Normalizing Flows for Calorimeter Simulation

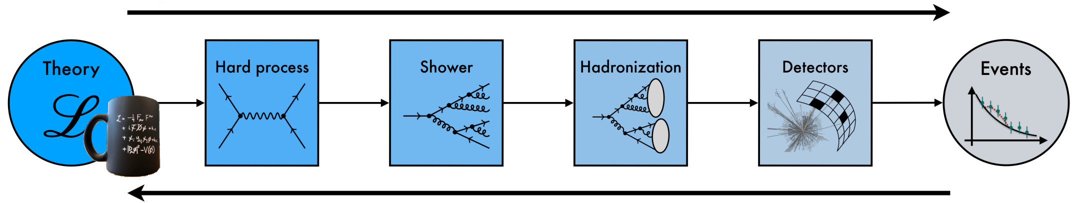
— COMETA WG2 Meeting —

Claudius Krause

Institute of High Energy Physics (HEPHY), Austrian Academy of Sciences (OeAW)

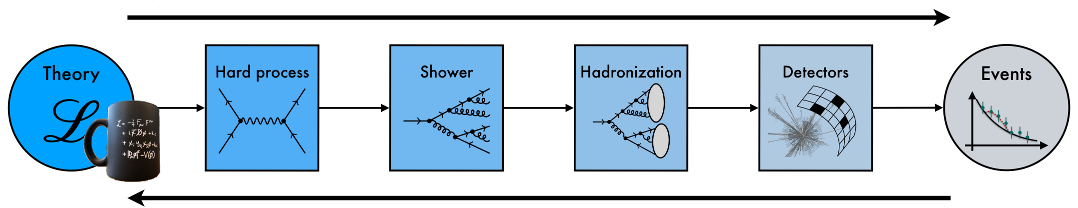
March 28, 2024

Simulation bridges Theory and Experiment.

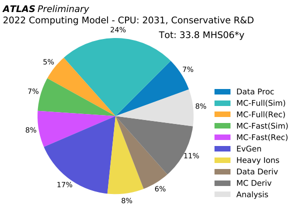


Machine Learning and LHC Event Generation, A. Butter et al. [2203.07460], R. Winterhalder

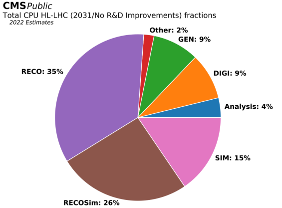
Simulation bridges Theory and Experiment.



Machine Learning and LHC Event Generation, A. Butter et al. [2203.07460], R. Winterhalder

