GraphNeT 2.0

A Deep Learning Library for Neutrino Telescopes

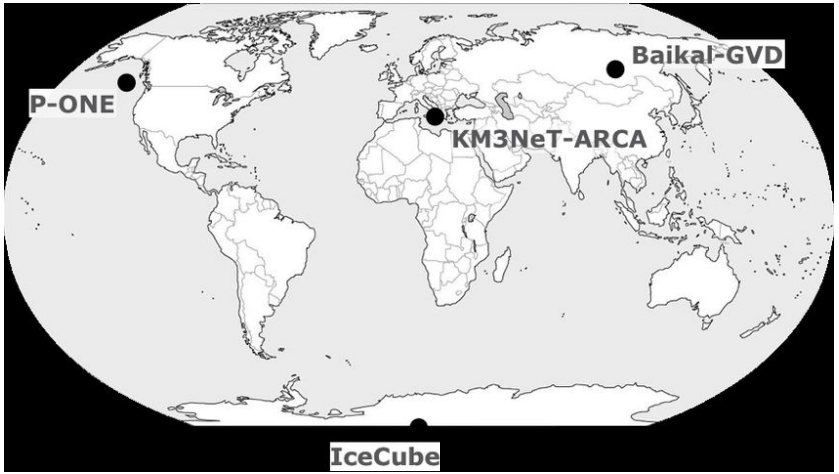
HAMLET / Copenhagen, August 2024

Rasmus Ørsøe

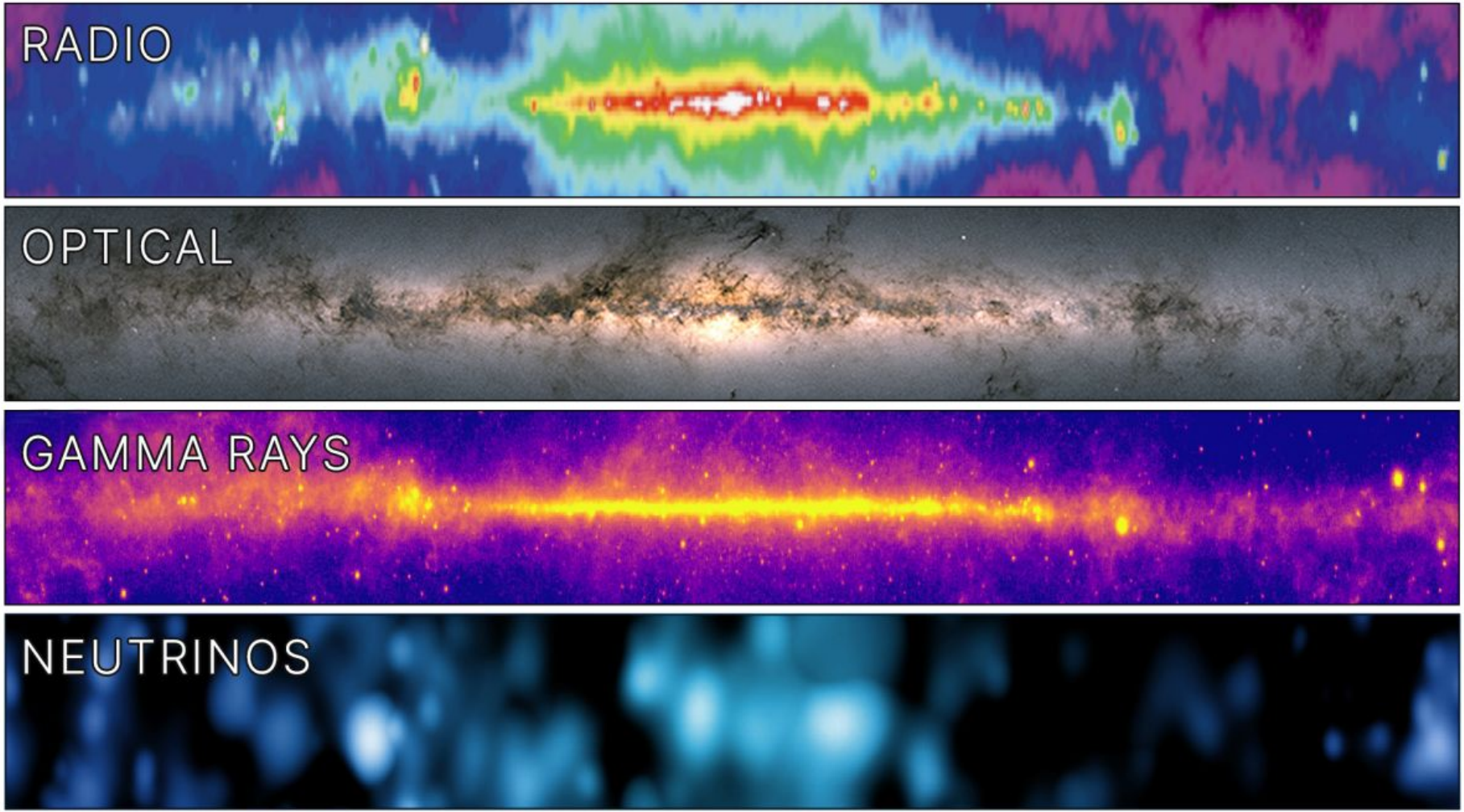
Technical University of Munich

Neutrino Telescopes

Neutrino telescopes are used to both characterize the neutrino itself and to provide a complementary view of astrophysical objects.



