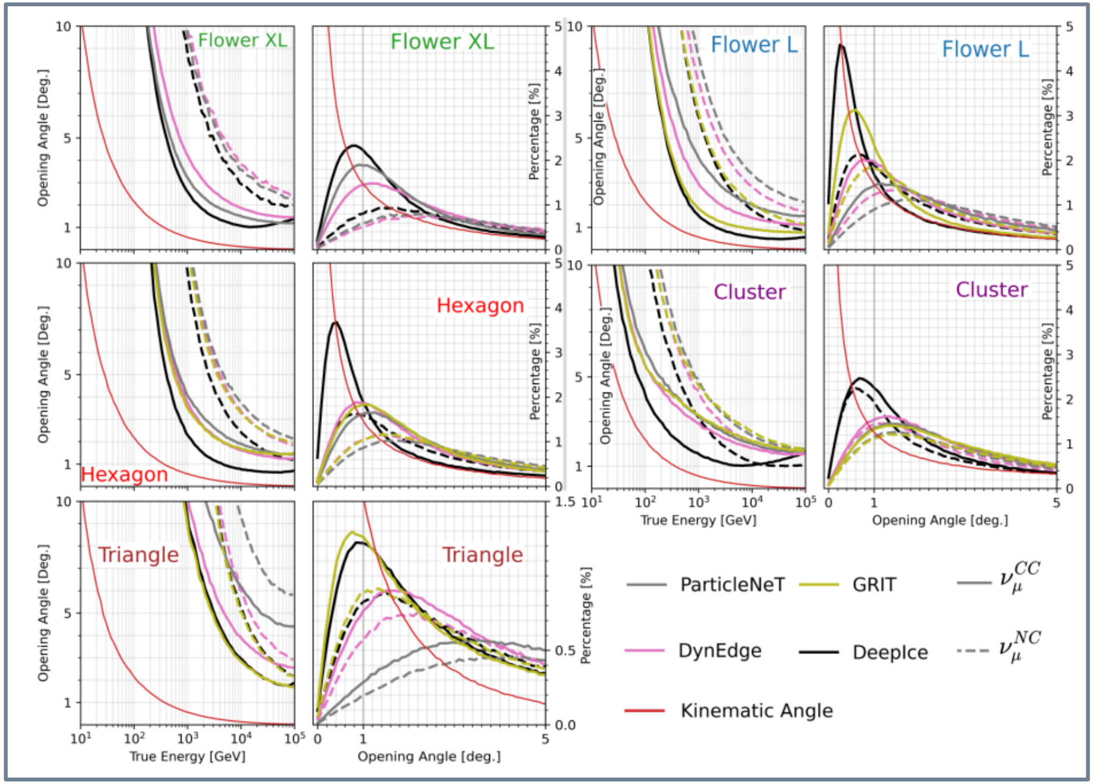
Direction reconstruction

Why is it useful to reconstruct the direction?
1) Astrophysical searches.
2) Measurement of neutrino oscillations.