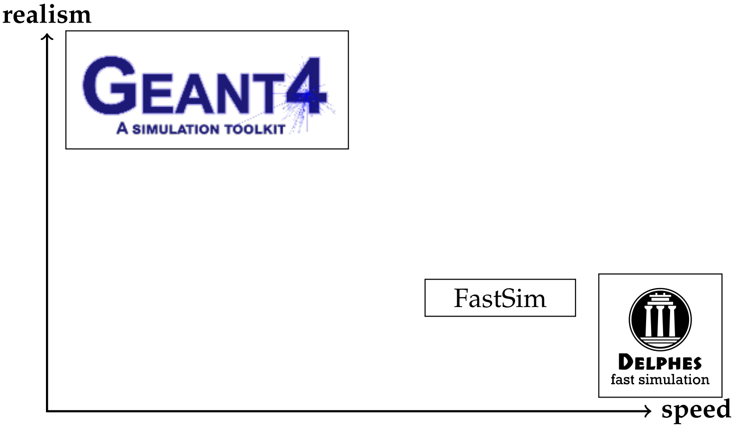


CERN-LHCC-2022-005

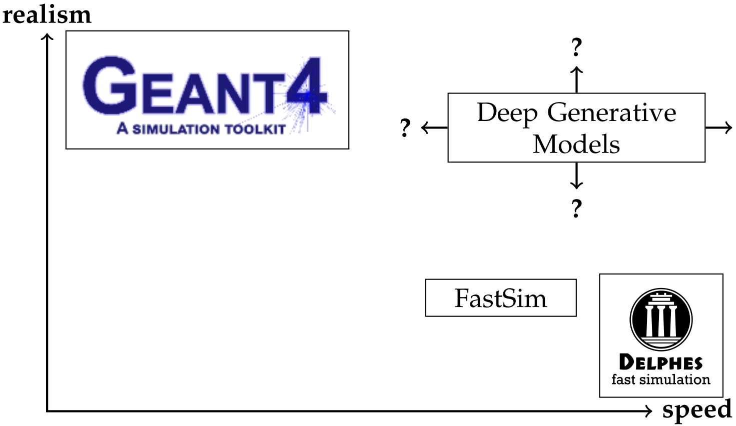


CMS-NOTE-2022-008

Detector simulation is computationally expensive.



Detector simulation is computationally expensive.



Evaluating Generative Models is a hard task.

We want to know if $p_{\text{generated}} = p_{\text{training data}}$ $(= p_{\text{truth}}?)$

- If we have access to $p(x)$, we can compute f-divergences.
  ↳ Example: KL-divergence $\int dx\, p(x) \log \frac{p(x)}{q(x)}$

- Alternatively, we could use Integral Probability Metrics.
  ↳ Example: Wasserstein distance

- In Computer Science, one uses the Frechét Inception Distance.

- In HEP, we usually look at histograms.

See also: Kansal et al. [arXiv:2211.10295]

A Classifier provides the “ultimate metric”.

According to the Neyman-Pearson Lemma we have:

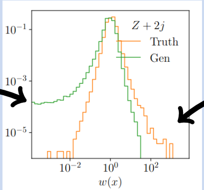
- The likelihood ratio is the most powerful test statistic to distinguish two samples.
- A powerful classifier trained to distinguish the samples should therefore learn (something monotonically related to) $w = \frac{p_{\text{data}}}{p_{\text{model}}}$.
- If this classifier is confused, we conclude $\Rightarrow$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

$\Rightarrow$ This captures the full phase space incl. correlations.

CK/D. Shih [2106.05285, PRD]

Failure modes of the model can now be seen in the w histogram:

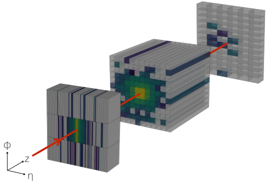
Data manifold over-populated by model: $\Rightarrow$ missmodeled feature

