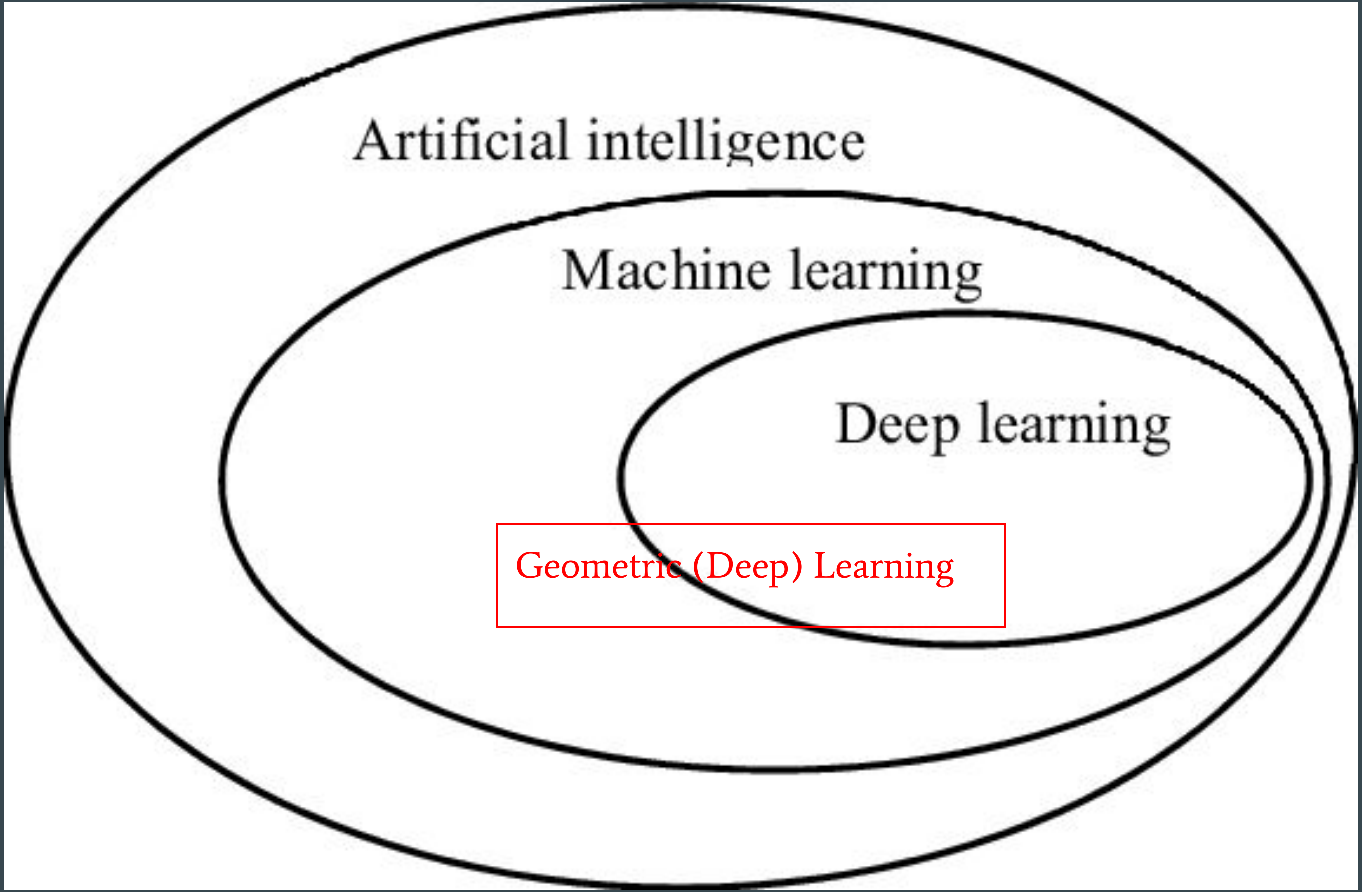
Graph Neural Networks

Using Graph Neural Networks to classify or reconstruct events in KM3NeT

Gijs Vermariën 24-09-2020 - KM3NeT group meeting

Machine Learning/ Artificial Intelligence/ Deep Learning



Flavors of Machine Learning

- Supervised Learning ${x}, {y}$
    - Support Vector Machines
    - Decision Trees
    - Linear Regression
    - Neural Networks
        - Vanilla Neural Networks
        - Convolutional Neural Networks
        - Graph (Convolutional) Neural Networks
- Unsupervised Learning ${x}$
    - (Deep) Reinforcement Learning
    - Clustering
    - Autoencoders
    - Generative Adversarial Networks