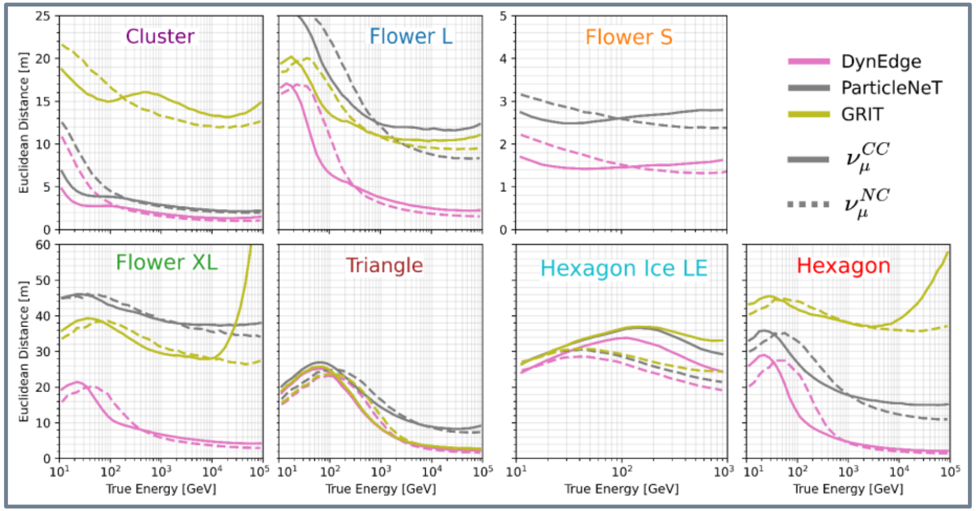
Benchmarking metrics: *opening angle*

Findings:
1) Better resolution for CC-interactions
2) Improved resolution for NC-interactions from CC-interactions
3) Good reconstruction in less sparse telescopes

Difficulties: reconstruction of cascades directions



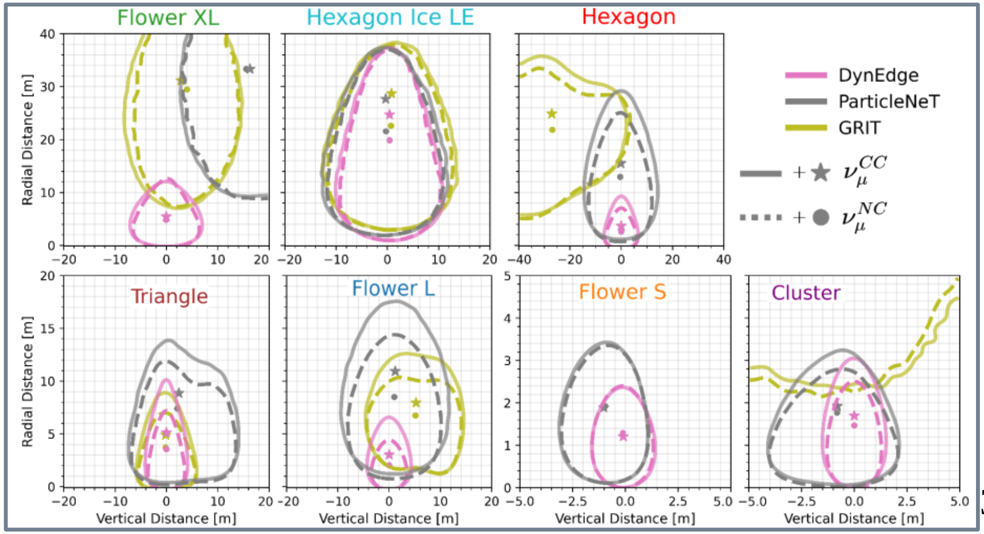
Vertex reconstruction



Why is it useful to reconstruct the interaction vertex?

1) Event containment.
2) Improvement of neutrino selection.

Benchmarking metrics: *euclidean distance*



Findings:

1) Better precision is small and less sparse telescopes
2) Similar performance for CC and NC interactions