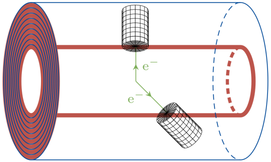


Data manifold not populated by model: $\Rightarrow$ missed feature

R. Das, CK, et al. [2305.16774, SciPost]

Part I: The CaloGAN Dataset

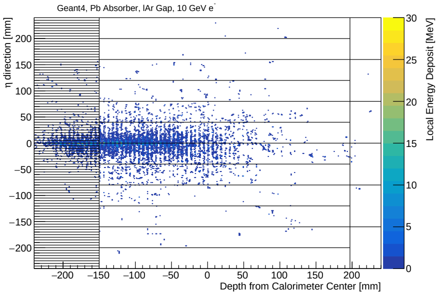


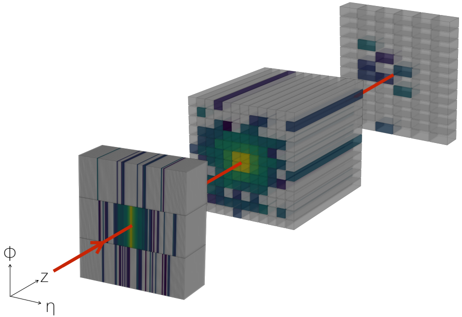


Part II: The CaloChallenge 2022

I: We use the same calorimeter geometry as CaloGAN

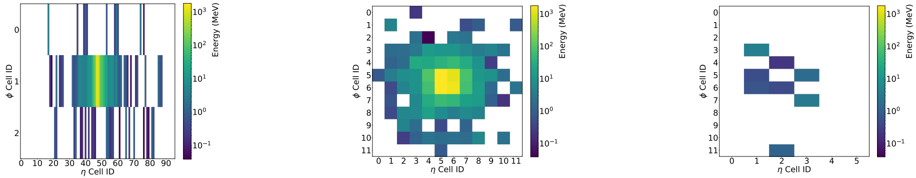
- We consider a toy calorimeter inspired by the ATLAS ECal: flat alternating layers of lead and LAr
- They form three instrumented layers of dimension 3×96, 12×12, and 12×6





I: 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
- $E_{\rm inc}$ 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]

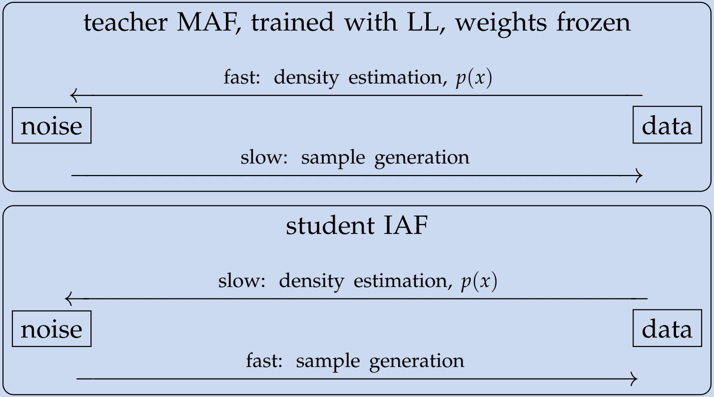
I: CaloFlow uses a 2-step approach to learn $p(\vec{\mathcal{I}}|E_{\text{inc}})$.

Flow I learns $p_1(E_0, E_1, E_2|E_{\text{inc}})$
$\Rightarrow$ is a Masked Autoregressive Flow, optimized using the log-likelihood.

Flow II learns $p_2(\hat{\vec{\mathcal{I}}}|E_0, E_1, E_2, E_{\text{inc}})$ of normalized showers

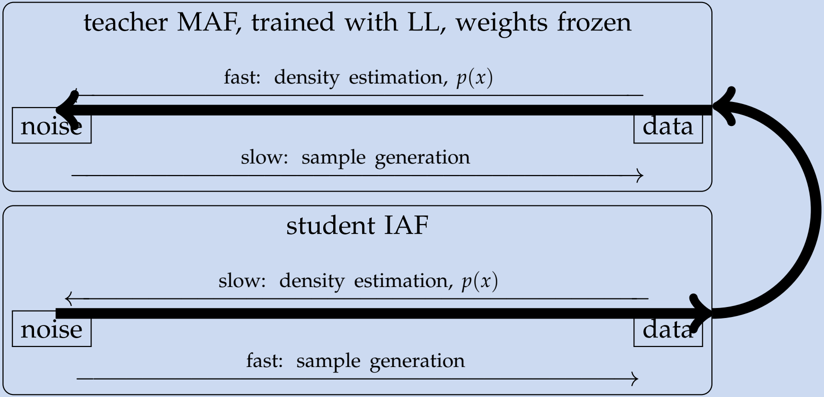
- in CaloFlow v1 (2106.05285 — called “teacher”):
  - Masked Autoregressive Flow trained with log-likelihood
  - $\Rightarrow$ Slow in sampling ($\approx 500\times$ slower than CaloGAN)
- in CaloFlow v2 (2110.11377 — called “student”):
  - Inverse Autoregressive Flow trained with Probability Density Distillation from teacher (log-likelihood prohibitive), i.e. matching IAF parameters to frozen MAF van den Oord et al.[1711.10433]
  - $\Rightarrow$ Fast in sampling ($\approx 500\times$ faster than CaloFlow v1)

