Moritz Scham[1,2,3], Dirk Krücker[1]

🌲DeepTreeGAN🌲
Fast Generation of High Dimensional Point Clouds

Generative models for particle showers

Motivation: Computing Challenge

- Particle physics: simulate detector response and physical processes to test theories
  ⟶ Immense computational effort
- High-Luminosity phase ⟶More particles to simulate
- Future CMS high granularity calorimeter (HGCAL): more then 6M channels
  ⟶ Time-consuming simulations
  ⟶ Projected compute budget insufficient

⇒ **Save CPU time by using a Neural Network to simulate HGCAL data**

ML Challenges

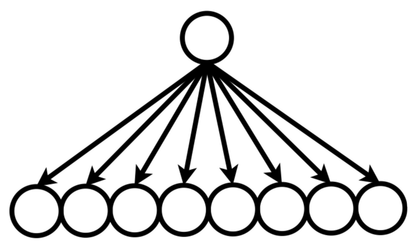
Future high granularity calorimeters:
- High number of Channels
- Irregular Geometry
- Sparse data

⟹No ML model powerful enough yet
⟹**Point Clouds** (PCs)

Target PC size: 2k Hits × 5 [E,t,x,y,z]

Core problem: How do we upsample PCs?

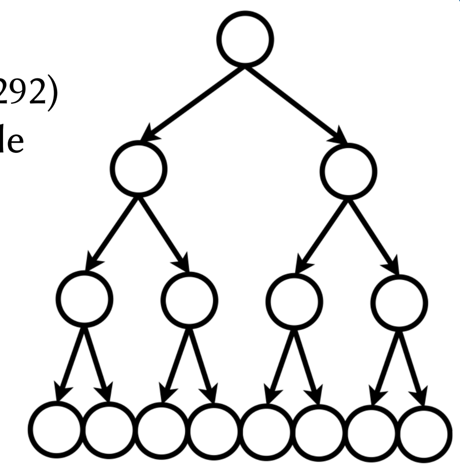
Naive approach:
Latent Vector ⟶ FFN ⟶ PCs



Number of parameters explodes
⟹ Not trainable

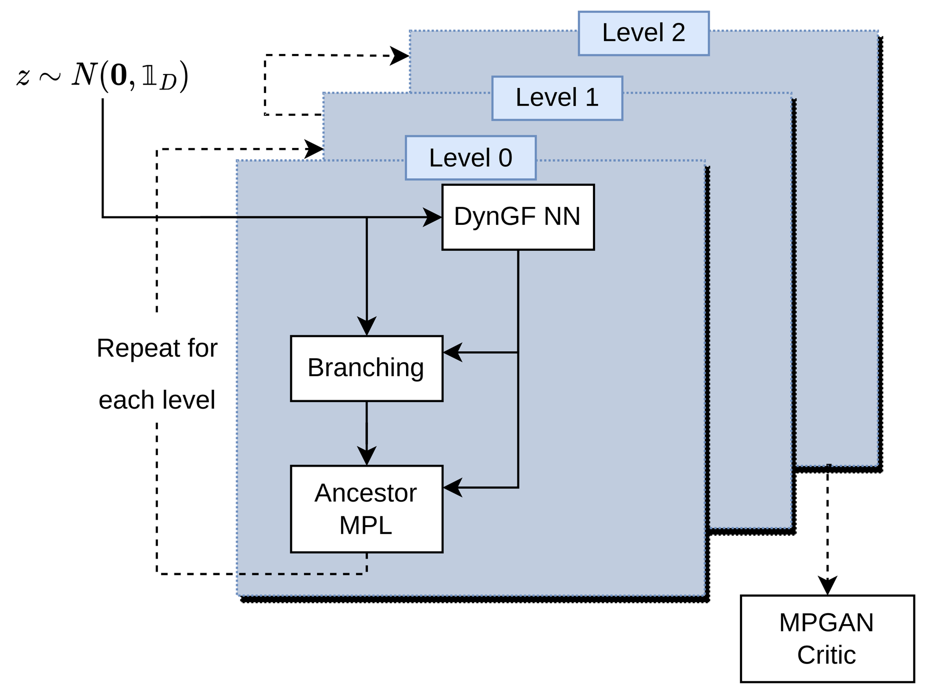
DeepTreeGAN:
- Inspired by TreeGAN (arXiv:1905.06292)
- FFN projects each particle to multiple
- Repeat to grow a tree
- After n projections: $\prod_i^k k_i$ particles

