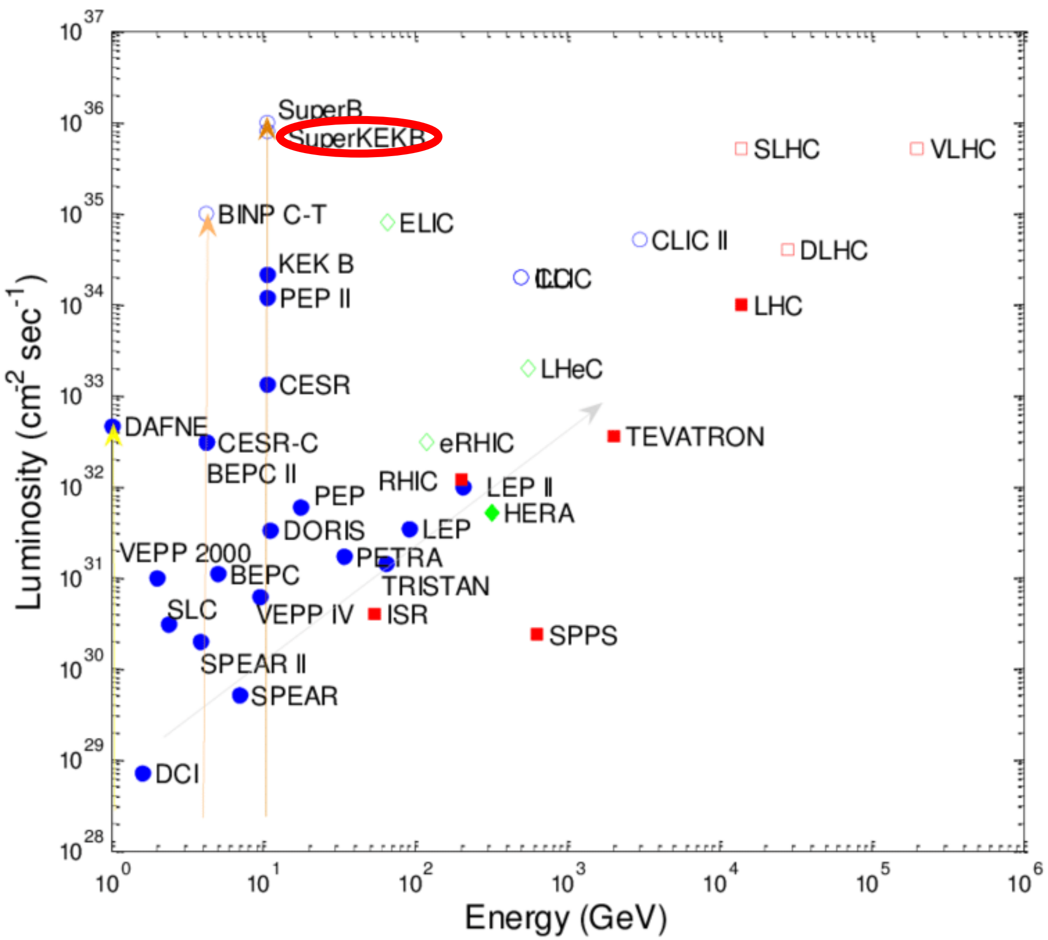
Smart Background Simulation with Graph Neural Networks

Boyang Yu [1], Nikolai Hartmann [1], Thomas Kuhr [1]
[1] *Ludwig-Maximilians-Universität München*

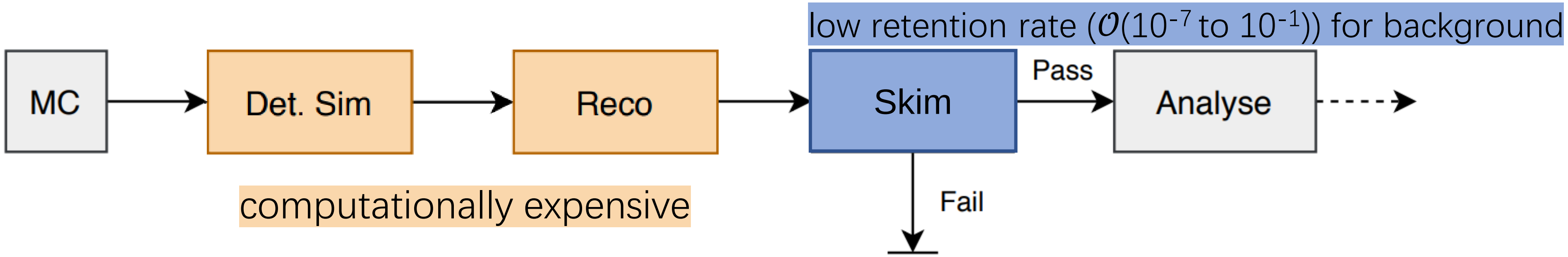
IDT-UM Collaboration Meeting, May 11th, 2021

Very High Luminosity means Tremendous Dataset

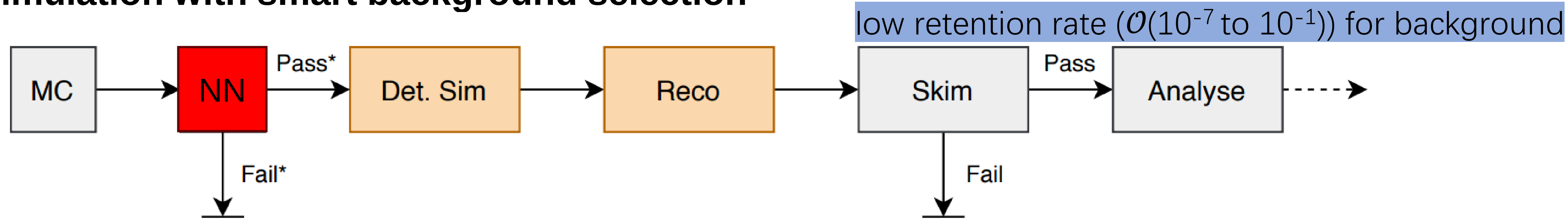


Biscari, Caterina. (0002). Accelerators R&D. Proceedings of Science.

Current Monte Carlo Simulation data flow



Simulation with smart background selection

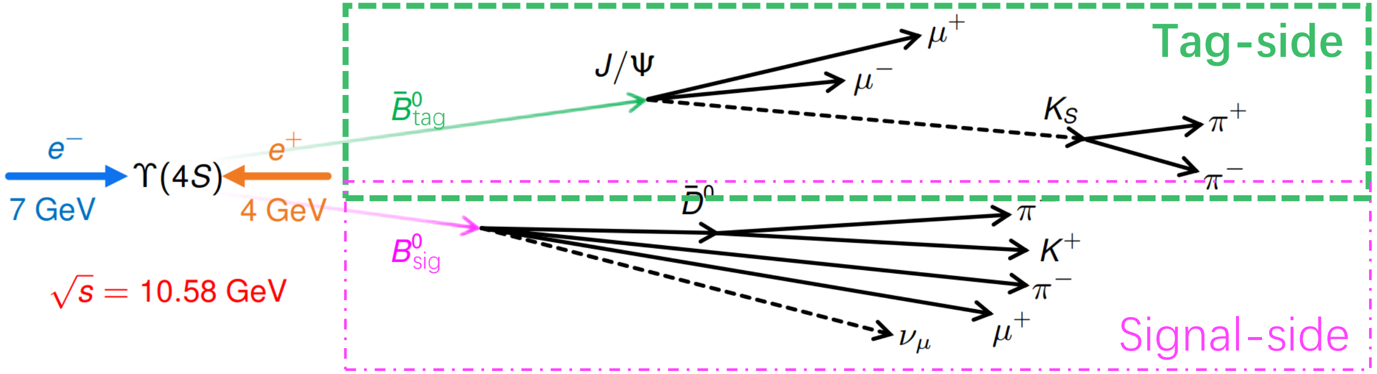


Previous Works:

