

Fast and Faithful Generation of Calorimeter Showers with Deep Generative Models: CALOFLOW

— The Mitchell Conference on Collider, Dark Matter,
and Neutrino Physics 2022 —

Claudius Krause

Rutgers, The State University of New Jersey

May 25, 2022

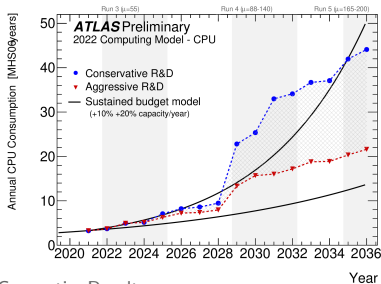
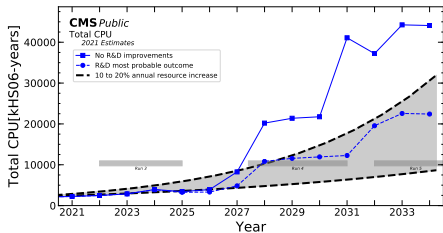


RUTGERS

UNIVERSITY | NEW BRUNSWICK

In collaboration with David Shih
arXiv: 2106.05285 and 2110.11377

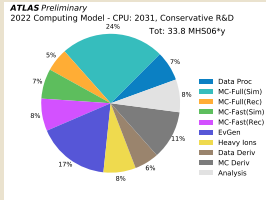
Deep Generative Models will be crucial for the LHC.



<https://twiki.cern.ch/twiki/bin/view/CMSPublic/CMSOfflineComputingResults>

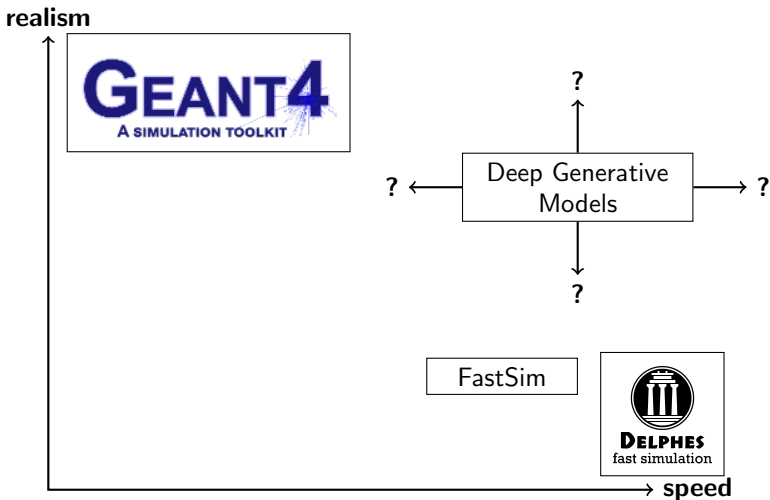
<https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/UPGRADE/CERN-LHCC-2022-005/>

- At the start of LHC Run 4, the computational needs will likely exceed the available budget.
- A large fraction goes into simulation.



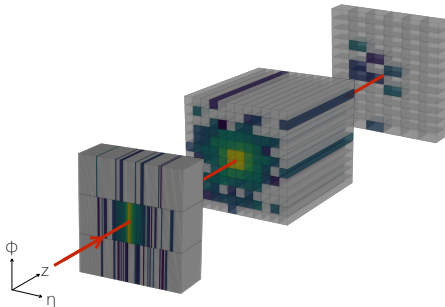
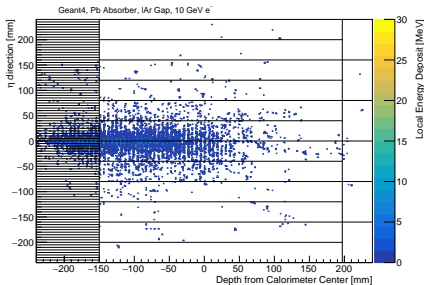
CERN-LHCC-2022-005

Detector Simulation needs to be fast and faithful



We use the same calorimeter geometry as CALOGAN

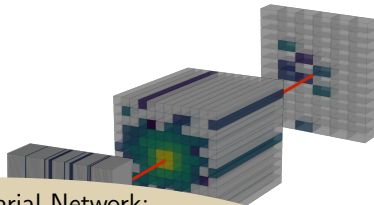
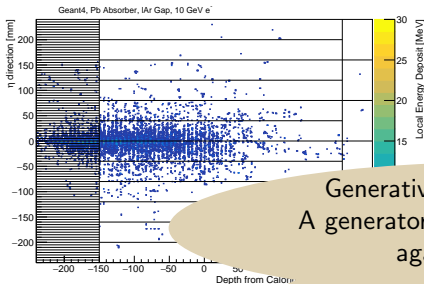
- We consider a simplified version of the ATLAS ECal: flat alternating layers of lead and LAr
- They form three instrumented layers of dimension 3×96 , 12×12 , and 12×6



CaloGAN: Paganini, de Oliveira, Nachman [1705.02355, PRL; 1712.10321, PRD]

We use the same calorimeter geometry as CALOGAN

- We consider a simplified version of the ATLAS ECal: flat alternating layers of lead and LAr
- They form three instrumented layers of dimension 3×96 , 12×12 , and 12×6

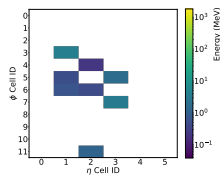
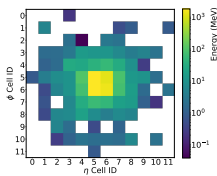
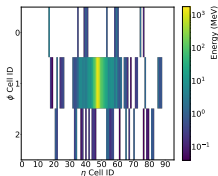


Generative Adversarial Network:
A generator and a critic play a game
against each other.

CaloGAN: Paganini, de Oliveira, Nachman [1705.02355, PRL; 1712.10321, PRD]

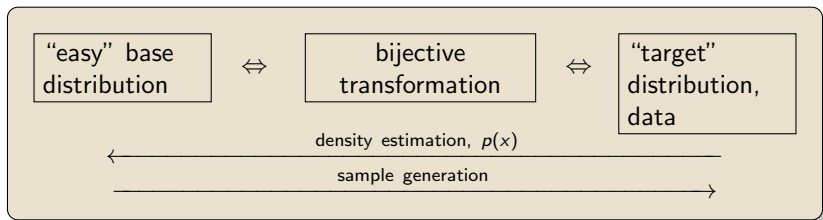
We use the same calorimeter geometry as CALOGAN

- The GEANT4 configuration of CALOGAN is available at <https://github.com/hep-lbdl/CaloGAN>
- We produce our own dataset: available at [DOI: 10.5281/zenodo.5904188]
- Showers of e^+ , γ , and π^+ (100k each)
- All are centered and perpendicular
- E_{inc} is uniform in [1, 100] GeV and given in addition to the energy deposits per voxel:



CaloGAN: Paganini, de Oliveira, Nachman [1705.02355, PRL; 1712.10321, PRD]

Normalizing Flows learn a change-of-coordinates efficiently.