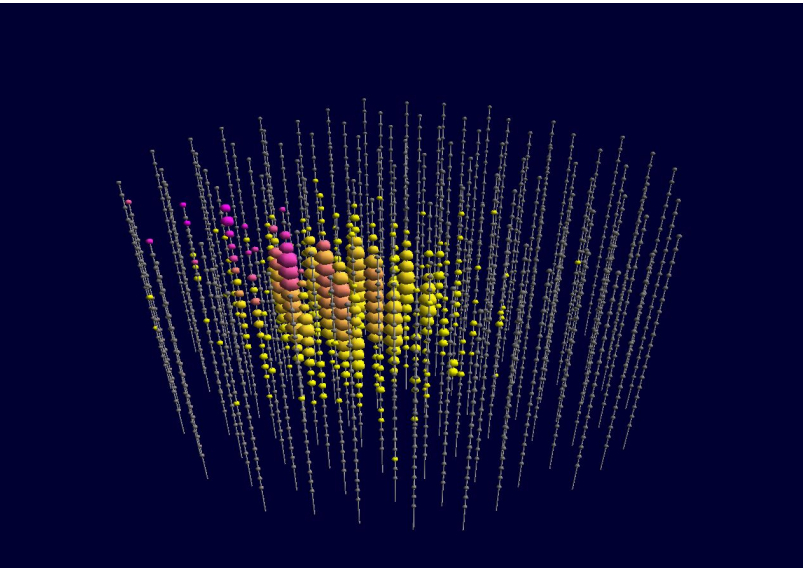
Partly or fully funded neutrino telescopes. IceCube is completed and the rest is under construction. **https://pos.sissa.it/358/028**



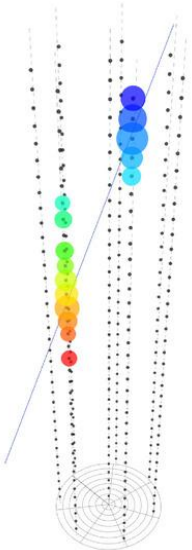
View of the Milky Way as seen from conventional telescopes (radio, optical, gamma rays) and neutrinos from IceCube Neutrino Observatory. Credit: IceCube Collaboration