- **Dissertation**: *Hadronic Tag Sensitivity Study of* $B \to K^{(*)}\nu\bar{\nu}$ *and Selective Background Monte Carlo Simulation at Belle II*, James Kahn, 2019
- **Talk**: *Selective background Monte Carlo simulation at Belle II*, James Kahn, CHEP 2019

Motivations

Tagging method:

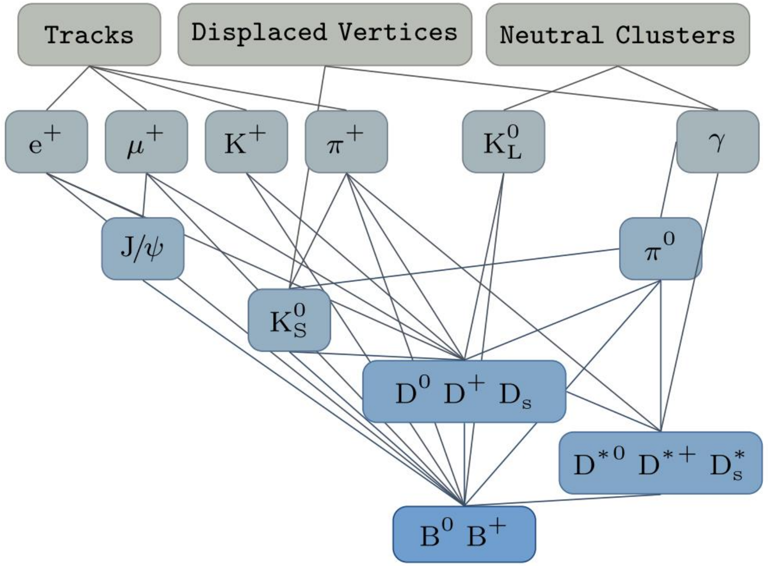


Process of FEI skim:

Events are kept if at least one of $\mathcal{O}(1000)$ considered decay chains is reconstructed.

Full Event Interpretation (FEI):

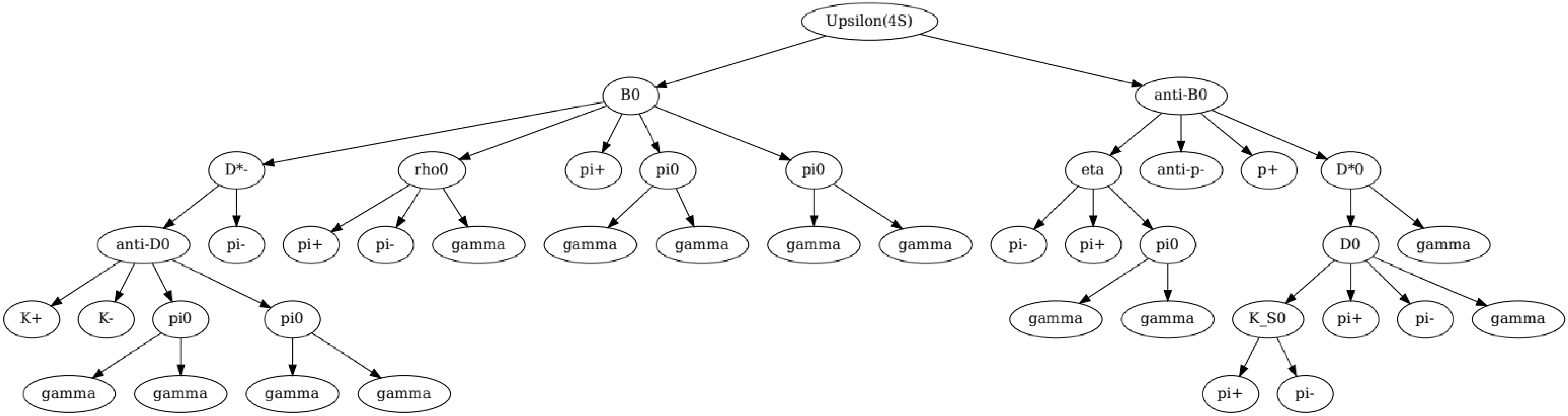
Hierarchically reconstruct from **tracks and clusters** to **intermediate particles** and finally to **B mesons**



Retention rate after FEI skim:

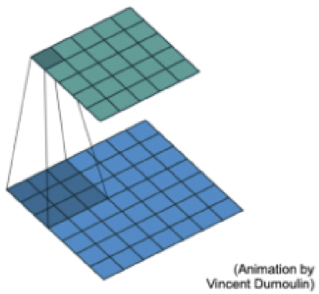
	Hadronic B^+	Hadronic B^0
Mixed ($\Upsilon(4s) \to B^0\bar{B}^0$)	5.62%	4.25%

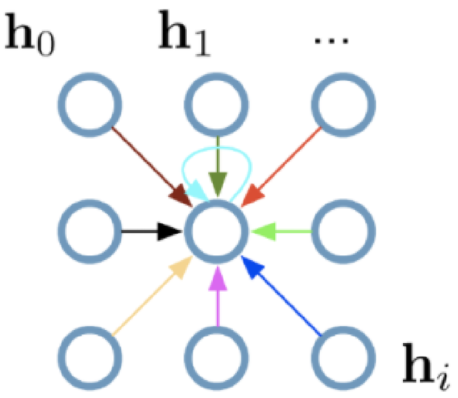
Tree Structures of Particle Decay inspire the use of Graph Neural Network



CNN vs GCN

Single CNN layer with 3x3 filter:



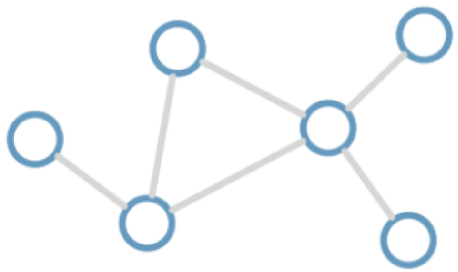


Full update:

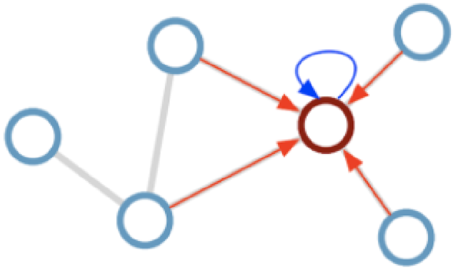
$$\mathbf{h}_4^{(l+1)} = \sigma\left(\mathbf{W}_0^{(l)}\mathbf{h}_0^{(l)} + \mathbf{W}_1^{(l)}\mathbf{h}_1^{(l)} + \cdots + \mathbf{W}_8^{(l)}\mathbf{h}_8^{(l)}\right)$$

Convolutional Neural Network (CNN)

Consider this undirected graph:



Calculate update for node in red:



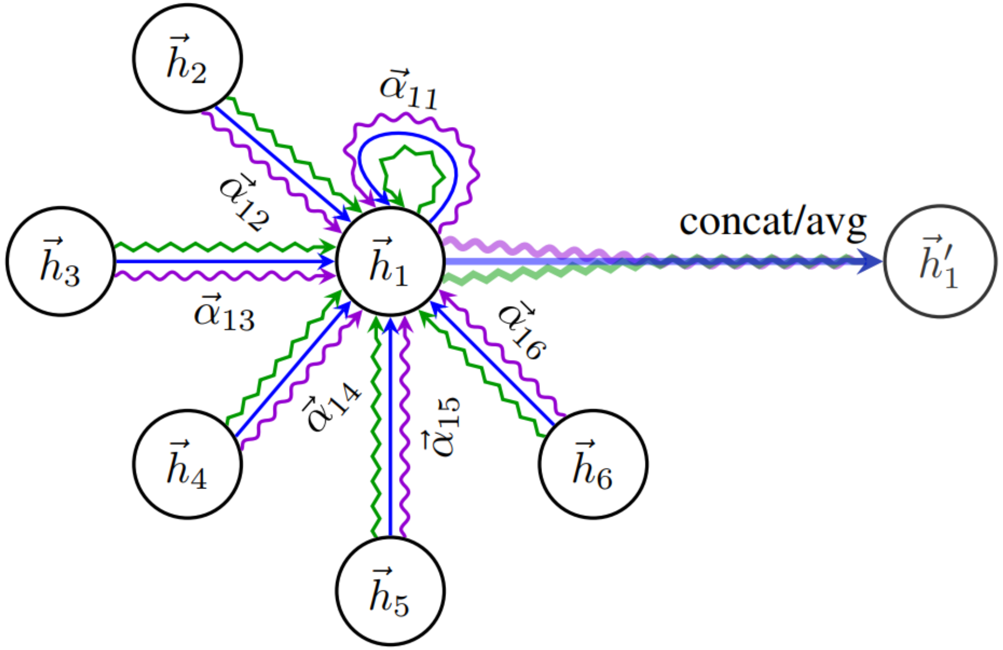
Update rule:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_j^{(l)}\right)$$