Normalizing Flows ...

Dinh et al. [arXiv:1410.8516],

Rezende/Mohamed [arXiv:1505.05770], Review: Papamakarios et al. [arXiv:1912.02762]

- ... learn the parameters of a series of easy transformations.

- Each transformation has an analytic Jacobian and inverse.

⇒ We use a piecewise Rational Quadratic Spline.

Durkan et al.

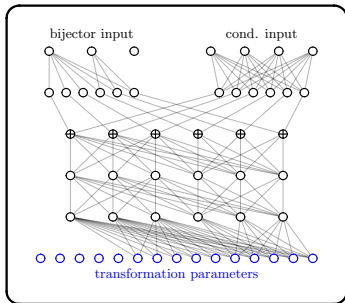
[arXiv:1906.04032]

- An autoregressive architecture ensures a triangular Jacobian.

⇒ Can be obtained by masking a DNN.

Masking Ensures the Autoregressive Property.

MADE Block



Implementation via masking:

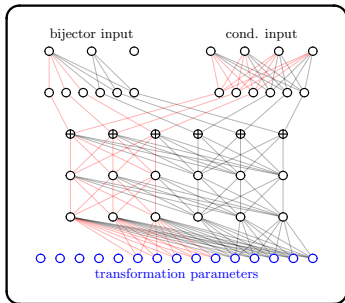
- a single “forward” pass gives the full output of all $p(x_i|x_{i-1} \dots x_1)$.
⇒ very fast
- the “inverse” needs to loop through all dimensions and gets a single $p(x_i|x_{i-1} \dots x_1)$ each time.
⇒ very slow

Germain/Gregor/Murray/Larochelle [arXiv:1502.03509]

- Masked Autoregressive Flow (MAF), introduced in Papamakarios et al. [arXiv:1705.07057], are slow in sampling and fast in inference.
- Inverse Autoregressive Flow (IAF), introduced in Kingma et al. [arXiv:1606.04934], are fast in sampling and slow in inference.

Masking Ensures the Autoregressive Property.

MADE Block



Implementation via masking:

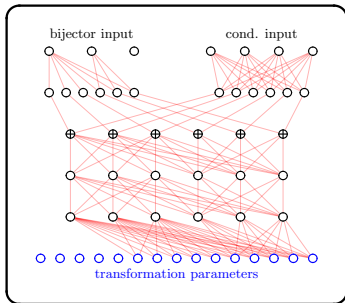
- a single “forward” pass gives the full output of all $p(x_i|x_{i-1} \dots x_1)$.
⇒ very fast
- the “inverse” needs to loop through all dimensions and gets a single $p(x_i|x_{i-1} \dots x_1)$ each time.
⇒ very slow

Germain/Gregor/Murray/Larochelle [arXiv:1502.03509]

- Masked Autoregressive Flow (MAF), introduced in Papamakarios et al. [arXiv:1705.07057], are slow in sampling and fast in inference.
- Inverse Autoregressive Flow (IAF), introduced in Kingma et al. [arXiv:1606.04934], are fast in sampling and slow in inference.

Masking Ensures the Autoregressive Property.

MADE Block



Implementation via masking:

- a single “forward” pass gives the full output of all $p(x_i|x_{i-1} \dots x_1)$.
⇒ very fast
- the “inverse” needs to loop through all dimensions and gets a single $p(x_i|x_{i-1} \dots x_1)$ each time.
⇒ very slow

Germain/Gregor/Murray/Larochelle [arXiv:1502.03509]

- Masked Autoregressive Flow (MAF), introduced in Papamakarios et al. [arXiv:1705.07057], are slow in sampling and fast in inference.
- Inverse Autoregressive Flow (IAF), introduced in Kingma et al. [arXiv:1606.04934], are fast in sampling and slow in inference.

CALOFLOW uses a 2-step approach to learn $p(\vec{\mathcal{I}}|E_{\text{inc}})$.

Flow I

- learns $p_1(E_0, E_1, E_2|E_{\text{inc}})$
- is a MAF that is optimized using the LL.

Flow II

- learns $p_2(\hat{\mathcal{I}}|E_0, E_1, E_2, E_{\text{inc}})$ of normalized showers
- in CALOFLOW v1 (2106.05285 — called “teacher”):

- MAF trained with LL
- Slow in sampling ($\approx 500\times$ slower than CALOGAN)

- in CALOFLOW v2 (2110.11377 — called “student”):

- IAF trained with Probability Density Distillation from teacher (LL prohibitive)
i.e. matching IAF parameters to van den Oord et al. [1711.10433] frozen MAF
- Fast in sampling ($\approx 500\times$ faster than CALOFLOW v1)

A Classifier provides the “ultimate metric”.

According to the Neyman-Pearson Lemma we have:

- The likelihood ratio is the most powerful test statistic to distinguish the two samples.
- A powerful classifier trained to distinguish the samples should therefore learn (something monotonically related to) this.
- If this classifier is confused, we conclude $p_{\text{GEANT4}}(x) = p_{\text{generated}}(x)$

⇒ This captures the full 504-dim. space.

? But why wasn't this used before?

⇒ Previous deep generative models were separable to almost 100%!

DCTRGAN: Diefenbacher et al. [2009.03796, JINST]

CALOFLOW passes the “ultimate metric” test.

According to the Neyman-Pearson Lemma we have:

$p_{\text{GEANT4}}(x) = p_{\text{generated}}(x)$ if a classifier cannot distinguish data from generated samples.

AUC		DNN based classifier		
		GEANT4 vs. CALOGAN	GEANT4 vs. (teacher) CALOFLOW v1	GEANT4 vs. (student) CALOFLOW v2
e^+	unnorm.	1.000(0)	0.859(10)	0.786(7)
	norm.	1.000(0)	0.870(2)	0.824(4)
γ	unnorm.	1.000(0)	0.756(48)	0.758(14)
	norm.	1.000(0)	0.796(2)	0.760(3)
π^+	unnorm.	1.000(0)	0.649(3)	0.729(2)
	norm.	1.000(0)	0.755(3)	0.807(1)

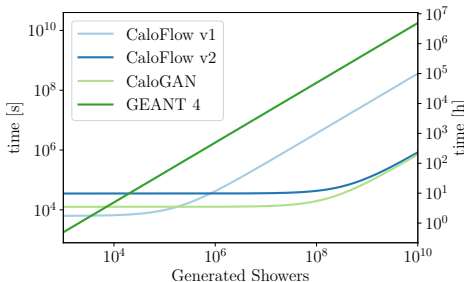
AUC ($\in [0.5, 1]$): Area Under the ROC Curve, smaller is better, i.e. more confused

Sampling Speed: The Student beats the Teacher!