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



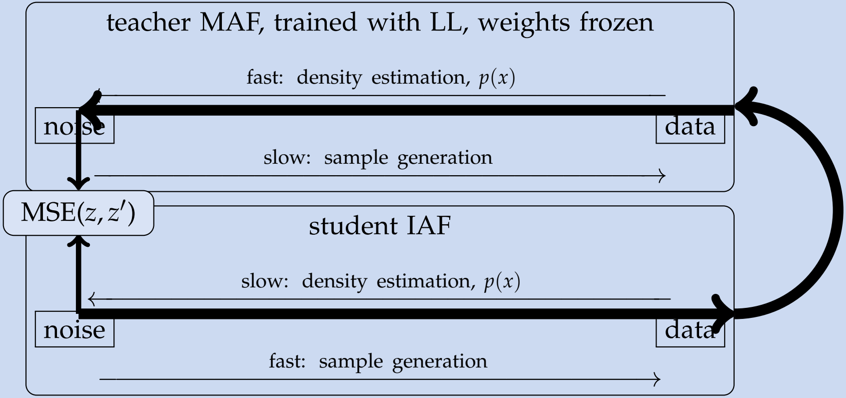
$$\text{Loss} = \text{MSE}(z, z') + \text{MSE}(x, x') + \text{MSE}(z_i, z_i') + \text{MSE}(x_i, x_i') + \text{MSE}(\theta_z, \theta_z') + \text{MSE}(\theta_x, \theta_x')$$

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



$$\text{Loss} = \text{MSE}(z, z') + \text{MSE}(x, x') + \text{MSE}(z_i, z'_i) + \text{MSE}(x_i, x'_i) + \text{MSE}(\theta_z, \theta'_z) + \text{MSE}(\theta_x, \theta'_x)$$

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

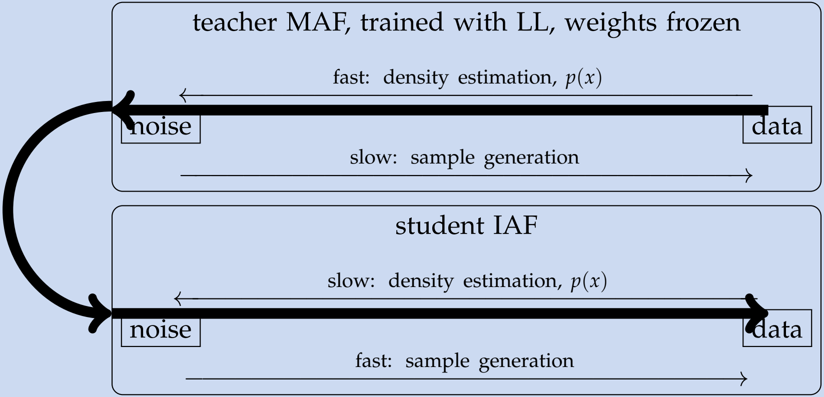


$$\text{Loss} = \text{MSE}(z,z') + \text{MSE}(x,x') + \text{MSE}(z_i,z_i') + \text{MSE}(x_i,x_i') + \text{MSE}(\theta_z,\theta_z') + \text{MSE}(\theta_x,\theta_x')$$