Graph Convolutional Network (GCN)

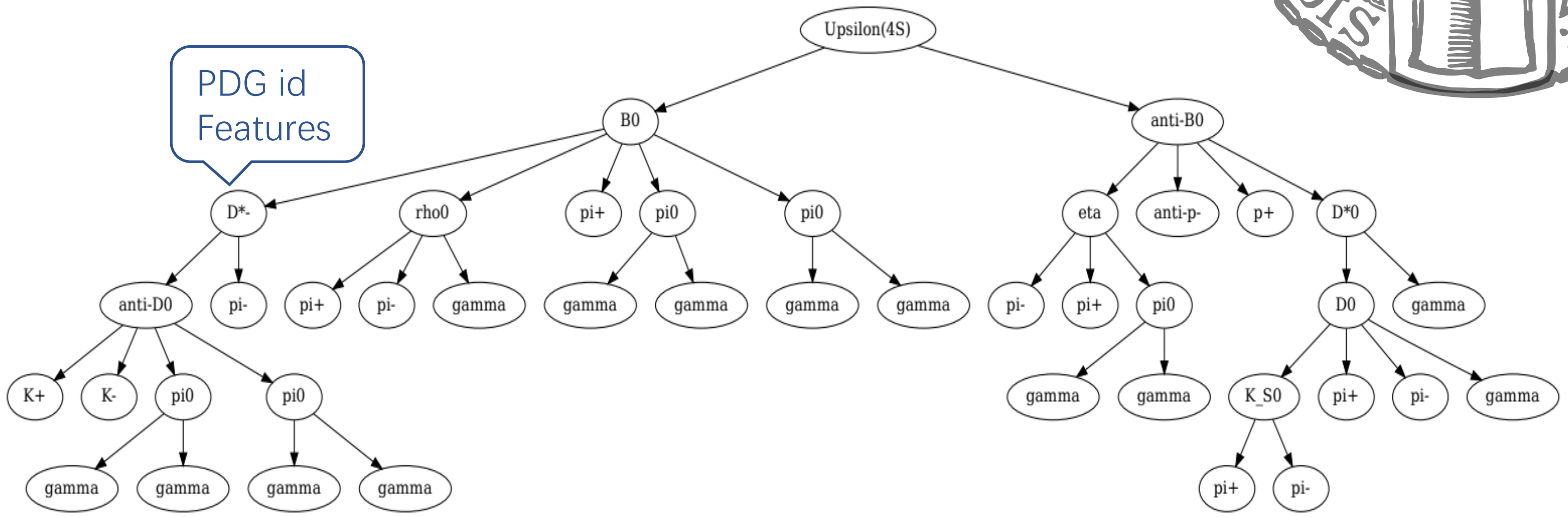
Graph Attention Networks

- Learn attention weights $\boldsymbol{\alpha}^k$ of edges from adjacent nodes
- Allow n_heads = K different heads
- Concatenate outputs from all heads



$$\vec{h}'_i = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij}^k \mathbf{W}^k \vec{h}_j\right)$$

Preparation



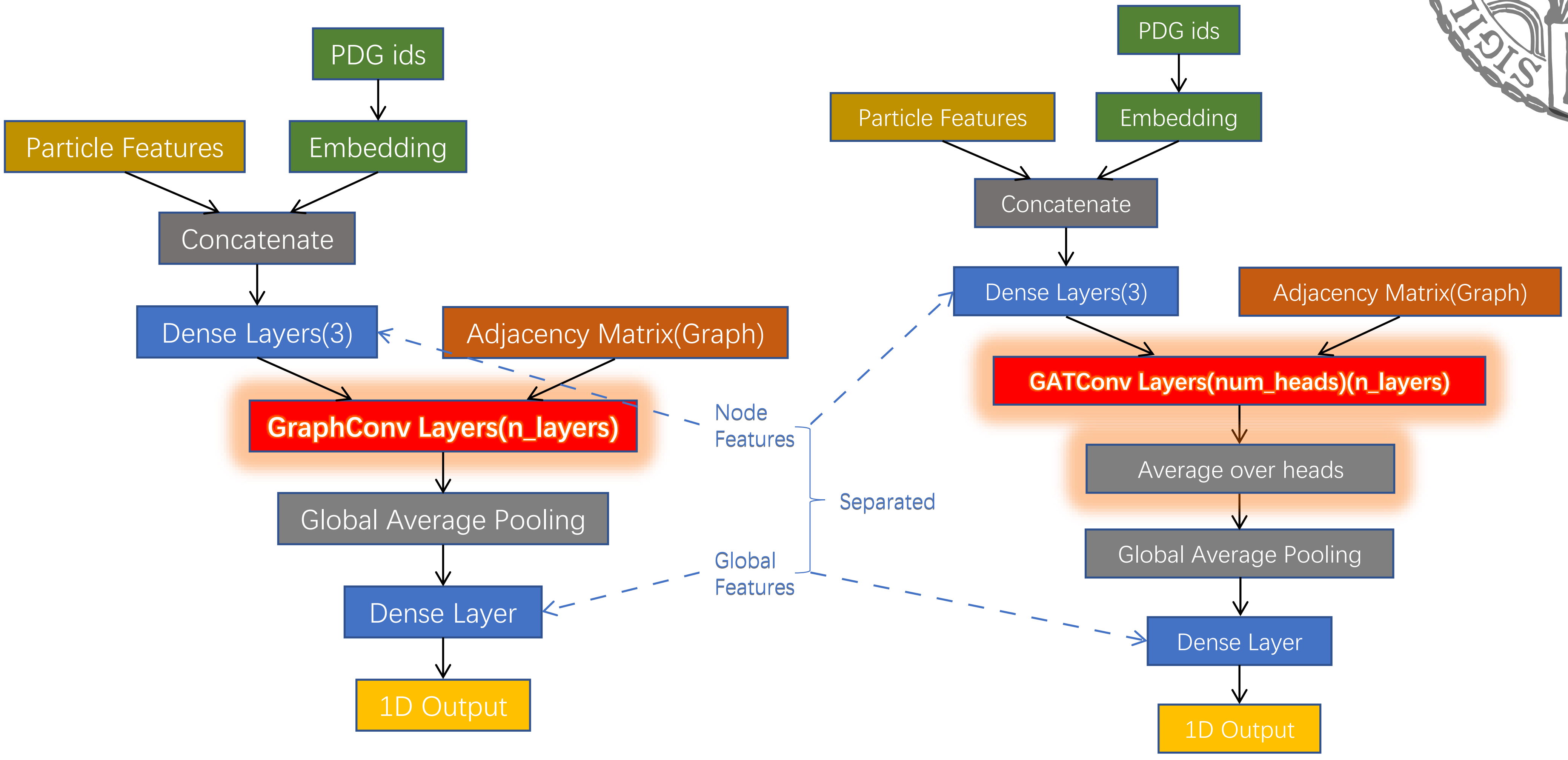
Dataset:

- Each event (each **Graph**):
  - Decay of $\Upsilon(4s) \rightarrow B^0\bar{B}^0$
  - 30-80 Particles (**Nodes**)
  - **PDG** ids
  - 8 **Features**: **Production time**, **Energy**, **Position** (3d), **Momentum** (3d)
- **Labels**: Pass/Fail after the reconstruction of B decays (FEI skims)
- Other attributions: e.g. M_{bc}
- Build graphs, shuffle and split: (number of events)