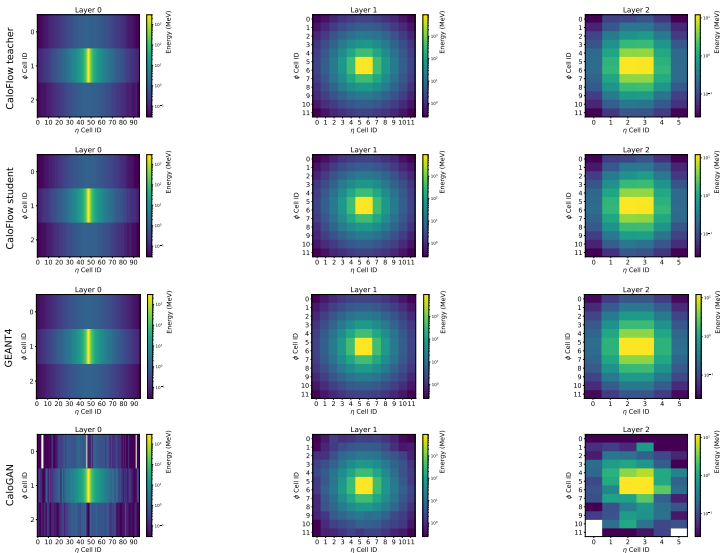
	CALOFLOW*		CALOGAN*	GEANT4†	
	teacher	student			
training	22+82 min	+ 480 min	210 min	0 min	
generation	time per shower				
batch size			batch size req.	100k req.	
10	835 ms	5.81 ms	455 ms	2.2 ms	1772 ms
100	96.1 ms	0.60 ms	45.5 ms	0.3 ms	1772 ms
1000	41.4 ms	0.12 ms	4.6 ms	0.08 ms	1772 ms
10000	36.2 ms	0.08 ms	0.5 ms	0.07 ms	1772 ms

*: on our TITAN V GPU

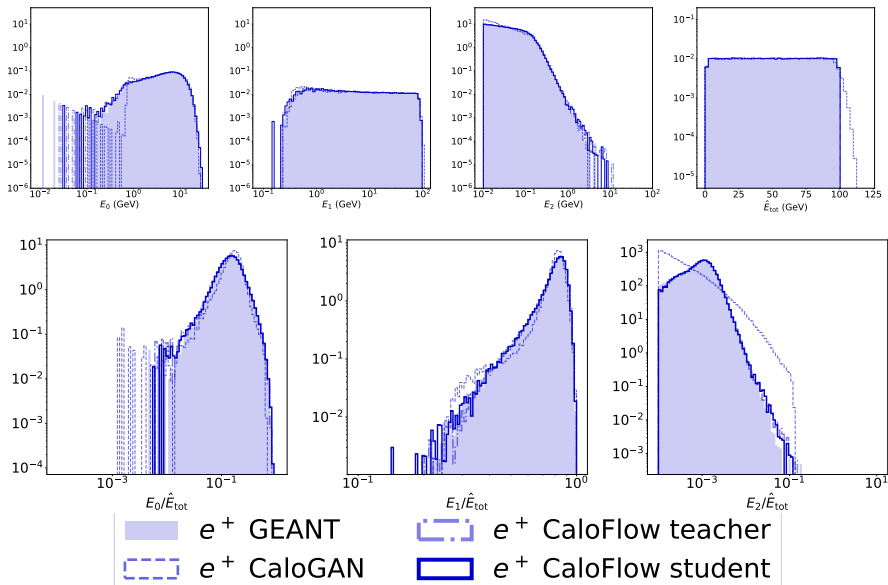
†: on the CPU of CaloGAN: Paganini, de Oliveira, Nachman [1712.10321, PRD]



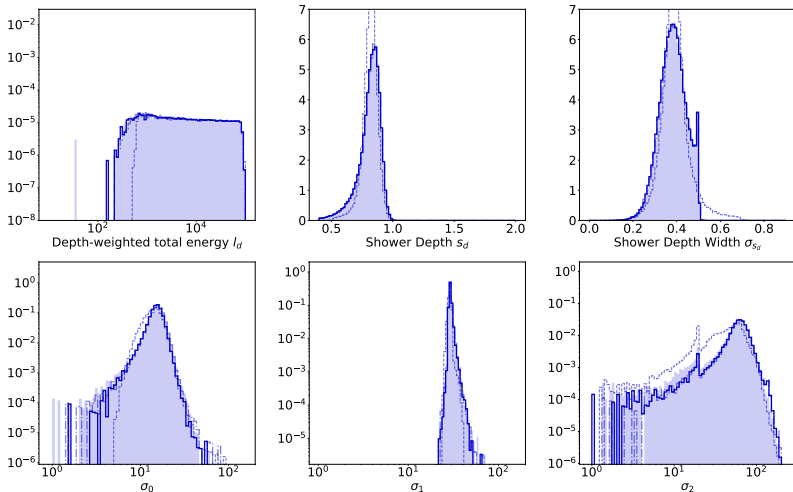
CALOFLOW: Comparing Shower Averages: e^+