ÖAW AUSTRIAN ACADEMY OF SCIENCES

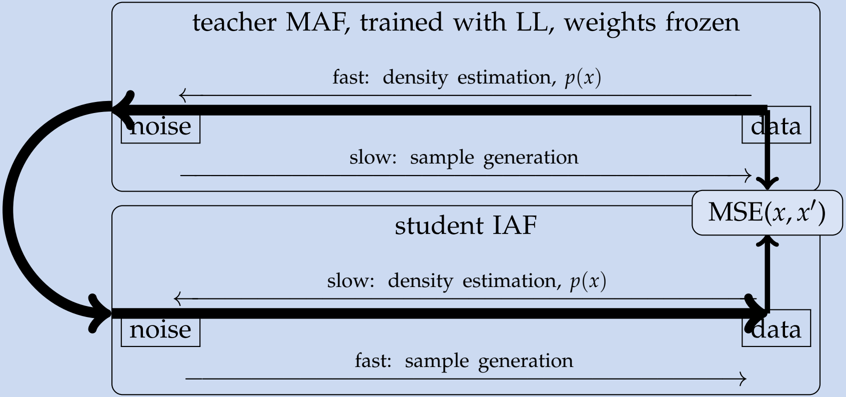
HEPHY INSTITUTE OF HIGH ENERGY PHYSICS

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



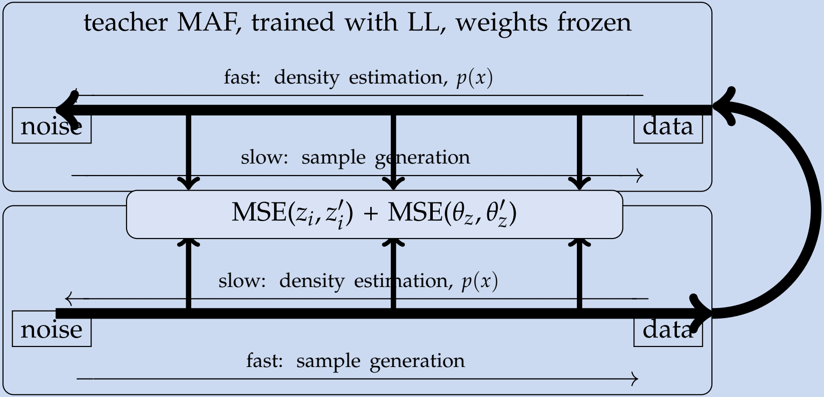
$$\text{Loss} = \text{MSE}(z, z') + \text{MSE}(x, x') + \text{MSE}(z_i, z_i') + \text{MSE}(x_i, x_i') + \text{MSE}(\theta_z, \theta_z') + \text{MSE}(\theta_x, \theta_x')$$

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



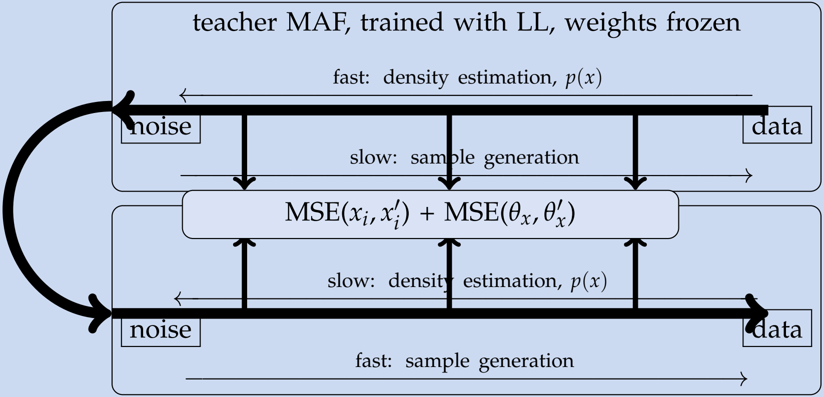
$$\text{Loss} = \text{MSE}(z, z') + \text{MSE}(x, x') + \text{MSE}(z_i, z_i') + \text{MSE}(x_i, x_i') + \text{MSE}(\theta_z, \theta_z') + \text{MSE}(\theta_x, \theta_x')$$

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



$$\text{Loss} = \mathrm{MSE}(z, z') + \mathrm{MSE}(x, x') + \mathrm{MSE}(z_i, z_i') + \mathrm{MSE}(x_i, x_i') + \mathrm{MSE}(\theta_z, \theta_z') + \mathrm{MSE}(\theta_x, \theta_x')$$

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



$$\text{Loss} = \mathrm{MSE}(z, z') + \mathrm{MSE}(x, x') + \mathrm{MSE}(z_i, z_i') + \mathrm{MSE}(x_i, x_i') + \mathrm{MSE}(\theta_z, \theta_z') + \mathrm{MSE}(\theta_x, \theta_x')$$

I: 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.