  From training set: **train: 900,000; validation: 100,000**

  From test set: **test: 500,000**

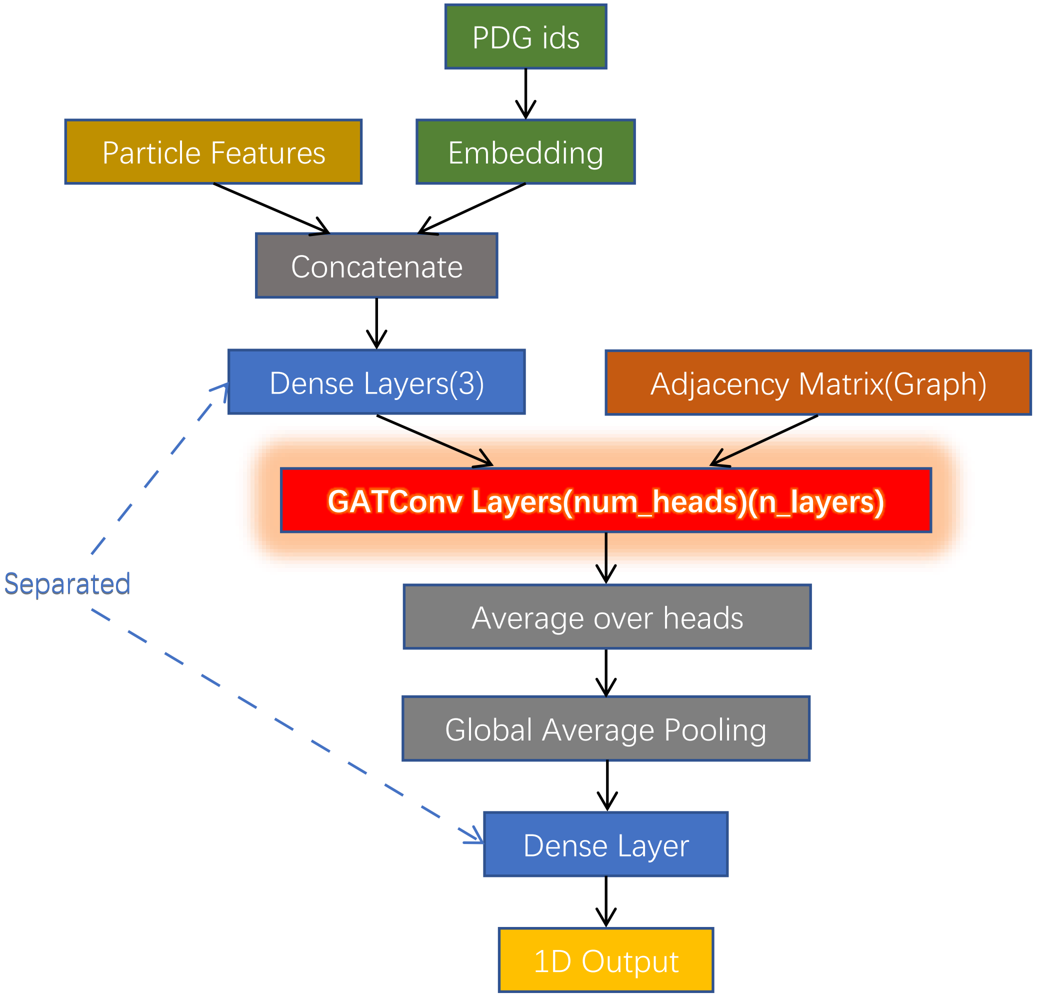
Network Structures



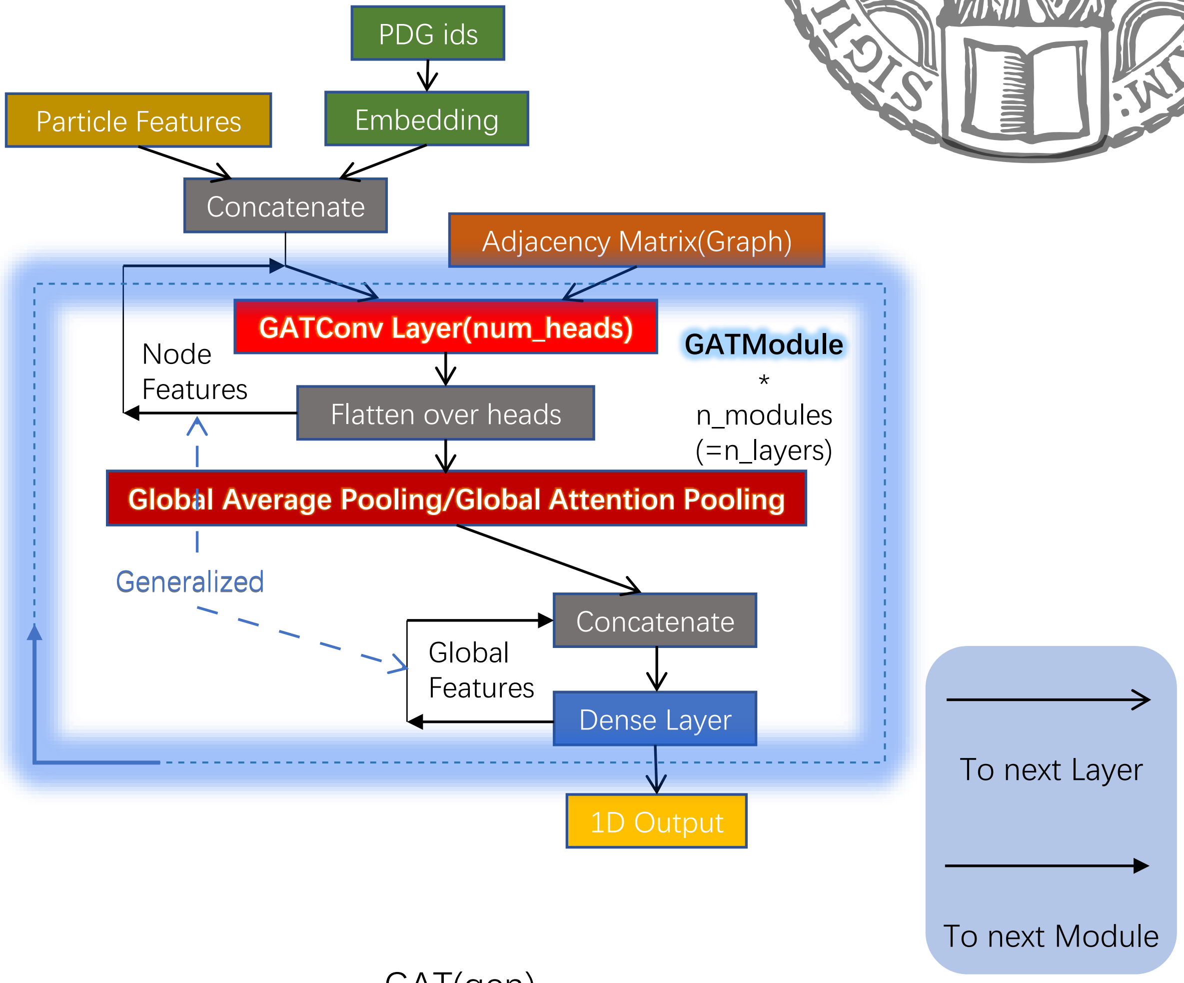
GCN(sep)

GAT(sep)