⟹ Small output space for each FFN
⟹ Small number of parameters
⟹ Sparse representation, no padding



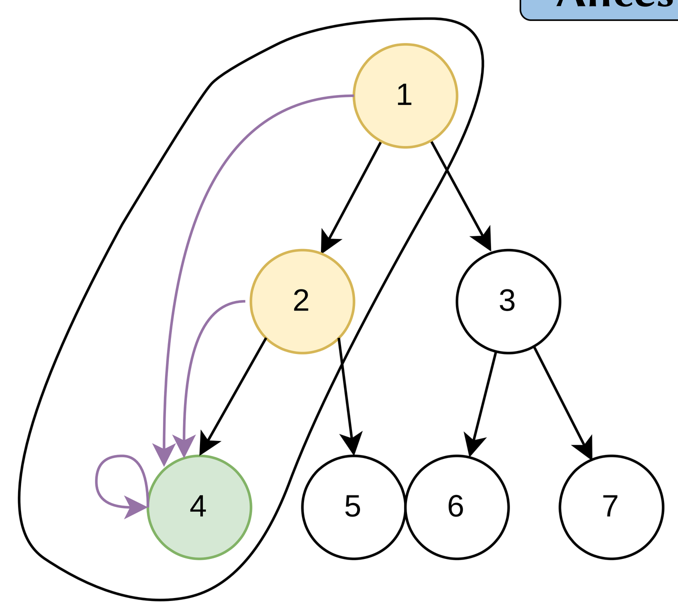
The Generator

Model Overview



- Start with random vector z
- Repeat for each level:
  1) **DynGF**: Encode the global state of the leaves (FFN ⟶ Sum ⟶ FFN)
  2) **Branching**: Split each of the leaves
  3) **Ancestor MPL**: Pass information down from ancestors to their children
- Last level: Output to discriminator (Currently: MPGAN critic from *JetNet*)

Ancestor Message Passing Layer



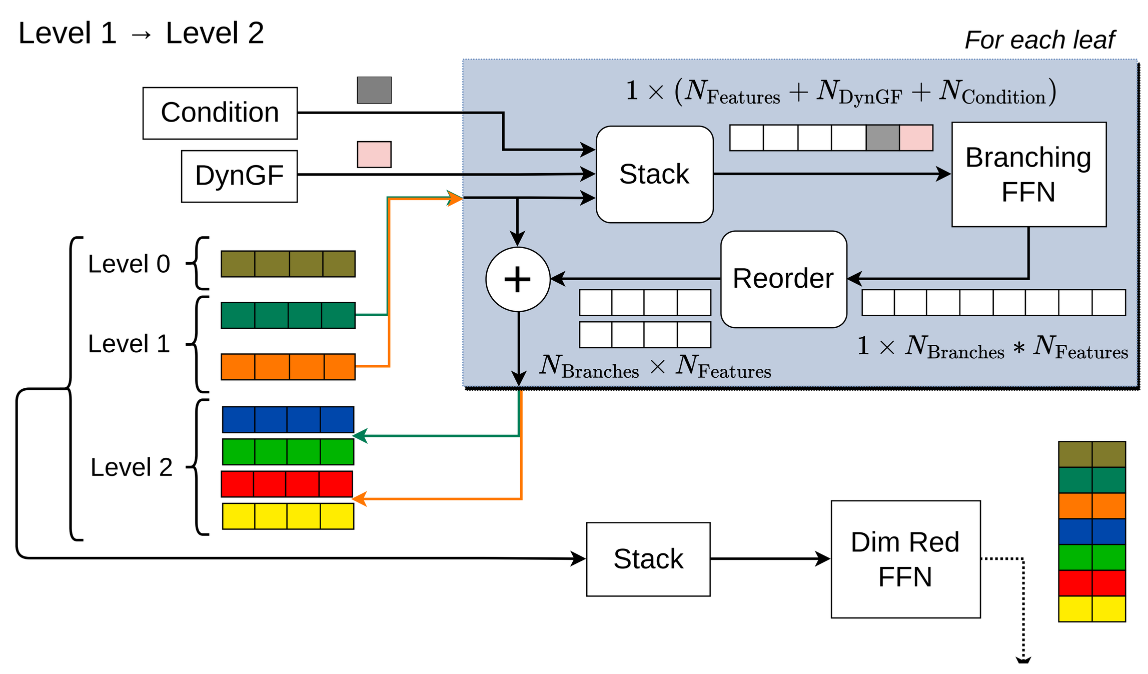
Selected MPL: GINConv (arXiv: 1810.00826)