Neutrino Telescopes

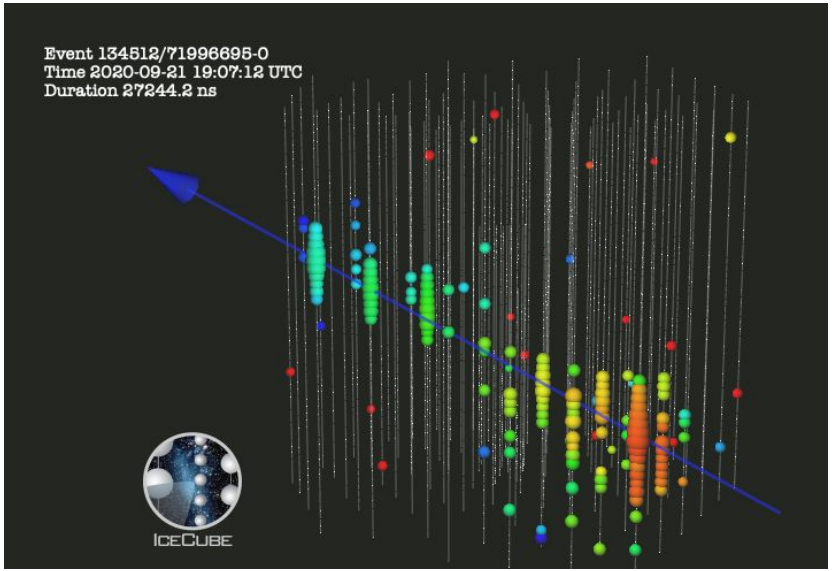
KM3NeT-ARCA



Baikal-GVD



IceCube



Left) Simulated neutrino event in the KM3NeT-ARCA detector. Credit: KM3NeT Collaboration **Middle)** Neutrino event in the Baikal-GVD. Credit: Baikal-GVD Collaboration. **Right)** Illustration of a neutrino interaction in the IceCube detector underneath the antarctic ice. Credit: IceCube Collaboration

Neutrino Telescopes

Re-use models from one telescope to another

Facilitate cooperation between deep learning experts and physics domain experts

Phrased in a way that is accessible to the general deep learning community, so they may help us out!

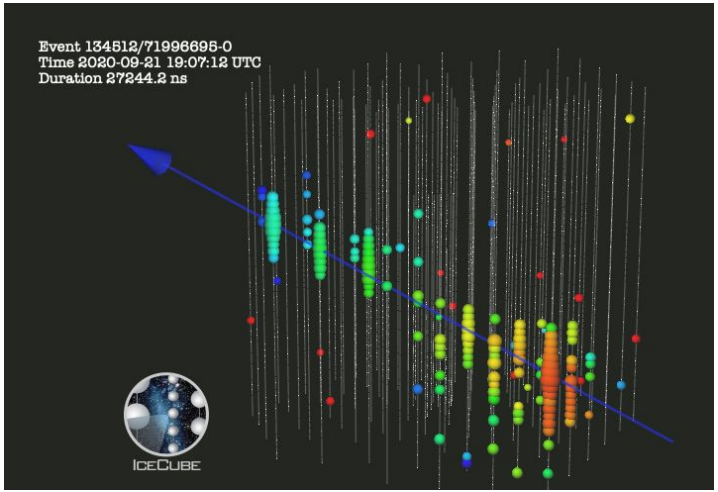


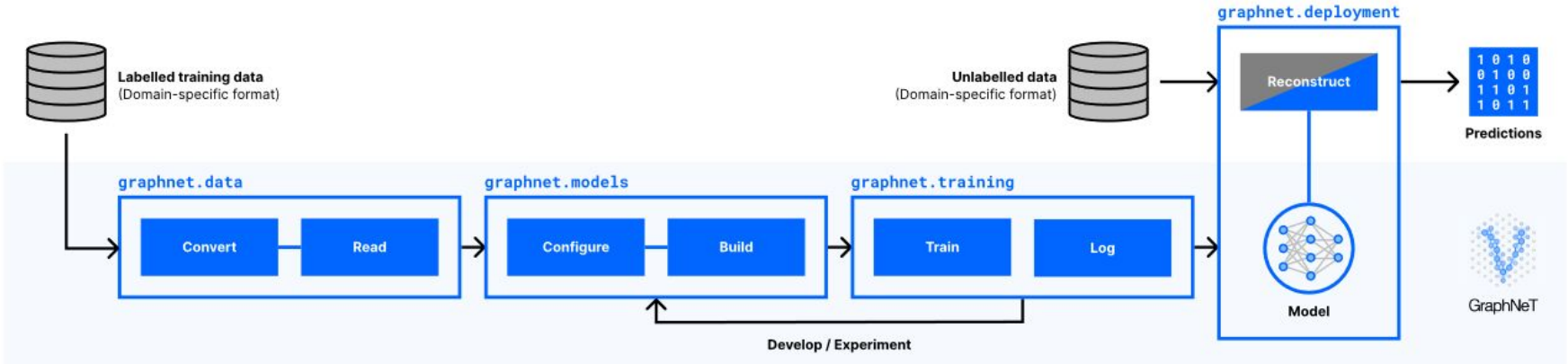
Illustration of a neutrino interaction in the IceCube detector underneath the antarctic ice. Credit: IceCube Collaboration