Network Structures

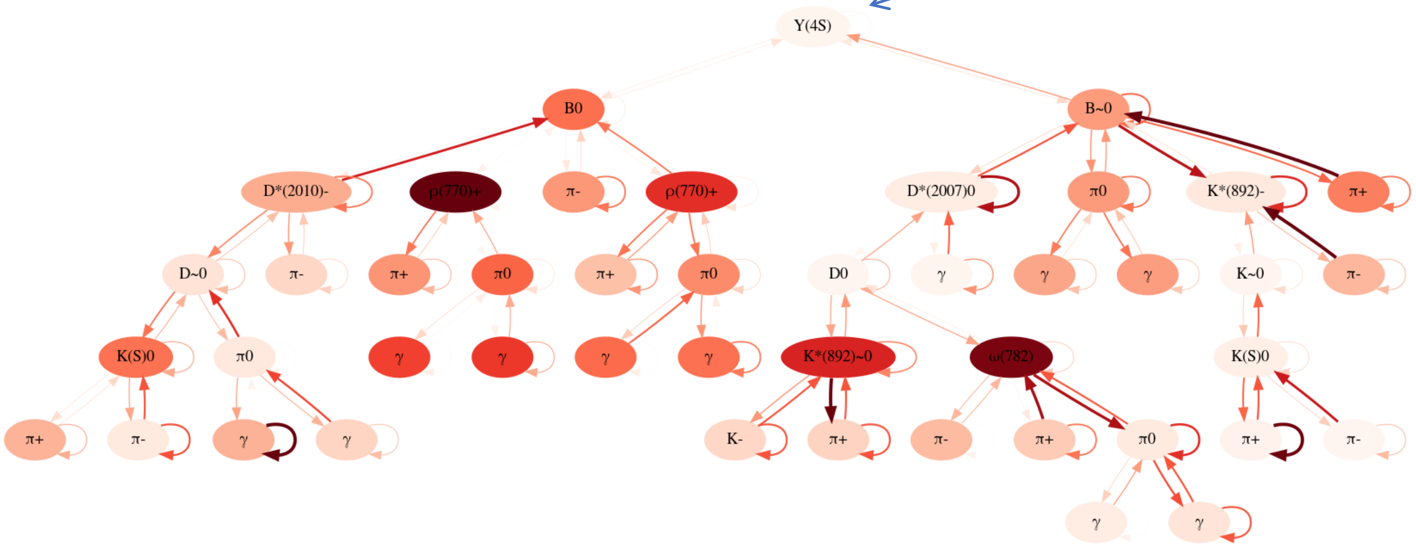


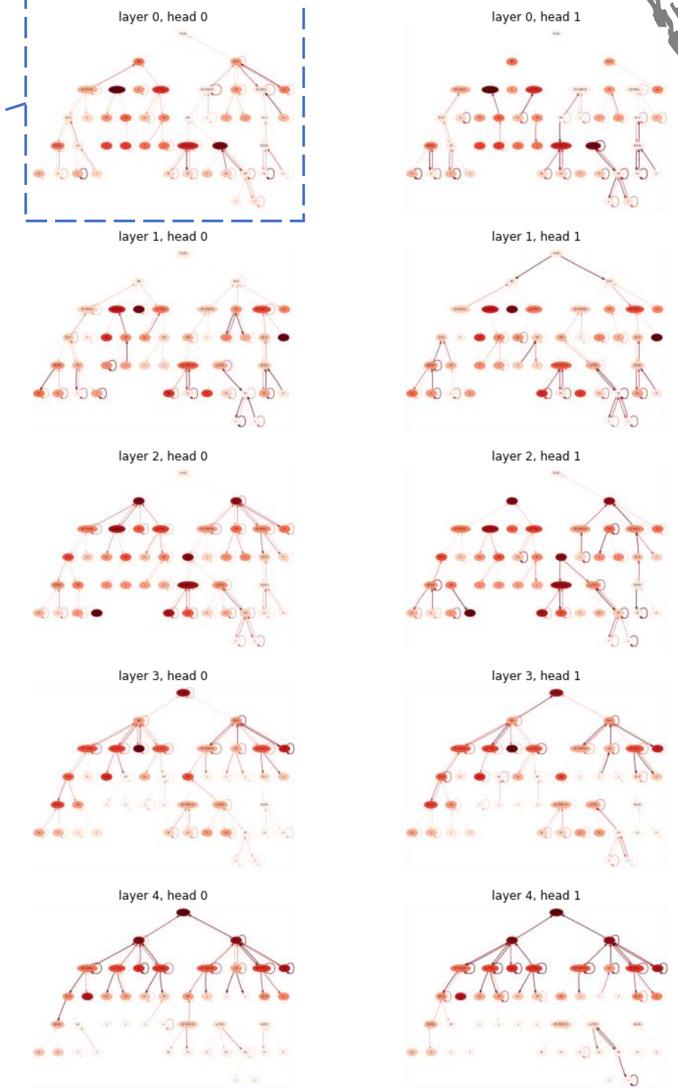
GAT(sep)



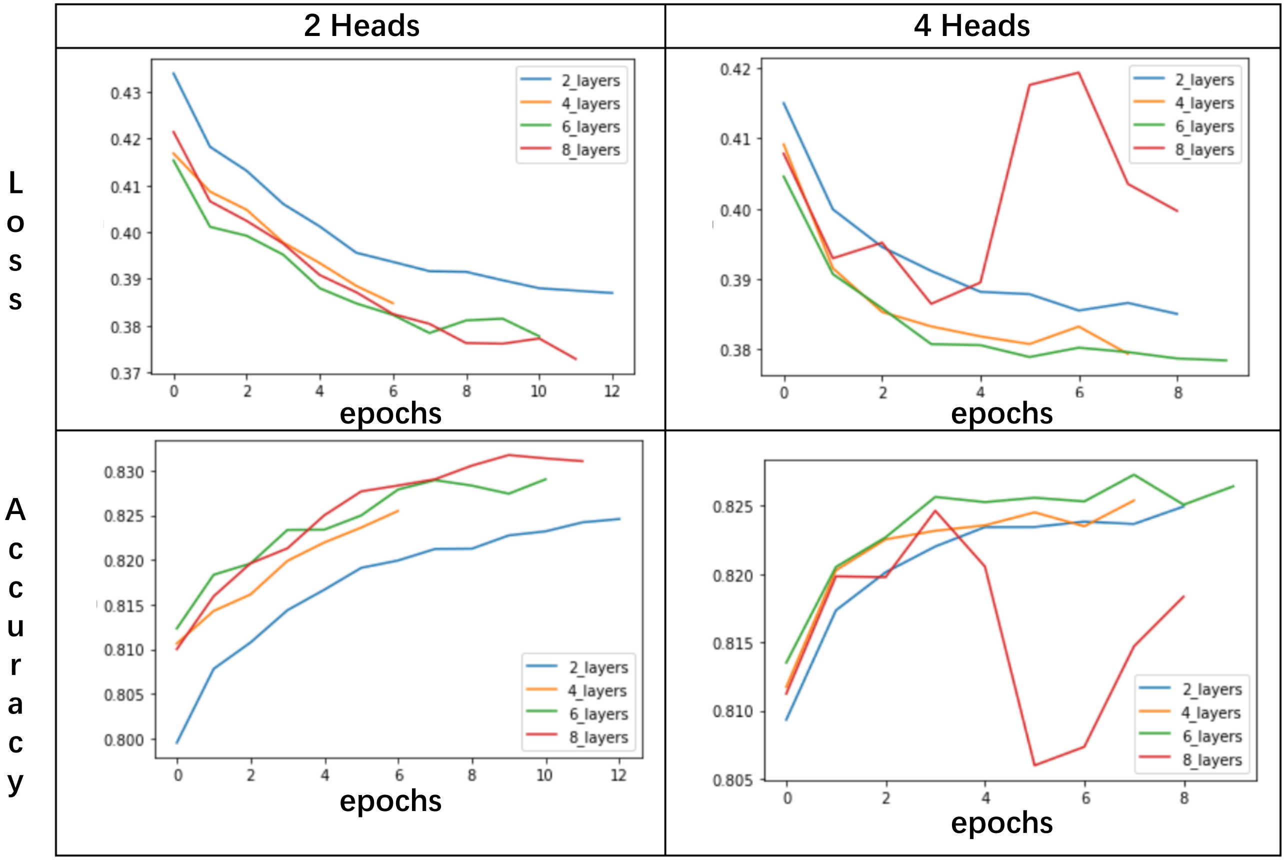
GAT(gen)
or
GAT+GAP(gen)