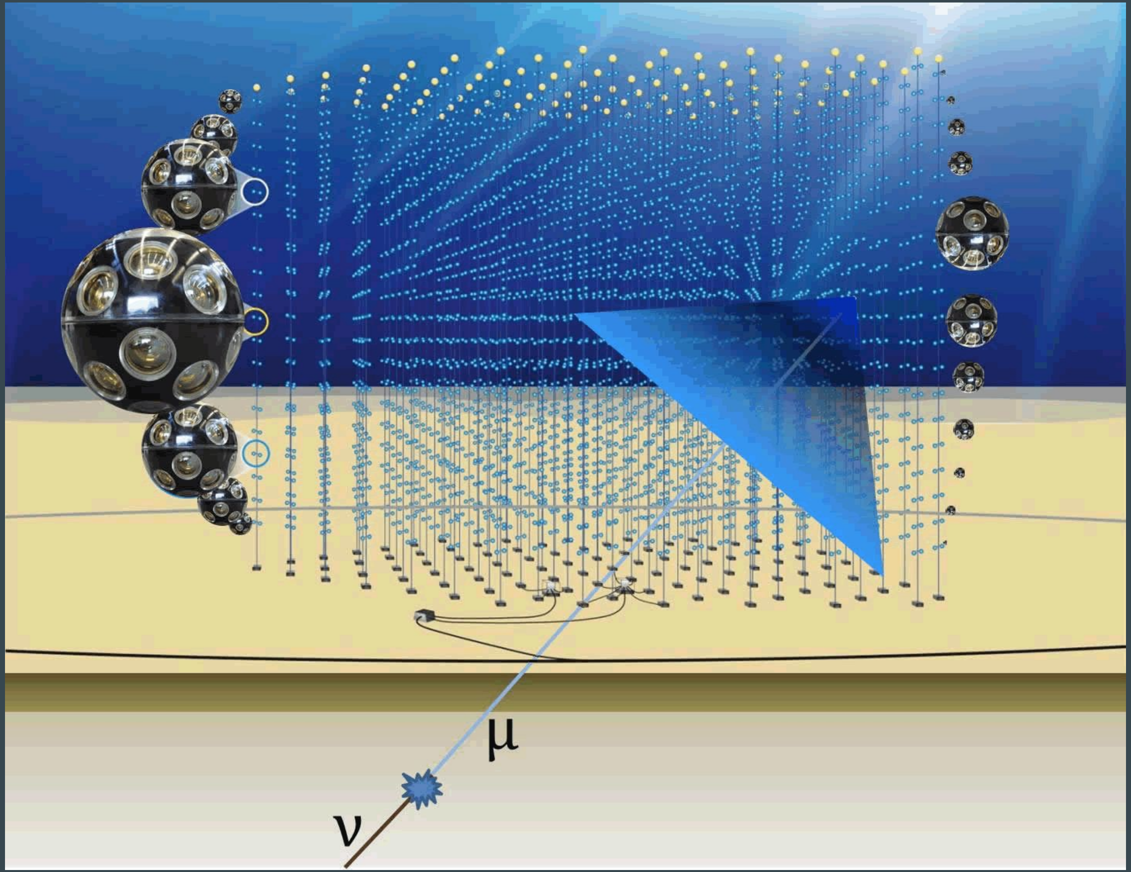
Vanilla to Convolutional to Graph

Neural	(2D) Convolutional	Graph Convolutional
$x \in \mathcal{R}^n, y \in \mathcal{R}^m$ $\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \to \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_m \end{bmatrix}$	$x \in \mathcal{R}^{m \times n}, y \in \mathcal{R}^l$ $\begin{bmatrix} x_{00} & \cdots & x_{0n} \\ \vdots & \ddots & \vdots \\ x_{m0} & \cdots & x_{mn} \end{bmatrix} \to \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_l \end{bmatrix}$	$\mathcal{G} = (\mathcal{V}, \mathcal{E}),\ \mathcal{V} \in \mathcal{R}^d, \mathcal{E} \in \mathcal{R}^{d \times d}$ $\left\{ \left\{ \left(\begin{bmatrix} v_0 \\ \vdots \\ v_m \end{bmatrix} \right) \right\}, \left\{ \left(\begin{bmatrix} e_{00} & \cdots & e_{0n} \\ \vdots & \ddots & \vdots \\ e_{m0} & \cdots & e_{mn} \end{bmatrix} \right) \right\} \right\} \to \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_l \end{bmatrix}$

where $\to$ is many nonlinear transformations with (many) parameters

Benefits?