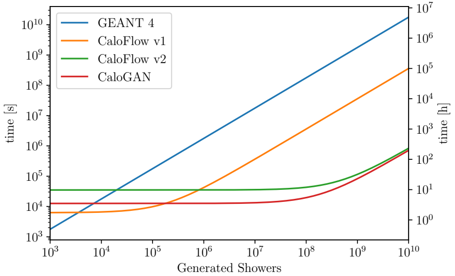
AUC		GEANT4 vs. CaloGAN	GEANT4 vs. (teacher) CaloFlow v1	GEANT4 vs. (student) CaloFlow v2
e^+	low-level	1.000(0)	0.870(2)	0.824(4)
	high-level	1.000(0)	0.795(1)	0.762(3)
γ	low-level	1.000(0)	0.796(2)	0.760(3)
	high-level	1.000(0)	0.727(2)	0.739(2)
π^+	low-level	1.000(0)	0.755(3)	0.807(1)
	high-level	1.000(0)	0.888(1)	0.893(2)

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

I: Sampling Speed: The Student beats the Teacher!

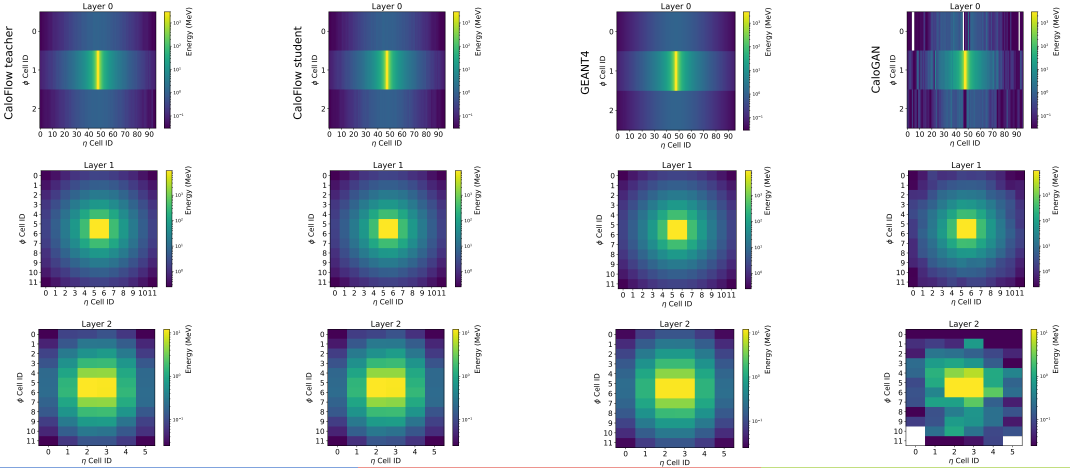
	CaloFlow* teacher	CaloFlow* student	CaloGAN*	Geant4†
training	22+82 min	+ 480 min	210 min	0 min
generation time per shower	36.2 ms	**0.08 ms**	**0.07 ms**	1772 ms

*: on our Titan V GPU, †: on the CPU of CaloGAN: Paganini, de Oliveira, Nachman [1712.10321, PRD]