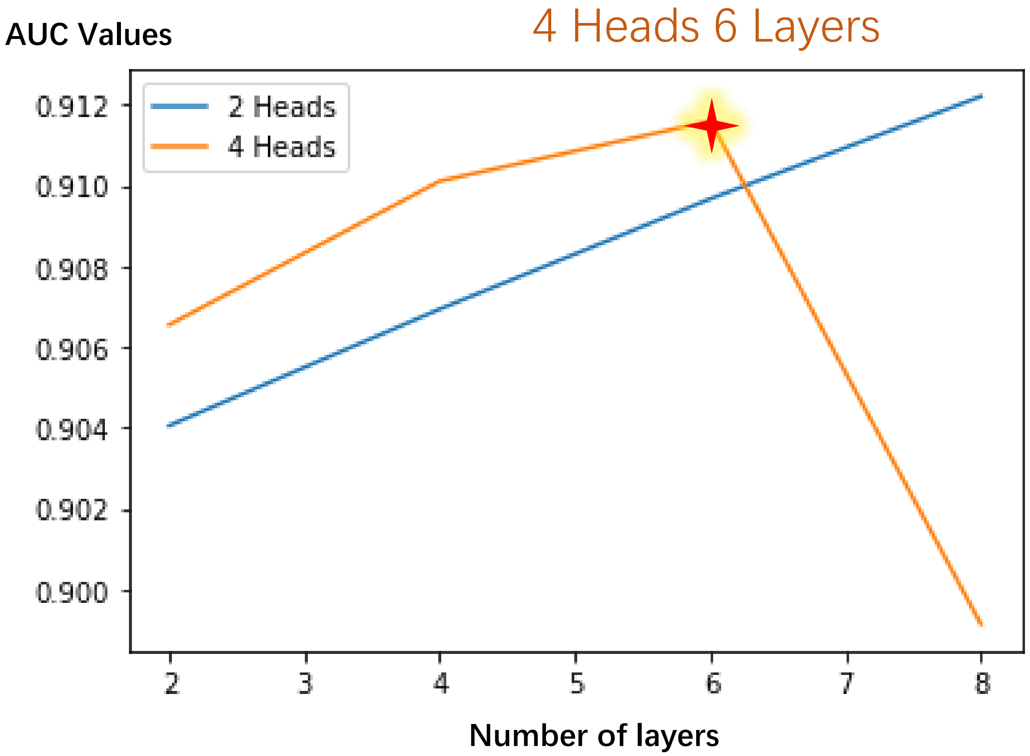
Visualization of Node/Edge-Attentions for GAT+GAP(gen)





Quantitative Studies





Comparison

Parameters:

- n_heads = 4
- n_layers = 6
- n_units = 128
- batch_size = 128
- n_train = 0.9M
- n_val = 0.1M
- n_test = 0.5M

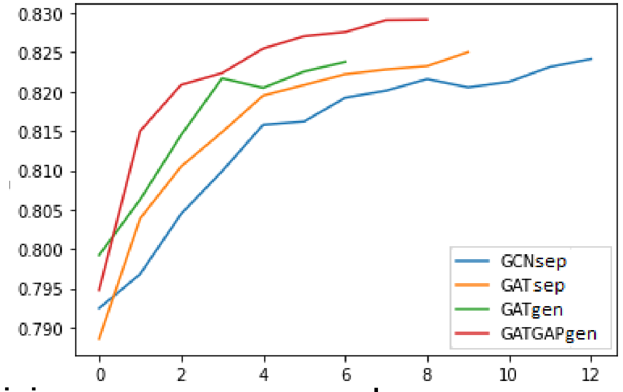
Loss:

- Entropy

EarlyStopping:

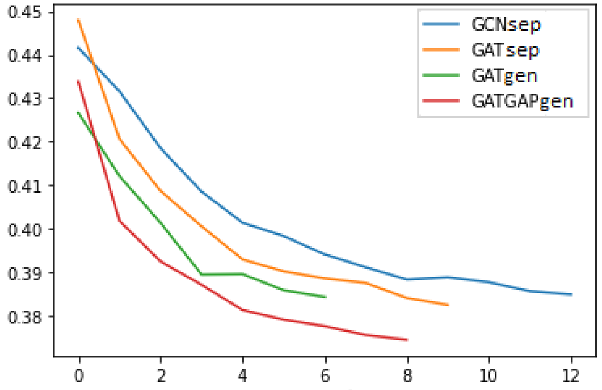
- patience = 3
- delta = 1e-5

Validation accuracy



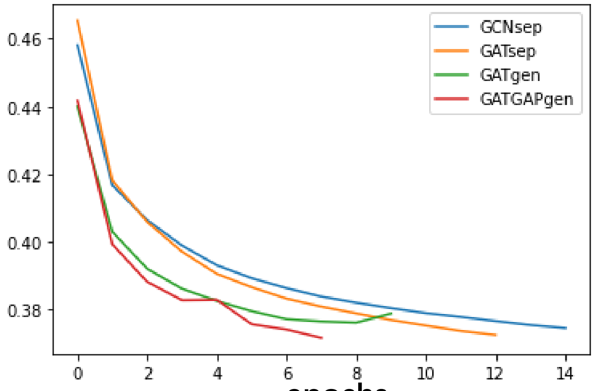
epochs

Validation loss

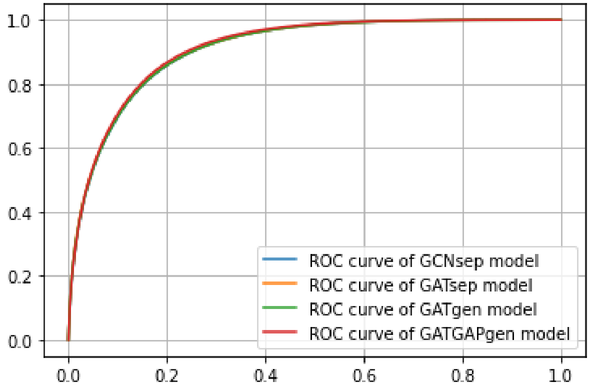


epochs

Training loss



epochs

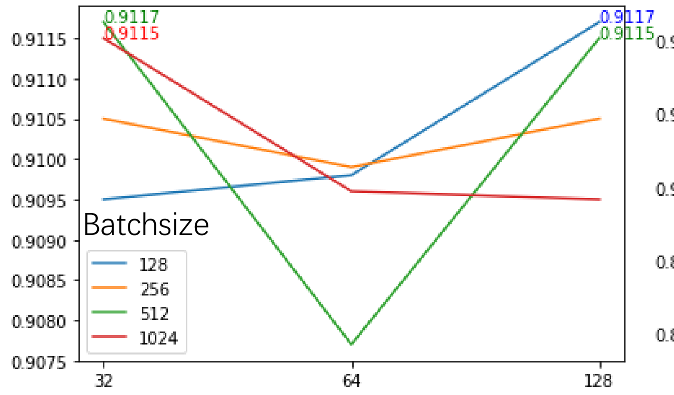


	GCN(sep)	GAT(sep)	GAT(gen)	GAT+GAP(gen)
TrainingTime	3619.46s	4047.47s	3471.48s	5049.81s
AUCValues	0.90831	0.90937	0.90891	0.91216