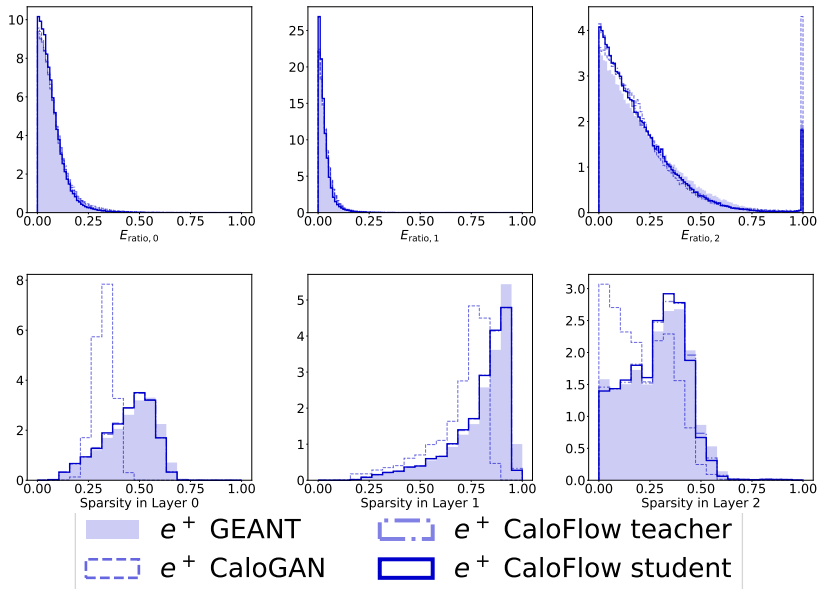
CALOFLOW: Flow I histograms: e^+



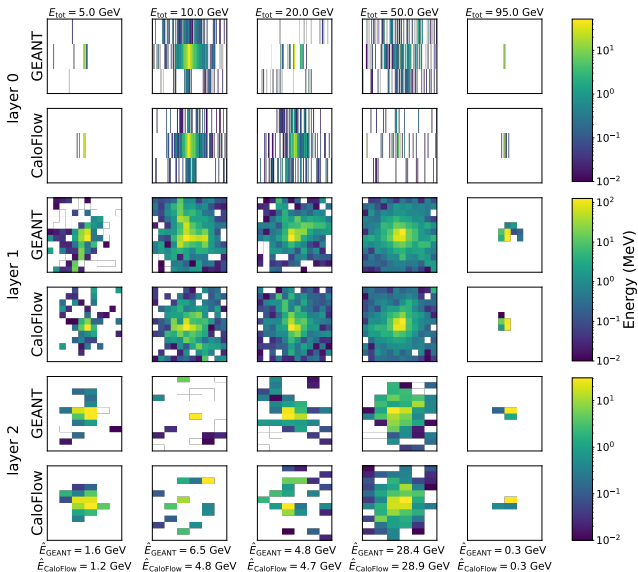
CALOFLOW: Flow I+II histograms: e^+



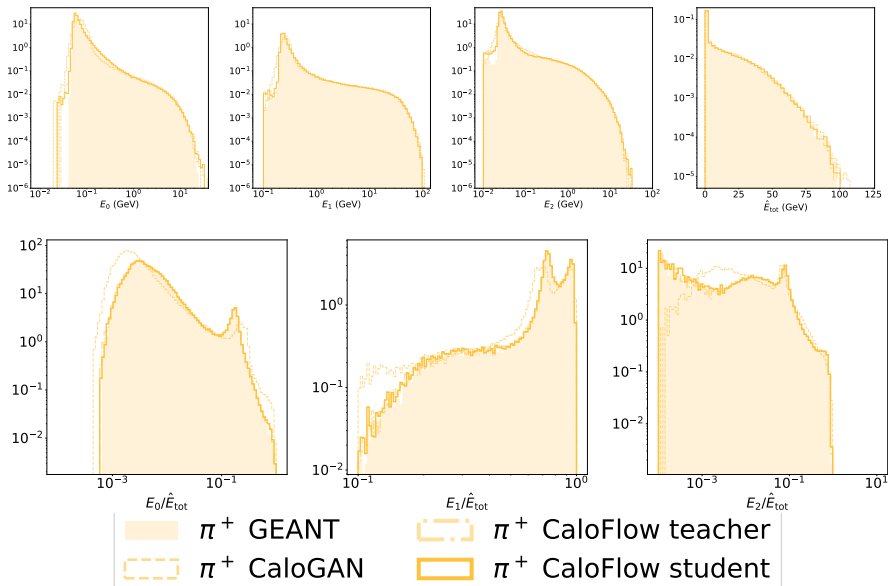
CALOFLOW: Flow II histograms: e^+



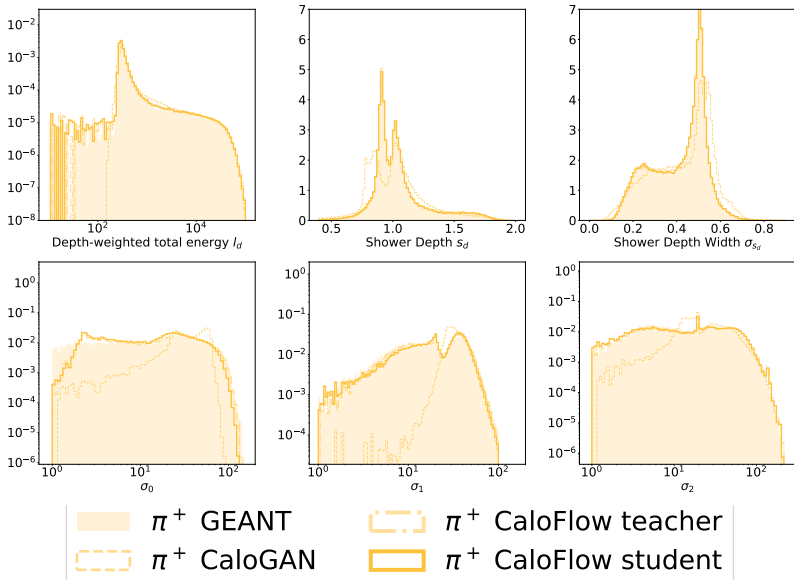
CALOFLOW: Nearest Neighbors: π^+ (student)



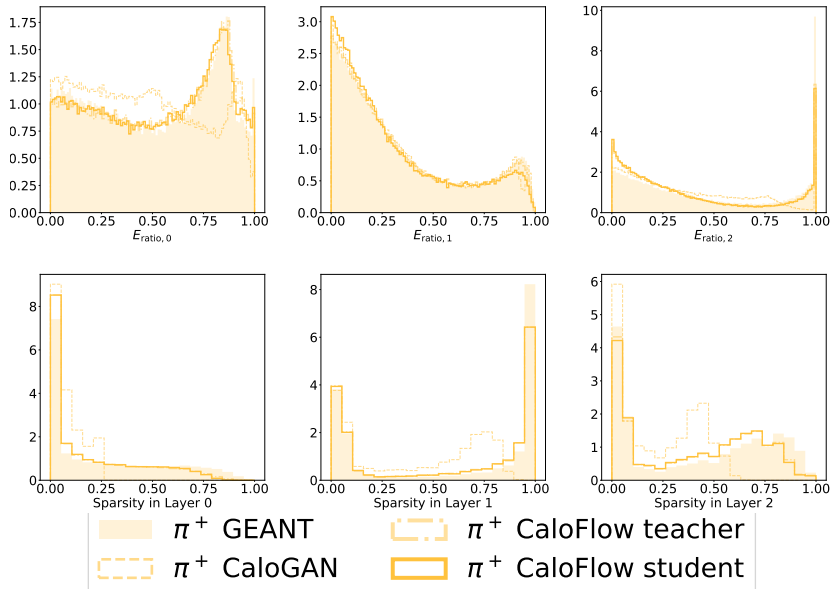
CALOFLOW: Flow 1 histograms: π^+



CALOFLOW: Flow I+II histograms: π^+



CALOFLOW: Flow II histograms: π^+



A little Advertisement — CaloChallenge 2022

Welcome to the home of the Fast Calorimeter Simulation Challenge 2022!