1) Message: $\mathrm{Msg}_{j \to i} = x_j$

2) Aggregate: $\mathbf{Aggr}_i = \sum_{i \in N(j)} \mathrm{Msg}_{j \to i}$

3) Update: $x_i \leftarrow \mathrm{NN}\left((1+\epsilon)\, x_i + \mathbf{Aggr}_i\right) + x_i$

Branching

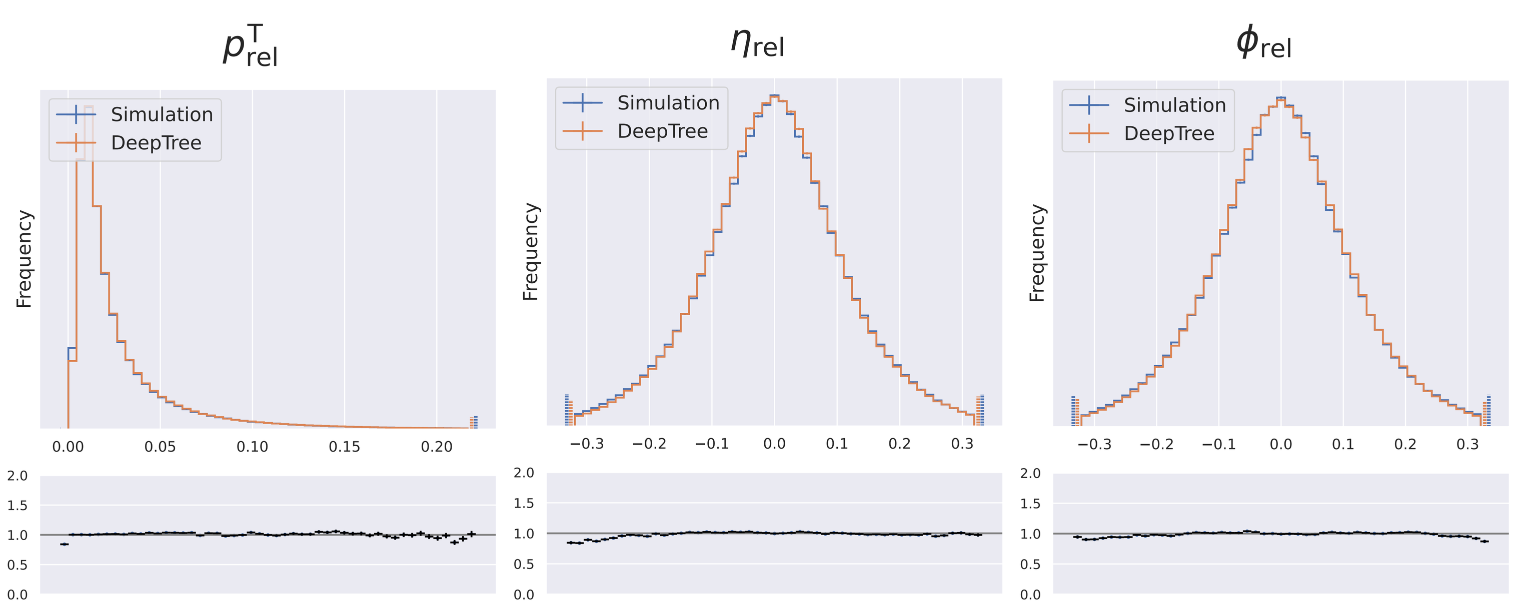


Branch from Level 1 to Level 2
- For each leaf separately:
  1) **DynGF**: Stack leaf with condition and global feature vector
  2) Branching FFN: Project to target dimension
  3) Split into the number of branches
  4) Add the parent ontop
- Stack up all levels of the tree
- Reduce the number of features with a FFN