Evaluations

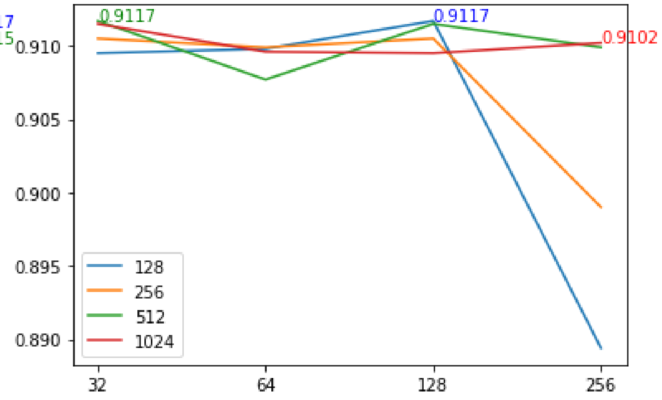
Grid Search

AUC



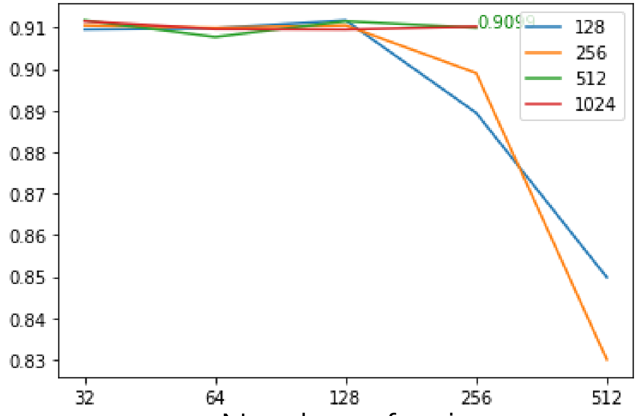
Number of units

AUC



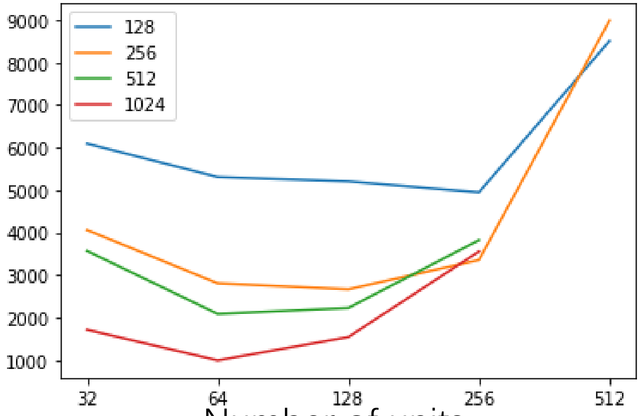
Number of units

AUC



Number of units

Training Time



Number of units

Best Combinations

Batch-size	Number of units	AUC	Training Time
128	128	0.9117	5205
256	32	0.9105	4061
256	128	0.9105	2666
512	32	0.9117	3568
512	128	0.9115	2228
1024	32	0.9115	1716
1024	256	0.9102	3556

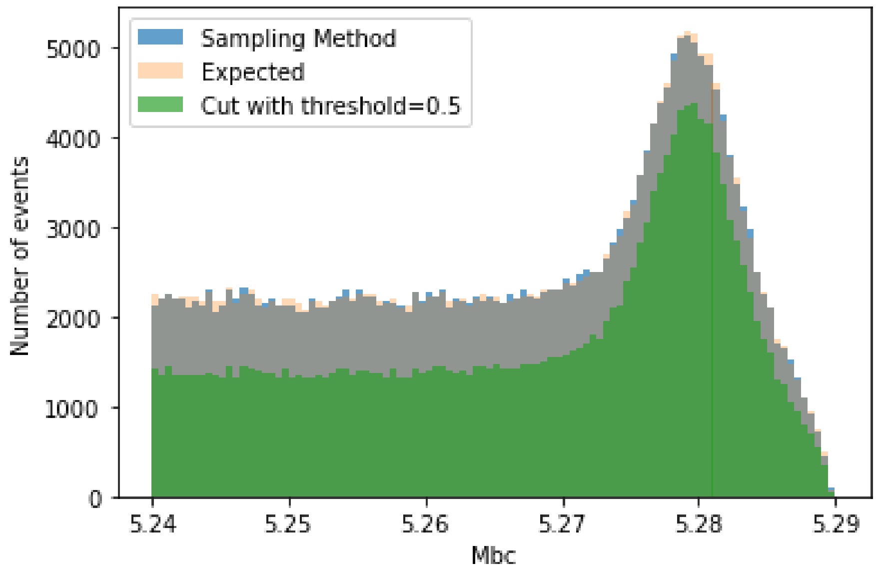
Network Sizes

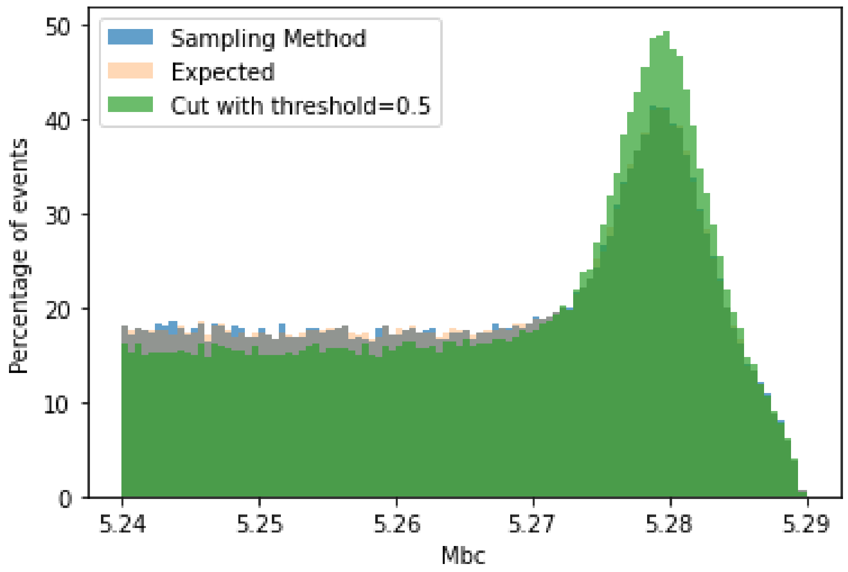
# Units	# Parameters
32	120,527
64	459,951
128	1,808,495
256	7,184,367
512	28,651,247

Sampling

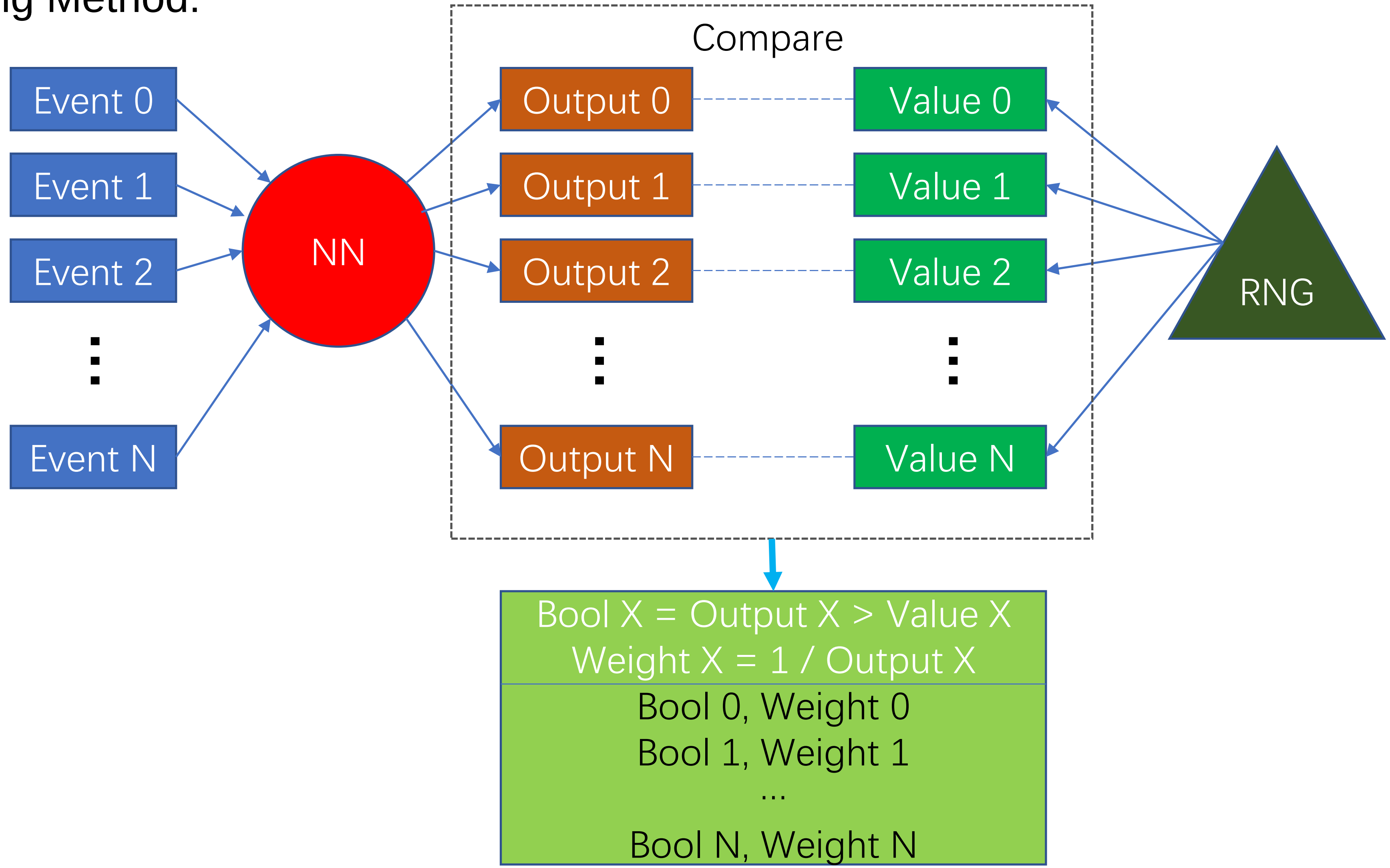
- Previous method: Cut according to neural network outputs
- Problem: Inevitable bias