GraphNeT

A Deep Learning Library for Neutrino Telescopes

graphnet-team/graphnet

- **A framework for developing DL-based tools for neutrino telescopes**
- **One stop shop: from model development to deployment**
  - **Developers**
    - Everything needed to build, train and validate models from scratch
  - **Physics Domain Experts**
    - Can choose from a library of pre-trained models and apply them



graphnet.data

code for reading, writing and processing data

Different data representations can be built in graphnet, making it compatible with the established deep learning paradigms like

CNNs, **GNNs**, **RNNs**, **Transformers**, etc.

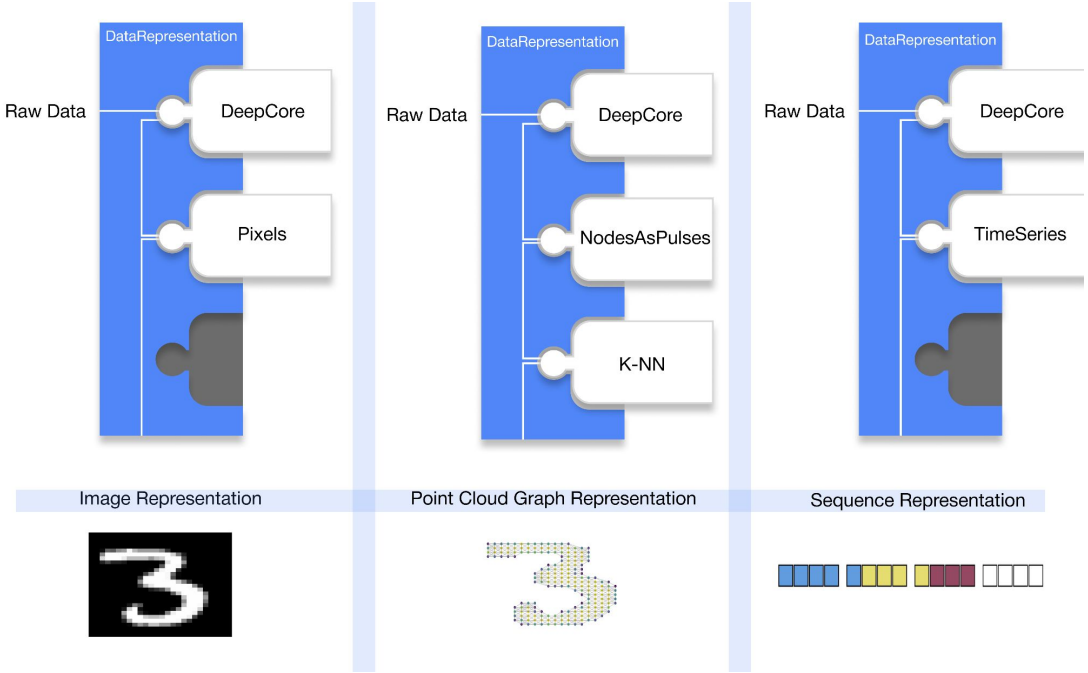


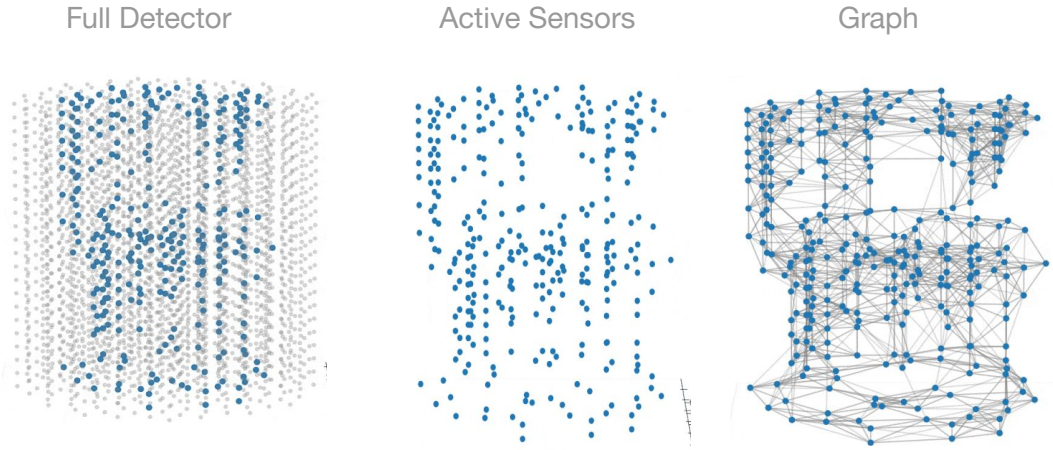