Homepage for the Fast Calorimeter Simulation Challenge 2022

[View on GitHub](#)

Welcome to the home of the Fast Calorimeter Simulation Challenge 2022!

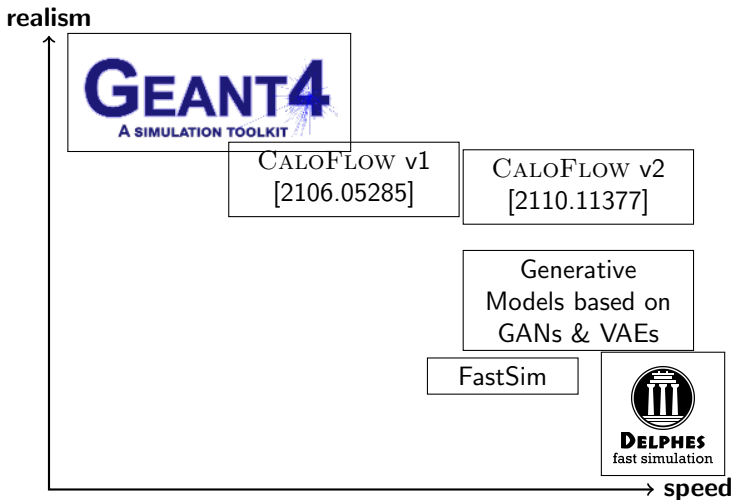
This is the homepage for the Fast Calorimeter Simulation Data Challenge. The purpose of this challenge is to spur the development and benchmarking of fast and high-fidelity calorimeter shower generation. Currently, generating calorimeter showers of elementary particles (electrons, photons, pions, ...) using GEANT4 is a major computational bottleneck at the LHC, and it is forecast to overwhelm the computing budget of the LHC in the near future. Therefore there is an urgent need to

Michele Fauci Giannelli, Gregor Kasieczka, Claudius Krause, Ben Nachman, Dalila Salamani, David Shih, and Anna Zaborowska

⇒ <https://calochallenge.github.io/homepage/>

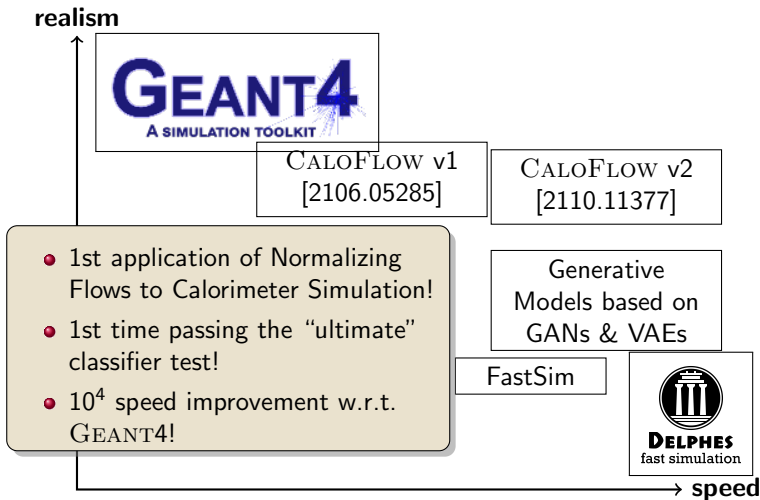
Fast and Faithful Generation of Calorimeter Showers with Deep Generative Models: CALOFLOW

— Summary —



Fast and Faithful Generation of Calorimeter Showers with Deep Generative Models: CALOFLOW

— Summary —



Backup

The Bijector is a chain of “easy” transformations.

Each transformation

- must be invertible and have analytical Jacobian

- is chosen to factorize:

$$\vec{C}(\vec{x}; \vec{p}) = (C_1(x_1; p_1), C_2(x_2; p_2), \dots, C_n(x_n; p_n))^T,$$

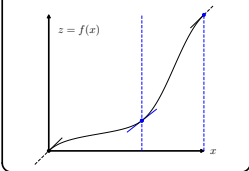
where \vec{x} are the coordinates to be transformed and \vec{p} the parameters of the transformation.

Rational Quadratic Splines:

Durkan et al. [arXiv:1906.04032]

Gregory/Delbourgo [IMA Journal of Numerical Analysis, '82]

Rational Quadratic Spline Transformation

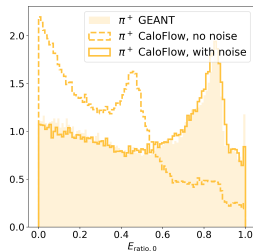
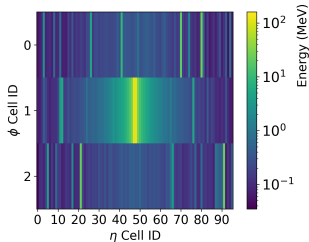
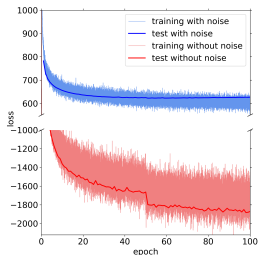


$$C = \frac{a_2\alpha^2 + a_1\alpha + a_0}{b_2\alpha^2 + b_1\alpha + b_0}$$

- numerically easy
- expressive

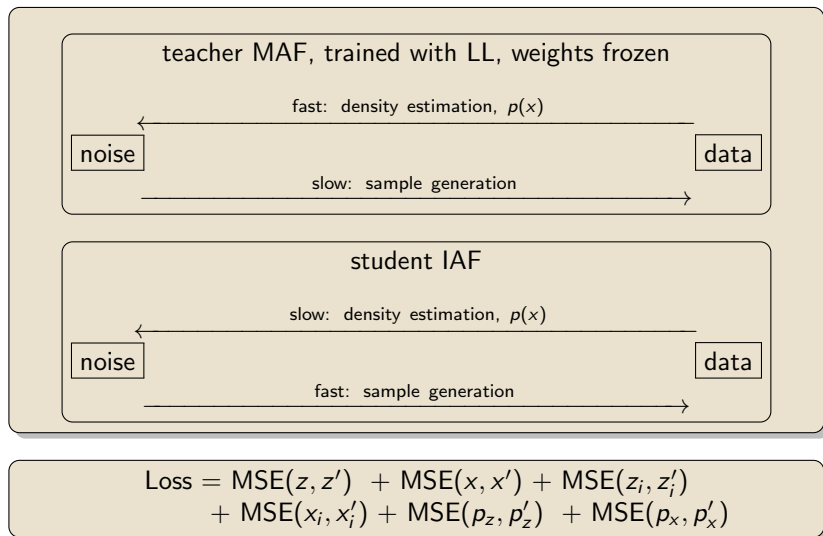
The NN predicts the bin widths, heights, and derivatives that go in a_i & b_i .