- Our method: Sampling with probability given by neural network outputs
- Problem: Statistical uncertainty





Sampling Method:



Relative statistical uncertainty and effective sample size

Variable	Formula	Remark
NN outputs / Probabilities to pass	$\{p_i\}$	'i' refers to each event in the whole sample (batch)
Weights	$\{\omega_i\} = \left\{\frac{1}{p_i}\right\}$	Infinities (at $p_i = 0$) are excluded and set to 0 Avoid the bias by construction
Relative statistical uncertainty	$S = \frac{\sqrt{\sum \omega_i^2 p_i}}{\sum \omega_i p_i}$	$\sum \omega_i^2 p_i = \sum \omega_i$ $\sum \omega_i p_i = N$ Here consider only passed events (label = 1)
Effective sample size	$N_{eff} = \frac{1}{S}$	Number of events needed to reach the same statistical uncertainty without sampling

Speedup rate

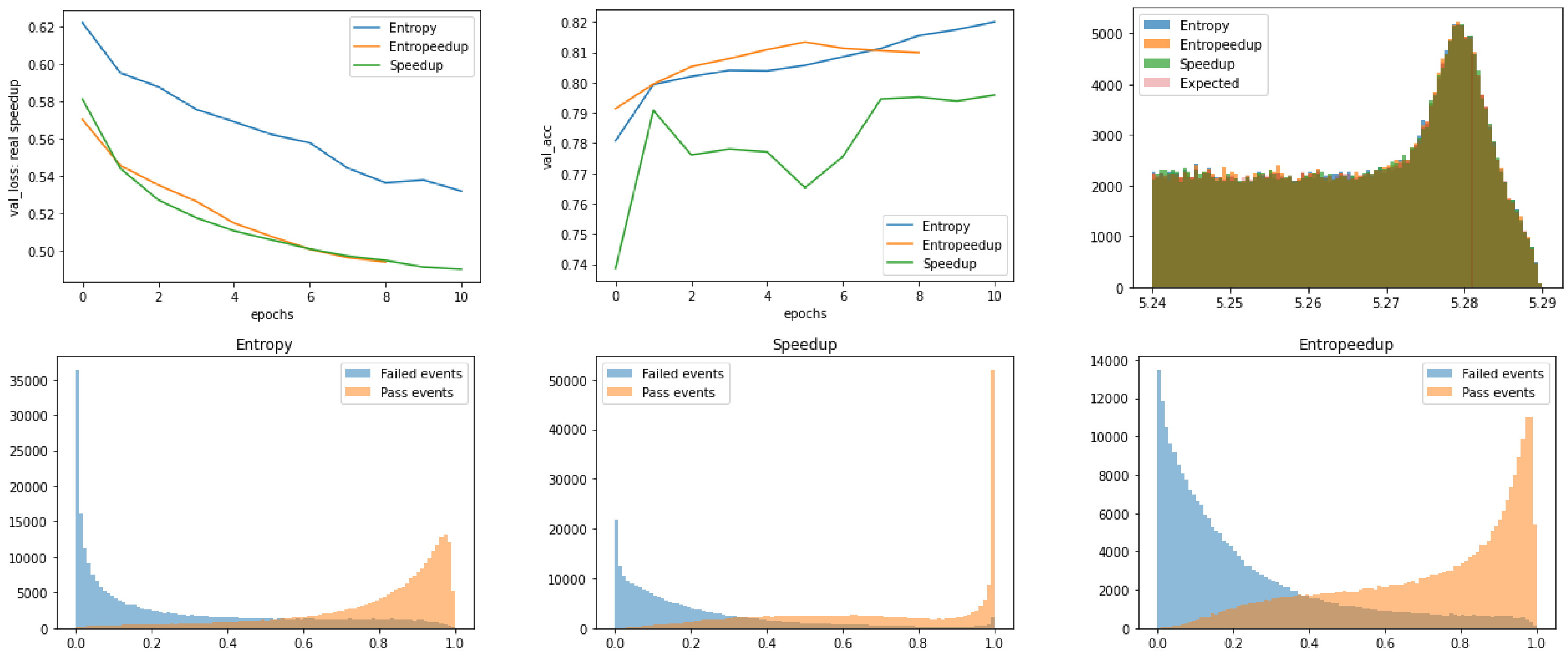
Variable	Formula	Remark
Skim retention rate	$r = 0.05$	Probability to pass the skim process
Times of different phases in ms	$t_{gen} = 0.08$ $t_{NN} = 0.63$ $t_{SR} = 97.04$	Time for MC Generation, NN and Det. Sim and Reconstruction Taken from previous studies
(Inverse) Speedup rate	$R = \frac{t_{filter}}{t_{no\_filter}} \sim \frac{1}{N_{eff}}$	Ratio between time consumings of the whole work flows with and without NN filters for producing **the same effective sample size**

Performances of different training losses

Speedup always as validation loss!

Entropy: Binary cross entropy between NN outputs $\{p_i\}$ and skim results $\{y_i \in \{0,1\}\}$
Speedup: (Inverse) Speedup rate R
Entropeedup: Training-epoch-dependent combination of Entropy and Speedup

$$\text{Entropeedup} = e^{-epoch/10}\,Entropy + Speedup$$



Results:

- Graph Neural Networks with attention mechanism work well for selective background monte carlo simulation
- Find some best combinations of hyperparameters of NN structure specified for our task
- Sampling method with minimization of Speedup loss : Could achieve a factor 2 speedup without introducing bias

Next steps:

- Improve the performance of sampling by trying other sampling methods and other losses
- Study speedup loss for general cases, study dependence on different time assumptions
- Generalize to other datasets

Thank You for your Attention

Boyang Yu [1], Nikolai Hartmann [1], Thomas Kuhr [1]
[1] *Ludwig-Maximilians-Universität München*
IDT-UM Collaboration Meeting, May 11th, 2021

Speedup rate

Variable	Formula	Remark
Skim retention rate	$r = 0.05$	Probability to pass the skim process
Times of different phases in ms	$t_{gen} = 0.08$ $t_{NN} = 0.63$ $t_{SR} = 97.04$	Taken from previous studies
Effective number of events after sampling	$n_+ = \sum p_i$ $n_- = \sum (1 - p_i)$	$\{p_i\}$ will be devided into two subsets where the events will/won't pass the skim process
Time consuming with NN filter	$t_+ = [n_{TP}r + n_{FP}(1-r)]\left(t_{gen} + t_{NN} + t_{SR}\right)$ $t_- = [n_{FN}r + n_{TN}(1-r)]\left(t_{gen} + t_{NN}\right)$	Positive/Negative: Result of sampling True/False: Result of sampling == skim process
Time consuming without NN	$t_0 = N_{eff}\left(t_{gen} + t_{NN}\right)$	To reach the same statistical uncertainty
(Inverse) Speedup rate	$R = \dfrac{t_+ + t_-}{t_0}$	The lower the better