Illustration of how fundamentally different data representations can be generated in GraphNeT 2.0, making the framework compatible with established deep learning paradigms.

graphnet.data

code for reading, writing and processing data

Different data representations can be built in graphnet, making it compatible with the established deep learning paradigms like

CNNs, **GNNs**, **RNNs**, **Transformers**, etc.



Stages of transforming a simulated neutrino event into a k-nn graph representation.

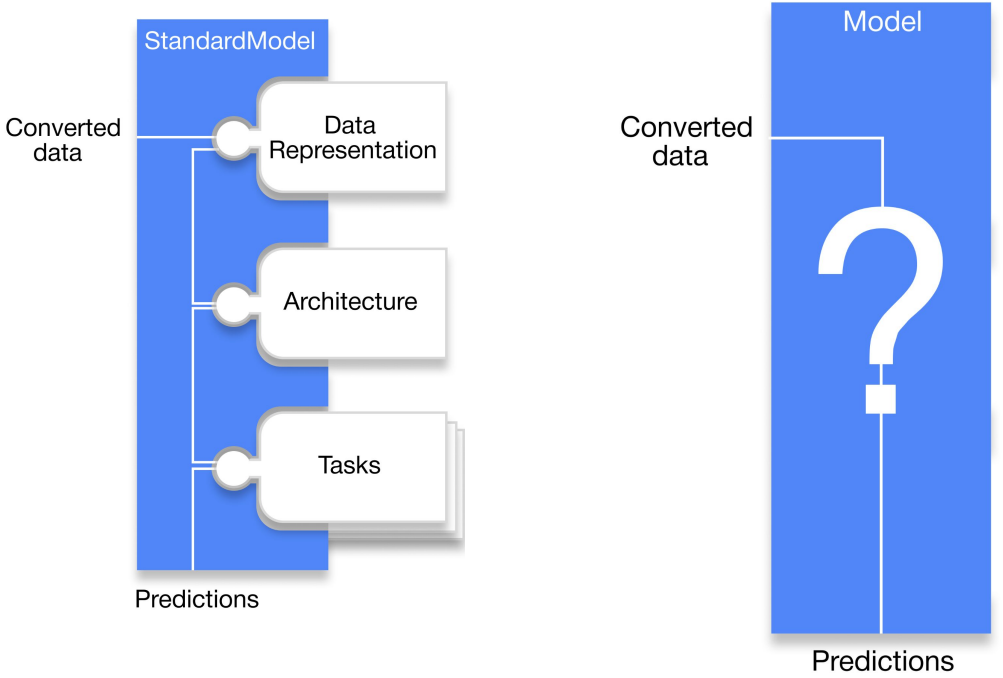
graphnet.models

code for building, configuring, training and saving models

StandardModel is a modularized model class where data representation, model architecture and physics task are interchangeable modules, allowing methods to be repurposed easily.