Adding Noise is important for the sampling quality.

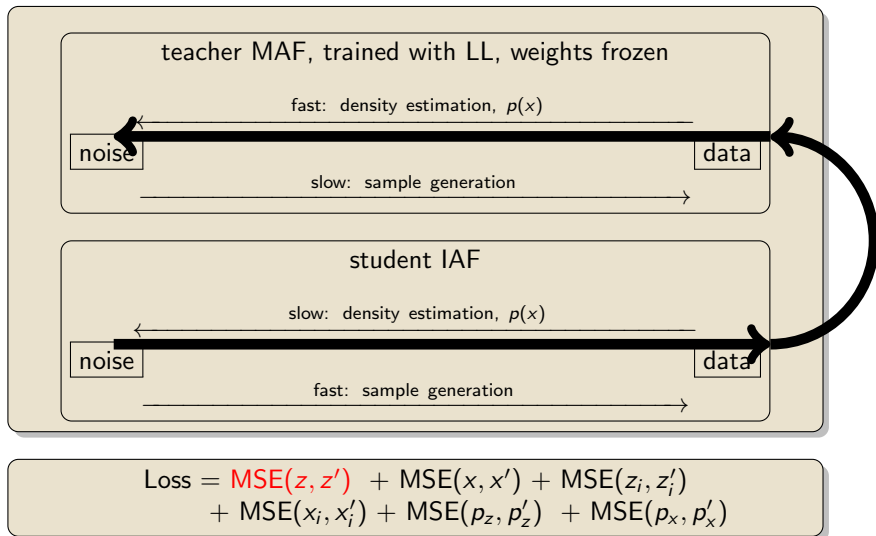


- The log-likelihood is less noisy, but smaller. Yet, the quality of the samples is much better!
- This is due to a “wider” mapping of space and less overfitting.

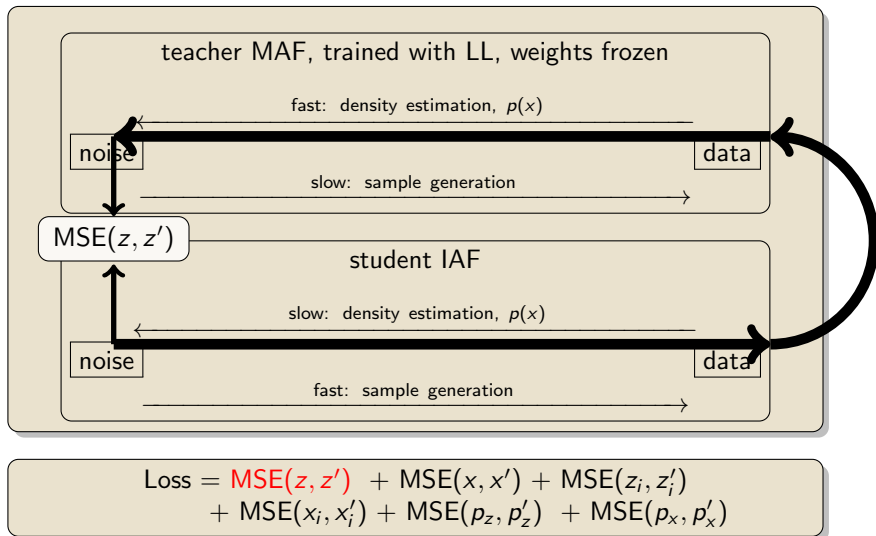
Probability Density Distillation passes the information from the teacher to the student



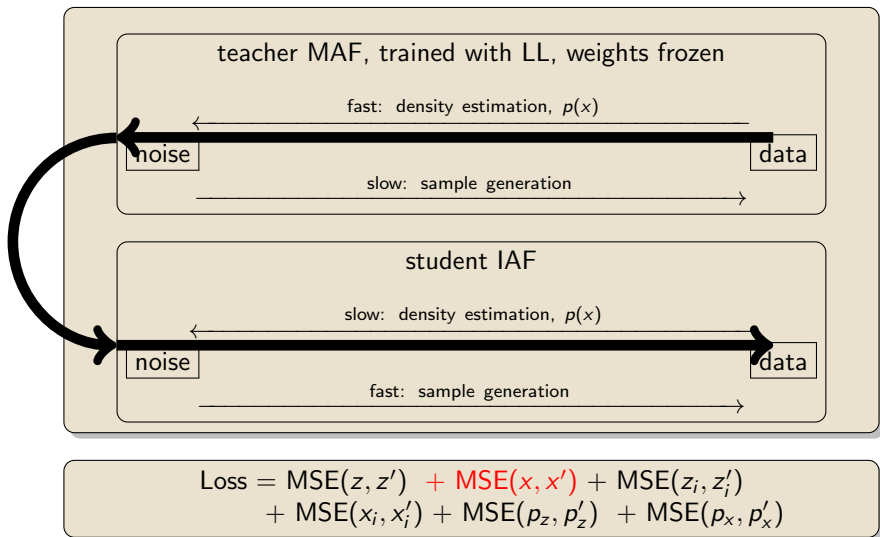
Probability Density Distillation passes the information from the teacher to the student