- Natural representation of data, no binning of data
- Sparsity
- More robust against curse of dimensionality
- Higher expressivity with lower number of parameters



Data Architecture*

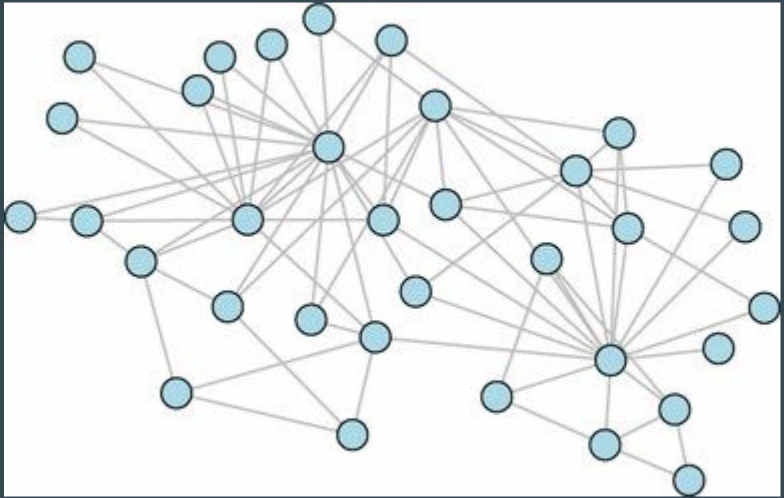
Simulated sensors (PMT) data:

One event ~1k rows:

- Spatial Features: `x ,y ,z ,dx, dy, dz`
- Features:
  - Time `t`
  - Charge `Q`
  - Rate `v`

Dense Graph

Gaussian kernel
$d_{ij}=\exp(|\mathbf{r}_i-\mathbf{r}_j|^2/\sigma^2)$
Introduce sparsity



* Prior work by me and Luuk Oudshoorn on Antares data

Current work within KM3NeT collaboration

- Michael Moser, Thomas Eberl: 4D CNN [1] development of OrcaSong and OrcaNet [2]
- Stefan Reck, Thomas Eberl: Implementation of EdgeConv [3] and contributions to Orca{Song,Net}
- Daniel Guderian: Research ORCA4 classification and reconstruction [3]
- Arumoy Shome, Brían Ó Fearraigh, Ronald Bruijn: Event classification in the KM3NeT pipeline [4]
- Other projects see [3]

Outside the collaboration:

- Graph Neural Networks for IceCube Signal Classification [5]

[1] M. Moser, "Sensitivity Studies on tau neutrino appearance with KM3NeT/ORCA using Deep Learning Techniques," p. 113, 2020

[2] https://git.km3net.de/ml/{OrcaNet,OrcaSong} and https://ml.pages.km3net.de/{OrcaNet,OrcaSong}

[3] https://indico.cern.ch/event/952545/

[4] https://github.com/arumoy-shome/km3net

[5] https://arxiv.org/abs/1809.06166

Outlook (work to be done)

- implement the PyTorch Geometric Framework into Orcasong [1]
  - PyTorch backend
  - State of the art Graph Neural Network implementations (paper to first implementation sometimes on order of days) [2]
  - CUDA support
- Compare PyG implementation to current work done by Daniel and Stefan

Possible science topics:

- Classifying Tau neutrino tracks in ORCA-6, that originate from neutrino flavor oscillations in the atmosphere from energetic cosmic beams (distinguish from NC and electron showers).
- Resolution and direction reconstruction of ARCA
- Use real background data on ORCA/ARCA and see how this compares to current MC simulations (Maarten de Jong came up with this idea) in the classifier/reconstruction
- High energy Tau neutrino identification

[1] https://arxiv.org/abs/1903.02428 and https://github.com/rusty1s/pytorch_geometric

[2] https://twitter.com/mmbronstein/status/1255096720763113472