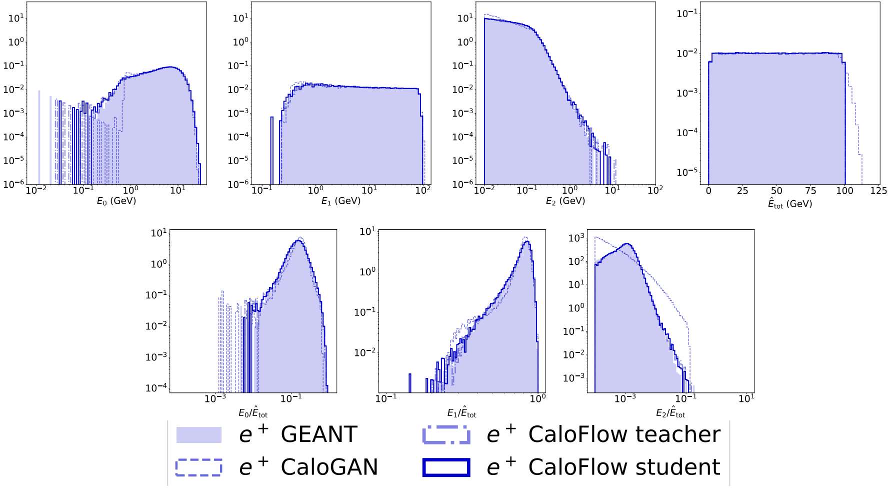


I: CaloFlow: Comparing Shower Averages: e^+

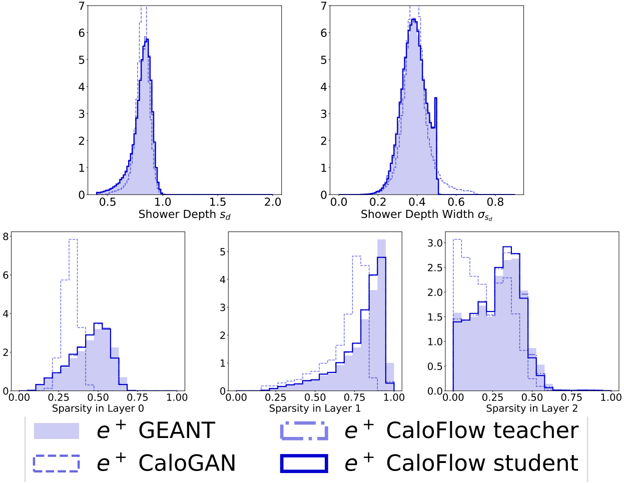
CaloFlow teacher | CaloFlow student | Geant4 | CaloGAN



I: CaloFlow: histograms: e^+



I: CaloFlow: histograms: e^+



I: We don't need a MAF / IAF: an INN is working well

	CaloFlow		INN	Geant4†
	teacher	student		
training	22+82 min *	+ 480 min *	$\mathcal{O}(400-500)^{\#}$ min	0 min
generation time per shower	36.2 ms*	**0.25 ms**‡	**0.21 ms**‡	1772 ms

*: on Rutgers Titan V GPU, ‡: in container (end-to-end) on Rutgers Titan V GPU, #: on the machines in Heidelberg

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

AUC	e^+	γ	π^+
low-level	0.53	0.53	0.66
high-level	0.66	0.67	0.79

Ernst/Favaro/CK/Plehn/Shih [2312.09290]

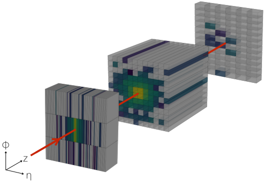
AUC ($\in [0.5, 1]$): Area Under the ROC Curve, smaller is better, i.e. more confused

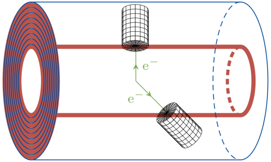
ÖAW AUSTRIAN ACADEMY OF SCIENCES

HEPHY INSTITUTE OF HIGH ENERGY PHYSICS

Normalizing Flows for Calorimeter Simulation

Part I: The CaloGAN Dataset





Part II: The CaloChallenge 2022

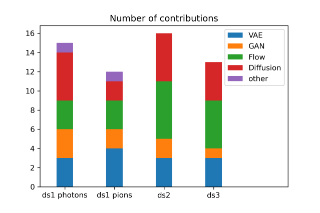
II: Going the next step: towards deployment in FastSimulation.

Have a rapidly evolving field: need a survey of current approaches on a common dataset!