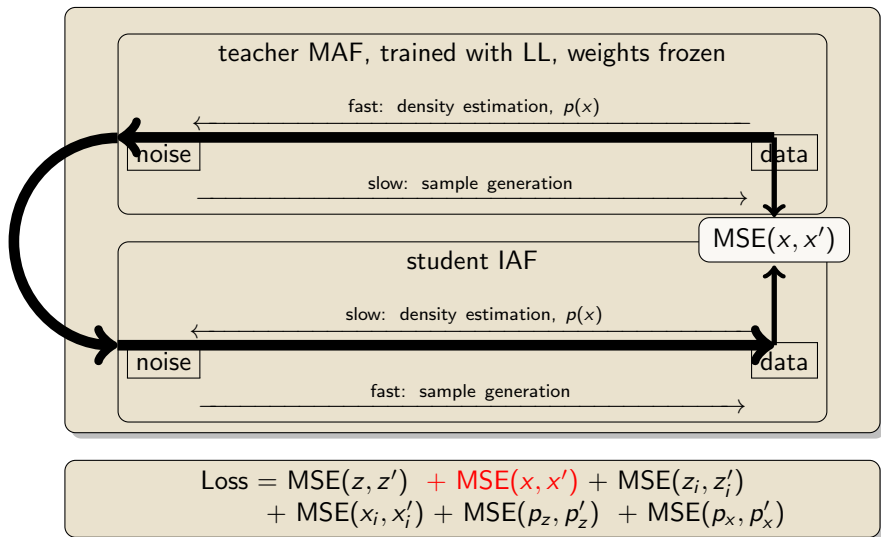
Probability Density Distillation passes the information from the teacher to the student



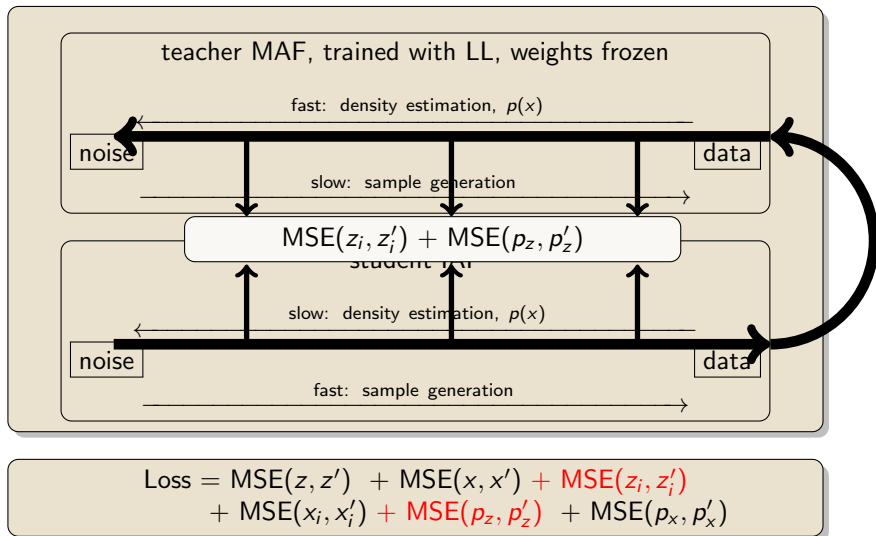
Probability Density Distillation passes the information from the teacher to the student



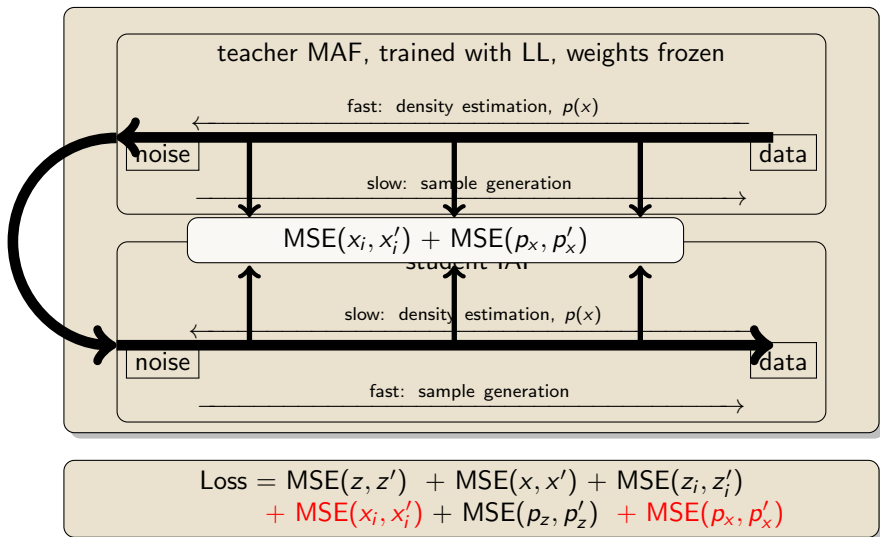
Probability Density Distillation passes the information from the teacher to the student



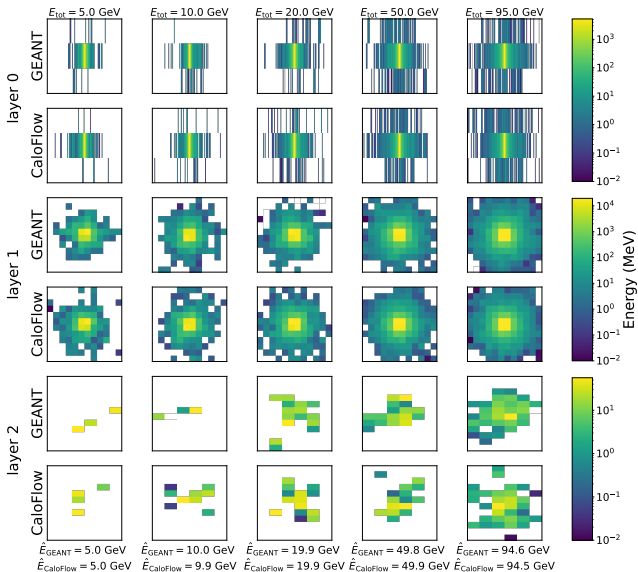
Probability Density Distillation passes the information from the teacher to the student



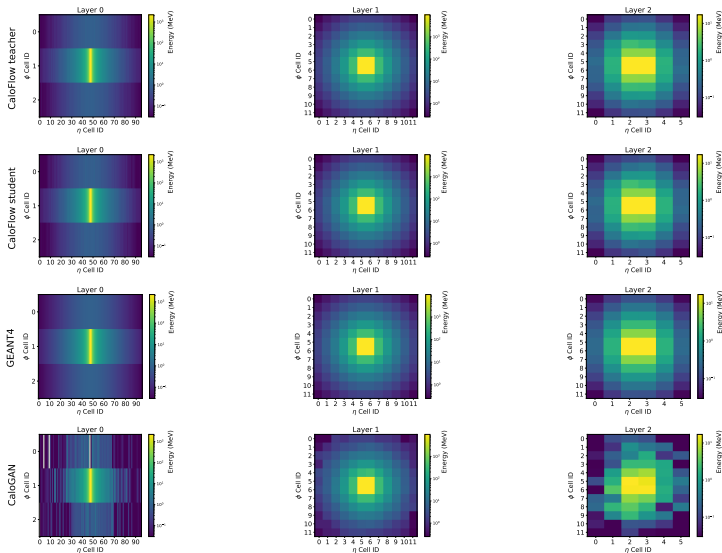
Probability Density Distillation passes the information from the teacher to the student