This class supports the vast majority of supervised learning tasks.

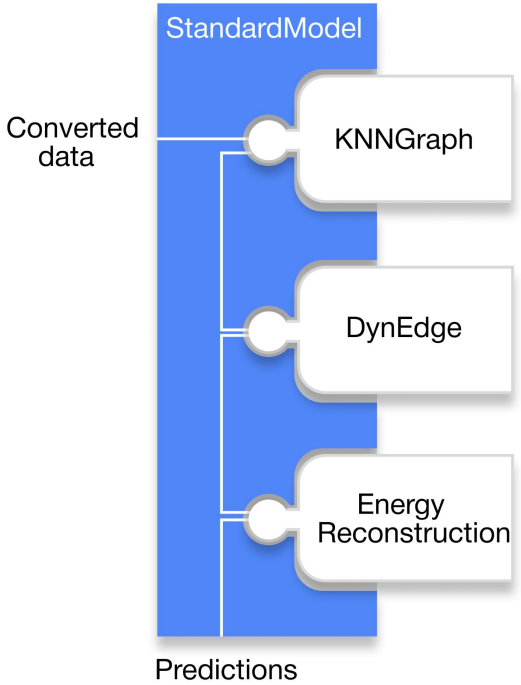
Model is a freeform model class with no rails, attended for niche methods and advanced users.



Left) StandardModel, a modularized DL model where components are interchangeable. **Right)** Freeform Model class with no rails.

graphnet.models

code for building, configuring, training and saving models



Conceptual illustration of the code to the right

```python
from graphnet.models.detector.icecube import IceCubeDeepCore
from graphnet.models.task.reconstruction import EnergyReconstruction
from graphnet.training.loss_functions import LogCoshLoss
from graphnet.models.graphs import KNNGraph
from graphnet.models.gnn import DynEdge

from graphnet.models import StandardModel

# Instantiate Modules
graph_definition = KNNGraph(detector = IceCubeDeepCore())
backbone = DynEdge(nb_inputs = graph_definition.nb_outputs)
task = EnergyReconstruction(loss_function = LogCoshLoss(),
                            hidden_size = backbone.nb_outputs)

# Build GraphNeT Model
model = StandardModel(
        graph_definition=graph_definition,
        backbone=backbone,
        tasks=[task]
        )
```

Example of defining a StandardModel in GraphNeT

graphnet.models

code for building, configuring and saving models

The StandardModel provides a very simple training syntax, significantly lowering the technical threshold for training complex models, without compromising on functionality.

```
# Train Model w. EarlyStopping On GPU 0
model.fit(train_dataloader = train_dataloader,
          val_dataloader = val_dataloader,
          max_epochs = max_epochs,
          gpus = [0])

 # Predict on Test Set Using GPUs
 results = model.predict_as_dataframe(dataloader = test_dataloader,
                                      gpus = [0])

 # Save model and predictions to disk
 model.save_state_dict('state_dict.pth')
 model.save_config('model_config.yml')

 results.to_csv('results.csv)
```

Example of training, predicting and saving

graphnet.models

code for building, configuring and saving models

Models in GraphNeT can be fully summarized using config files and their weights.

```python
from graphnet.models import StandardModel

# Re-Create Model From Config File
model = StandardModel.from_config('model_config.yml')

# Load Weights from Training Session
model.load_state_dict('state_dict.pth')


# Re-use model!
```

Example of re-instantiating a pre-trained model from configuration files.