Difficulties: low energies

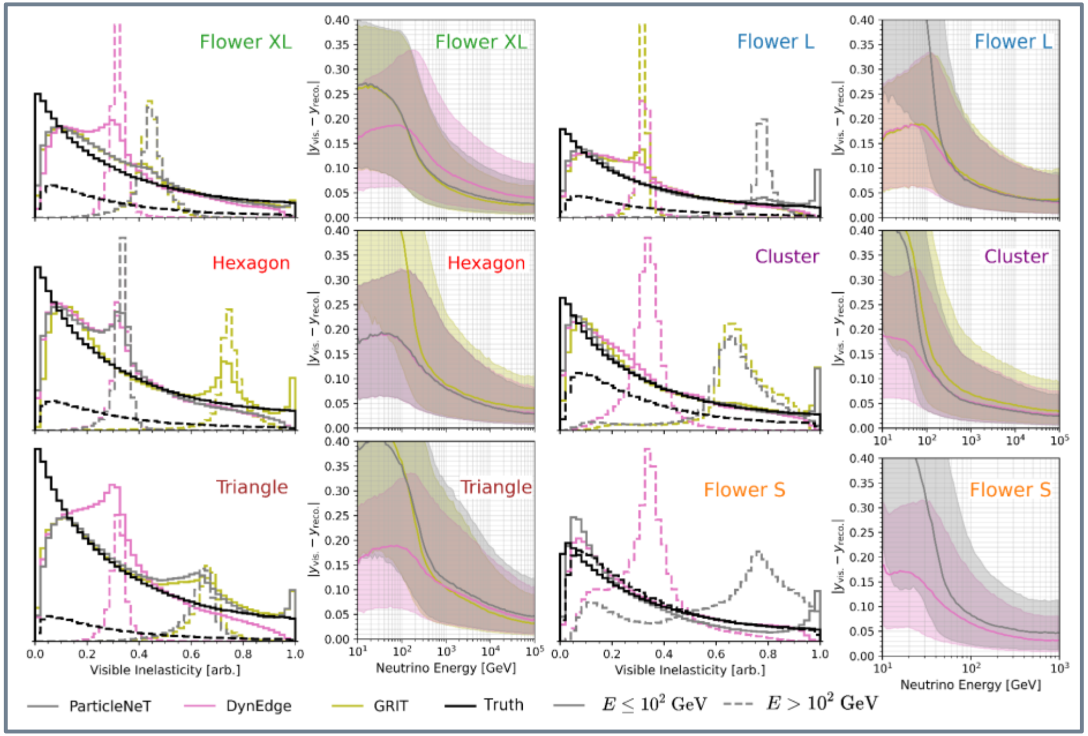
Inelasticity reconstruction

Why is it useful to reconstruct the inelasticy?
To differentiate neutrinos from anti-neutrinos

Benchmarking metrics: *median absolute error*

Findings:
1) Different behaviour below and above 100 GeV
2) Models minimize globally across all energies

Difficulties: extremely difficult below 100 GeV, spatial separation from the leptonic and hadronic components