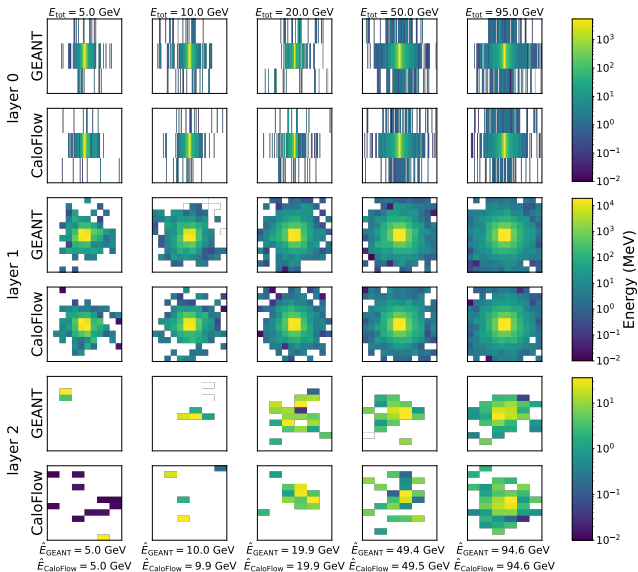
Nearest Neighbors: e^+ (student)



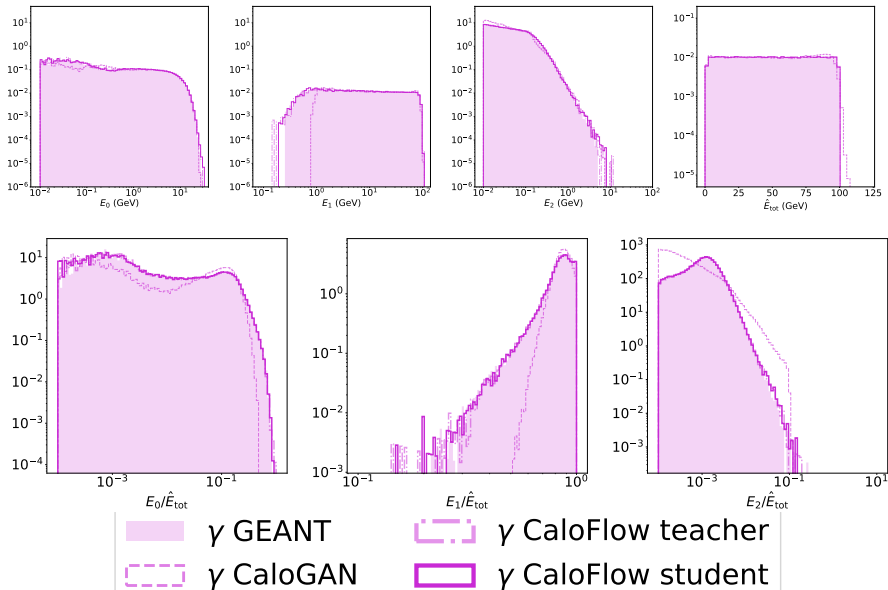
Comparing Shower Averages: γ



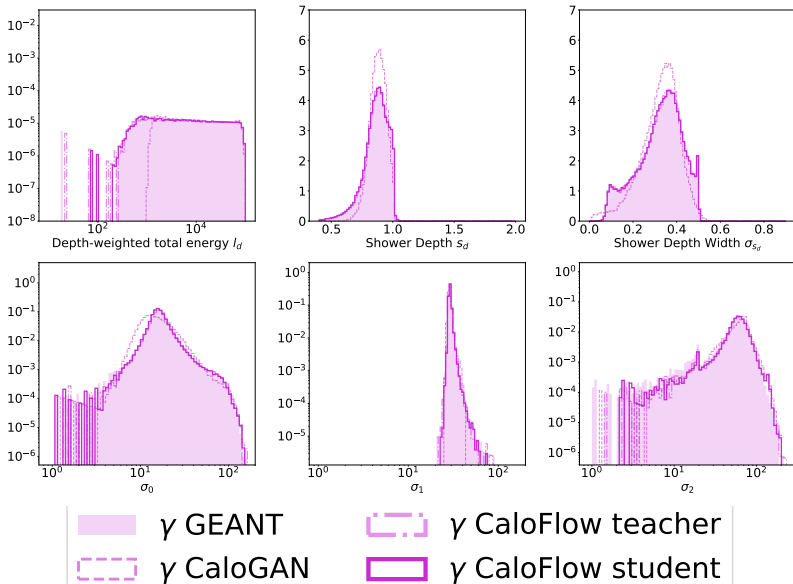
Nearest Neighbors: γ (student)



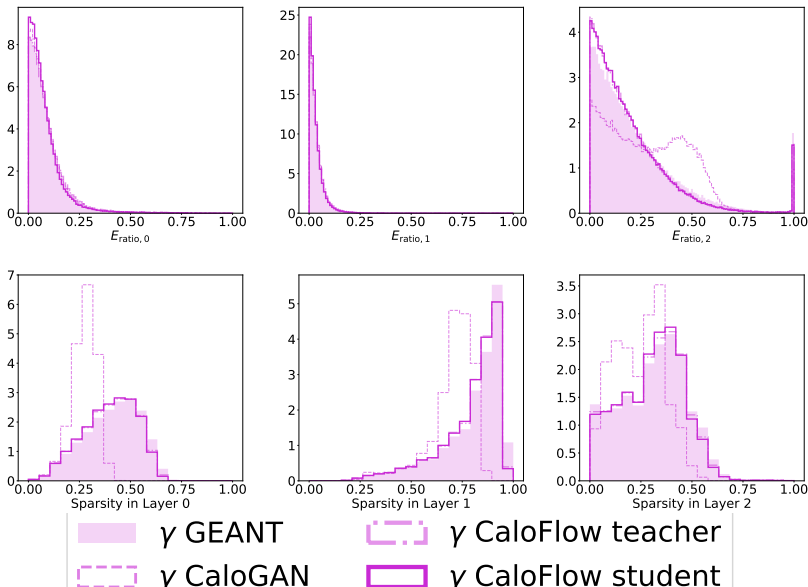
Flow I histograms: γ



Flow I+II histograms: γ



Flow II histograms: γ



Comparing Shower Averages: π^+