GraphNeT is a Community

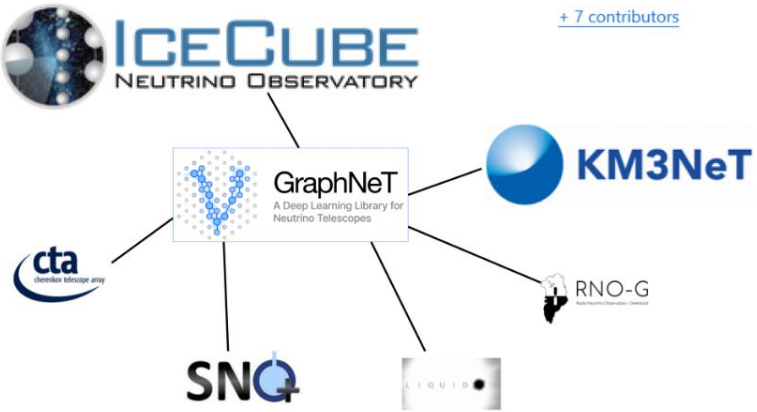
graphnet Public

Fork 84 Starred 85

GraphNeT is built by a community of deep learning enthusiasts working at the intersection of ML and neutrino physics.

Contributors 21

+ 7 contributors



2023, Bornholm, Denmark

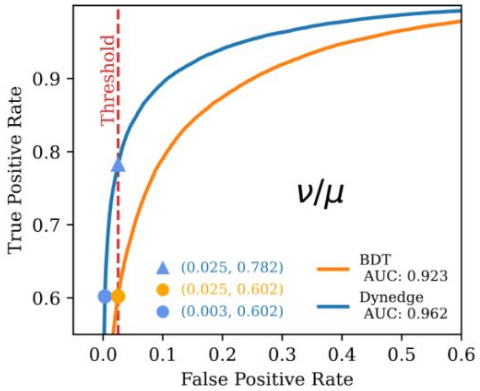
2024, Munich, Germany

Two most recent annual GraphNeT Workshops. The workshops aim to bring the community together and focus on solving common problems using GraphNeT.

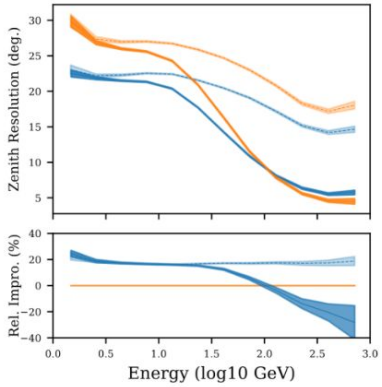
Examples of Usage

GraphNeT can be used to build and apply deep learning techniques at every step of a physics analysis

Classify events

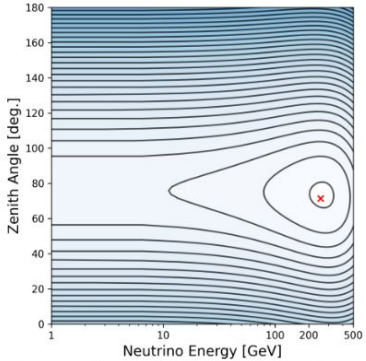


Reconstruct labels



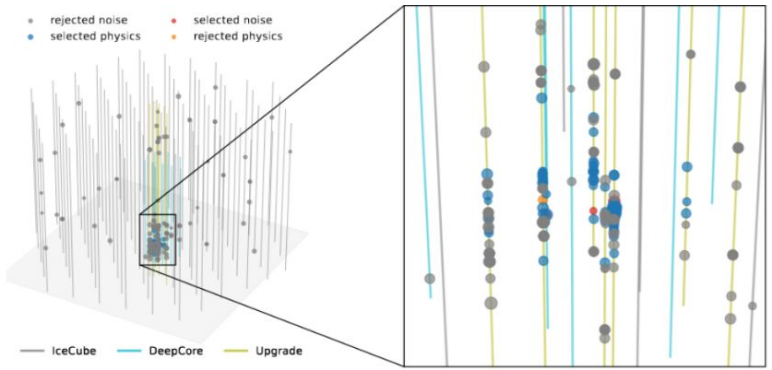
Resolution and ROC curves (blue) of models trained to reconstruct the zenith angle and distinguish between muon and neutrino events . Compared against SOTA (orange). **arXiv:2209.03042**