The Benchmark: JetNet 30

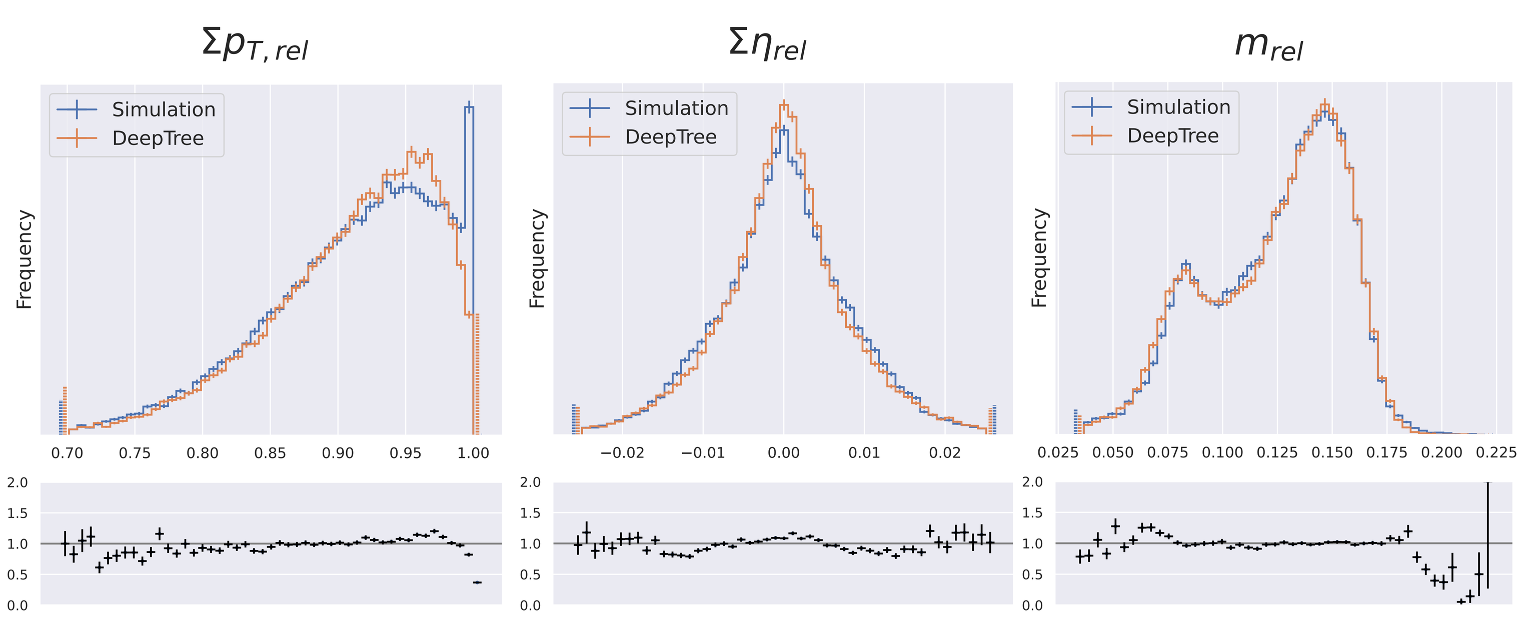
JetNet (arXiv:2106.11535) provides :
- Dataset (Pythia)
  - Hadronized jets, anti-kT (R=0.8)
  - 170k gluon/light quark/top jets
  - Leading 30 constituents by p^T
  - Metrics for benchmarking
  - Baseline model: MPGAN
  - Generator+Critic: Message Passing Networks on densely connected PCs
    ⟹ Good performance & bad scaling $\mathcal{O}(n^2)$

Dataset	Model	$W_1^M \times 10^3$	$W_1^P \times 10^3$	$W_1^{EFP} \times 10^5$	FPND
Gluon	In Sample	0.7±0.2	0.44±0.09	0.63±0.07	
	MPGAN	**0.7**±0.2	**0.9**±0.3	**0.7**±0.2	**0.12**
	DeepTree	2.5±0.3	1.7±0.3	2.6±0.5	0.35
Light Quarks	In Sample	0.5±0.1	0.5±0.1	0.46±0.04	
	MPGAN	**0.7**±0.2	4.9±0.5	0.7±0.4	0.35
	DeepTree	1.0±0.5	**1.7**±0.6	0.8±0.5	**0.15**
Top	In Sample	0.51±0.07	0.55±0.07	1.1±0.1	
	MPGAN	0.6±0.2	2.3±0.3	2±1	0.37
	DeepTree	0.7±0.2	**0.9**±0.4	2±1	**0.09**