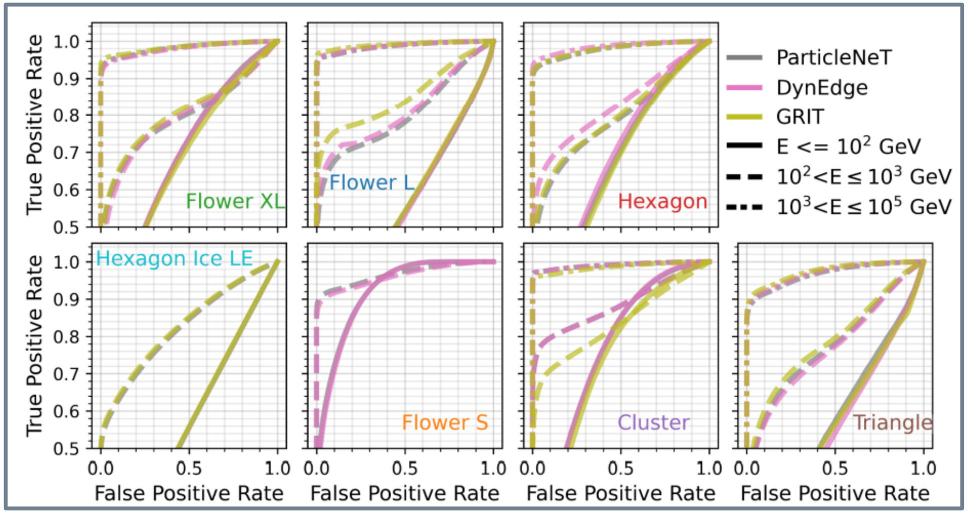
Track cascade classification

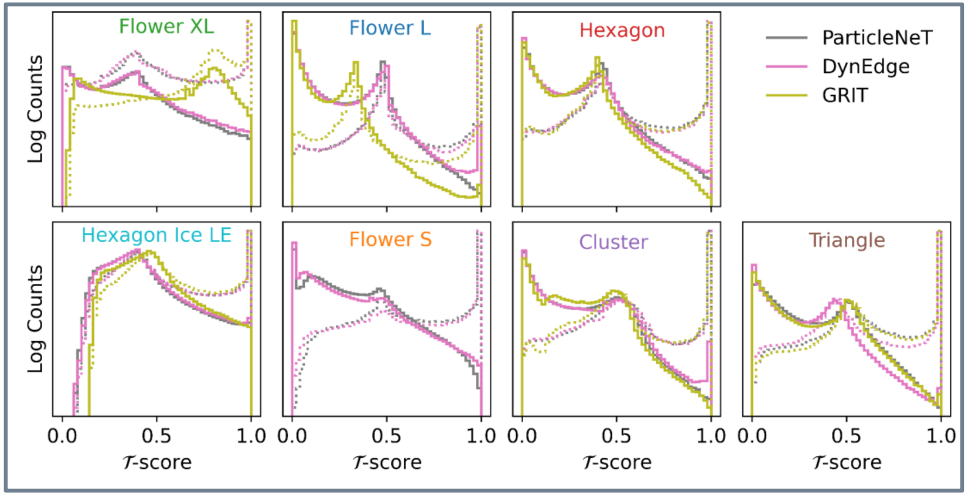
Why is it useful to classify events?
Tracks: good directionality ⟶ astrophysical searches
Cascades: better energy reconstruction and containment

Benchmarking metrics: *ROC curve*

Findings:
1) Improved separation with increasing energies
2) Better separation at low energies for dense telescopes

Difficulties: high dependence on energy range (event grouping)





Conclusions

NuBench questions and answers

- **Is there a gold model?** Not at all. All models are good and competitive
- **Is there a model that performs better for a given task?** Not really. Only DeepIce excels in direction reconstruction but it has a high computational cost in training
- **GNN or transformers?** Both show competitive results. Transformers suffer from the sequence length size and the dot-product attention focuses more on local relationships, as opposite to GNNs

NuBench message

- Great benchmarking strategy across 6 different geometries with 4 different models
- Collaboration team from different experiments
- Not intended to be a realistic neutrino interaction simulation, but to compare *apples to apple*

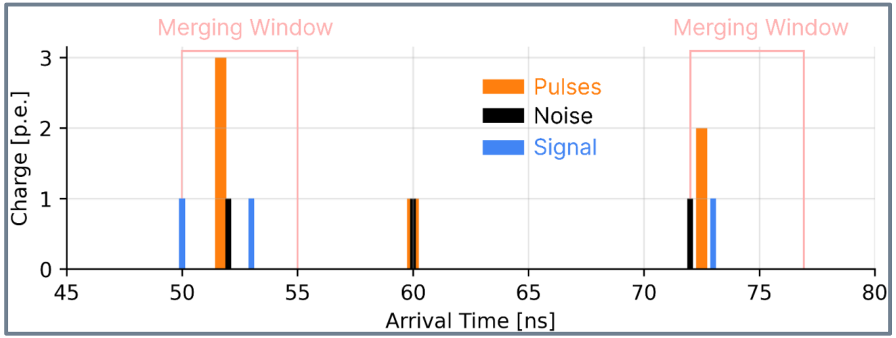
Have a look for yourself! **https://iopscience.iop.org/article/10.1088/1748-0221/21/05/T05001**

Thank you for your attention!
Many thanks to NuBench team and collaborators

Noise injection

Detector response

- Individual photon arrival times are linearly shifted into a trigger window of at least 5 µs around the mean arrival time
- Noise is artificially injected around the trigger window
- Pulse: photon-noise merging based on TTS



Considerations:

1) No directional information from multi-PMTs OMs
2) Effective OMs with increased photo-detection rate
3) Ice/Water propagation: different OMs angular acceptance

The expected total number of noise pulses in the event is calculated via

$$N_{\text{total}} = \left(\frac{R_{\text{OM}} \cdot t_{\text{window}}}{10^6} \right) \cdot N_{\text{OMs}}$$

where

- $R_{OM} = N_{PMT} \times R_{noise}$ → Noise rate per OM
- $t_{window} = 5\mu s$

N_{total} OMs positions are randomly sampled for injecting the noise pulses.

The noise arrival times are sampled from a uniform distribution from 0ns to t_{max}

$$t_{\max} = \begin{cases} t_{\text{window}} & \text{if } \max(t) < t_{\text{window}} \\ \max(t) & \text{otherwise} \end{cases}$$