Learn pdfs



Example of using normalizing flows

Remove noise



Simulated IceCube Upgrade event cleaned by a model from GraphNeT. **arXiv:2307.15295.**

kaggle

IceCube - Neutrinos in Deep Ice

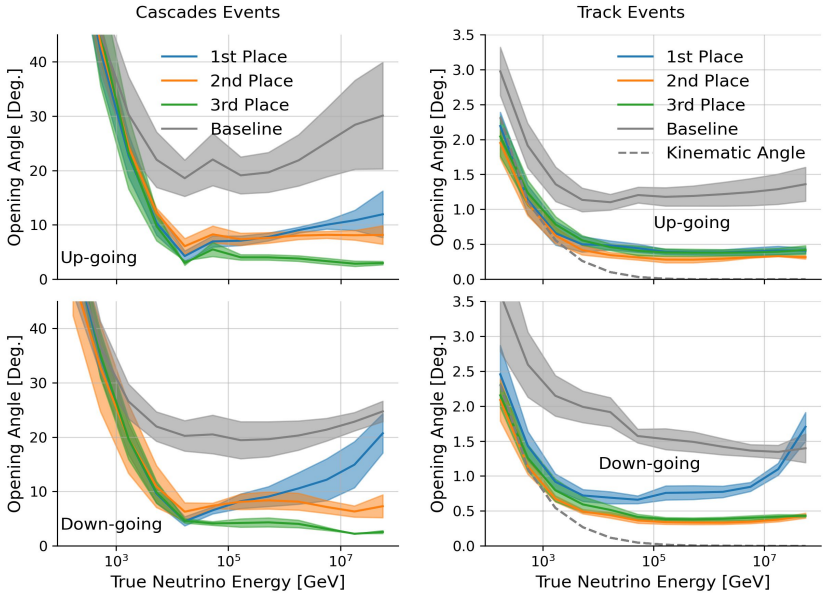
Reconstruct the direction of neutrinos from the Universe to the South Pole

On behalf of the IceCube Collaboration, Philipp Eller organized a Kaggle competition, where participants were tasked with producing algorithms that could accurately predict neutrino directions.

- 900 participants
- 11.200 submissions
- $50.000 prize pool

We provided them with a baseline to compare against using GraphNeT, along with technical material to introduce them to the library. Many participants used GraphNeT.

Several participants made contributions to GraphNeT.



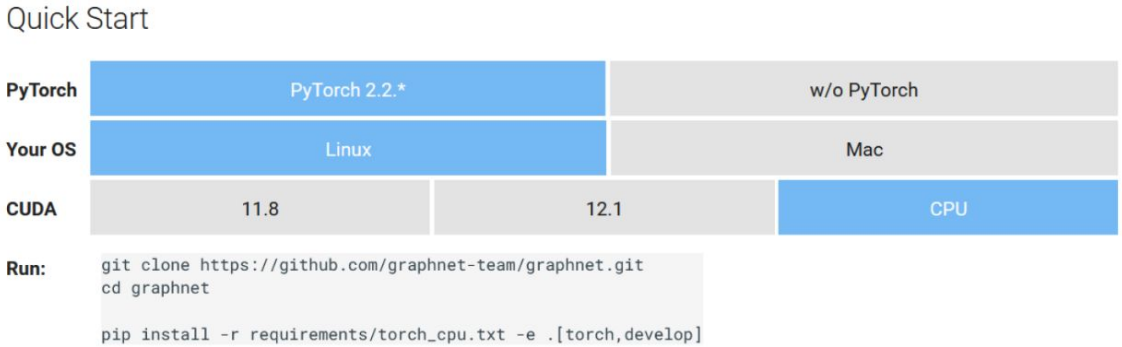
Comparison plots between the 1st, 2nd and 3rd place solutions against the baseline provided from GraphNeT. 1st and 2nd place solution is today implemented in GraphNeT, available for use by the general community. **https://doi.org/10.1140/epjc/s10052-024-12977-2**

Getting Involved is Easy

GraphNeT is rich with community resources to get you started.

We recommend checking out

- Colab Notebook
- Publications using GraphNeT



Snapshot of the installation matrix from the GraphNeT documentation

graphnet-team/graphnet

Join Slack

Rasmus Ørsøe
rasmus.orsoe@tum.de