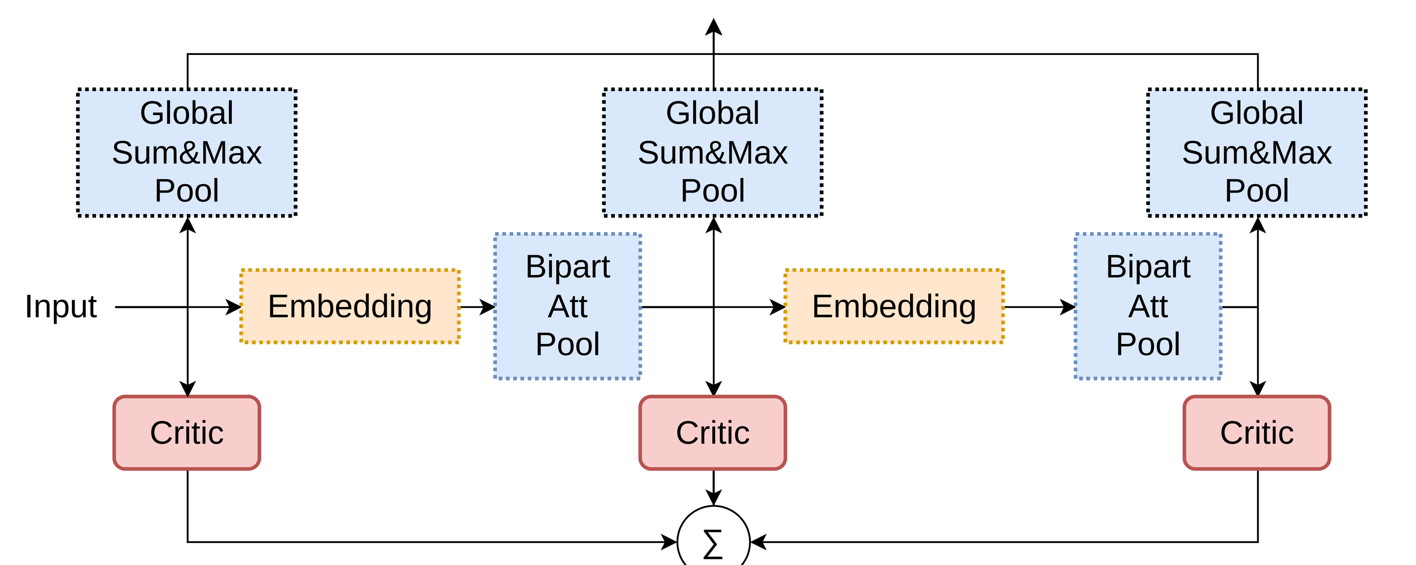
$$p^T_{\text{rel}} = \frac{p^T_{\text{particle}}}{p^T_{\text{jet}}}$$
$$\eta_{\text{rel}} = \eta_{\text{particle}} - \eta_{\text{jet}}$$
$$\phi_{\text{rel}} = \phi_{\text{particle}} - \phi_{\text{jet}}$$
$$m_{\text{rel}} = \frac{m_{\text{jet}}}{p^T_{\text{jet}}}$$

Top Jets with up to 30 constituents marginal↑ inclusive↓

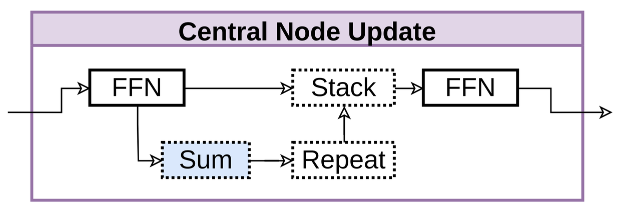


Preview: Discriminator on JetNet150
🌲VS🌲

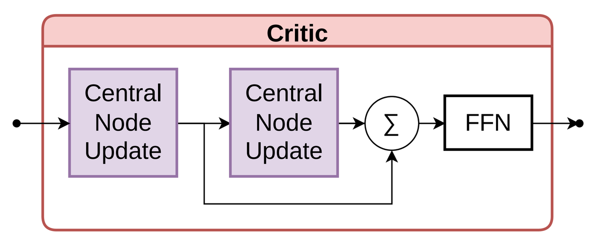
Baseline critic doesn't scale
⟹ Scalable critic



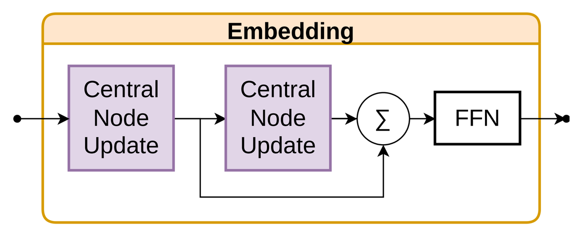
- 3 levels, 3 critics
- Embedding: Features↑
- Bipartite Attention Pool: Points↓
- Loss:
  - Feature Matching on input space of critics
  - Hinge Loss on sum of critics



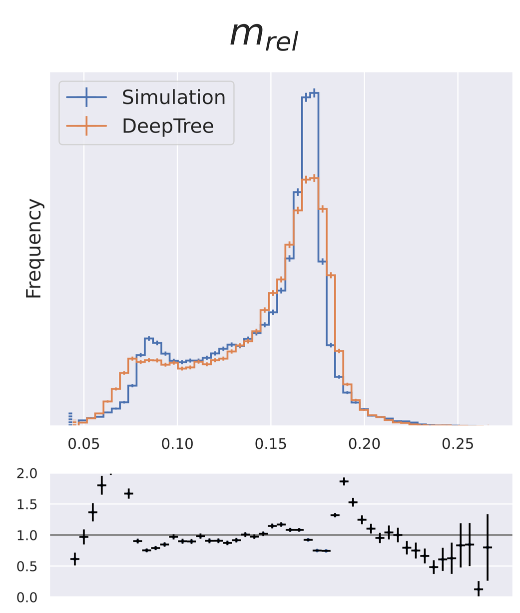
Updates the points with global information



Map "real" to 1, "fake" to -1



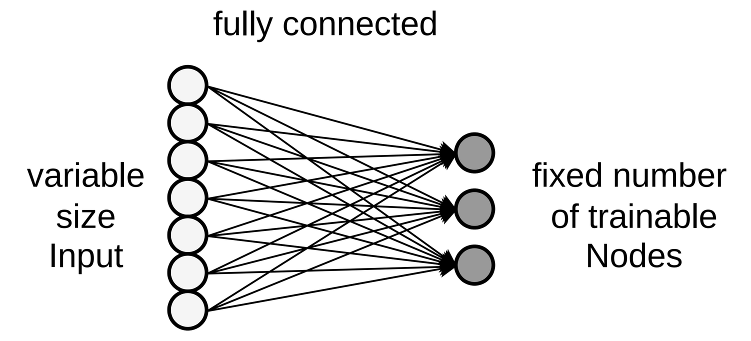
Project to higher dimension



First Results on Top Jets with up to 150 constituents

Bipartite Attention Pool
Goal: Pool variable number of points differentiable

Bipart Att Pool



GATv2Conv: $\mathbf{x}'_i = \alpha_{i,i}\mathbf{\Theta}\mathbf{x}_i + \sum_{j \in \mathcal{N}(i)} \alpha_{i,j}\mathbf{\Theta}\mathbf{x}_j$

⟹ **Excellent modeling in 30x3 dimensions**

⟹ **Already good modeling for 150 particles**

⟹ **New, differentiable up/downprojection method for point clouds**

⟹ **Promising model to scale to even larger point clouds (e.g. HGCAL)**

(1) DESY, Hamburg (2) FZJ and Jülich Supercomputer Center
(3) RWTH University of Aachen
Fund through Helmholtz AI - grant number: ZT-I-PF-5-3

HELMHOLTZ AI
ARTIFICIAL INTELLIGENCE COOPERATION UNIT
JÜLICH Forschungszentrum
RWTH AACHEN UNIVERSITY
DESY.