$\Rightarrow$ Fast Calorimeter Challenge 2022 `https://calochallenge.github.io/homepage/`

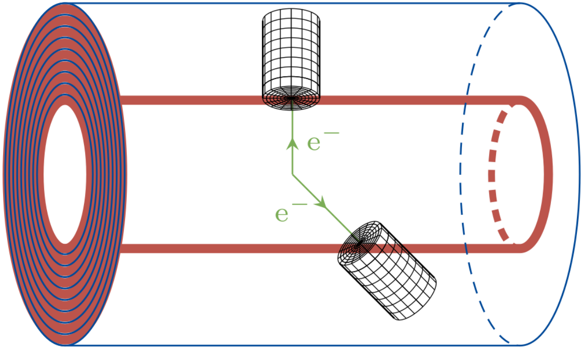
Michele Faucci Giannelli, Gregor Kasieczka, **CK**, Ben Nachman, Dalila Salamani, David Shih, and Anna Zaborowska

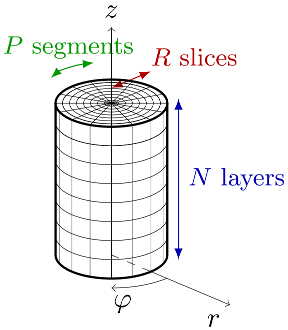
- Dataset 1: AtlFast3 trainig data (γ: 368, π: 533 voxels) [2109.02551, Comput.Softw.Big Sci.]
- Dataset 2: simulated detector (e^-: 6480 voxels)
- Dataset 3: simulated detector (e^-: 40500 voxels)



Submissions were presented at a workshop in Rome and at ML4Jets-22 / ML4Jets-23.

II: CaloChallenge datasets: Showers are centered.





Application to full detector: center coordinate system at shower.

https://g4fastsim.web.cern.ch/docs/ml_workflow/#dataset-description

II: Dataset 1 is small enough for CaloFlow and INN.

Dataset 1 — photons: 368 voxels in 5 layers

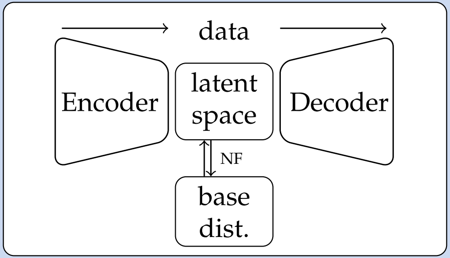
AUC	low-level	high-level
CaloFlow teacher [2210.14245]	0.701(3)	0.551(3)
CaloFlow student [2210.14245]	0.739(3)	0.556(3)
CaloINN [2312.09290]	0.626(4)	0.638(3)

Dataset 1 — pions: 533 voxels in 7 layers

AUC	low-level	high-level
CaloFlow teacher [2210.14245]	0.827(3)	0.692(2)
CaloFlow student [2210.14245]	0.866(2)	0.706(4)
CaloINN [2312.09290]	0.784(2)	0.732(2)

II: Targeting higher dimensionality via dimensionality reduction.

To combat the scaling, we reduce the dimensionality with a VAE.



Ernst/Favaro/CK/Plehn/Shih [2312.09290]

see also Cresswell et al. [2211.15380]

AUC	low-level	high-level
dataset 1 – photons	0.889(3)	0.966(1)
dataset 1 – pions	0.853(3)	0.921(2)
dataset 2	0.907(4)	0.999(0)
dataset 3	0.881(5)	1.000(0)

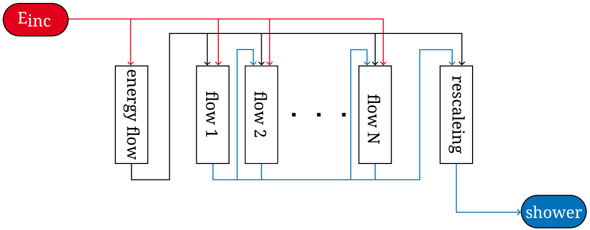
II: Targeting higher dimensions via autoregressive generation 1

L2LFlows: Split learning $p(\vec{\mathcal{I}}|E_{\text{inc}})$ into several steps, leveraging the detector geometry.

1. learns $p_0(E_1, E_2, E_3, \ldots, E_{45}|E_{\text{inc}})$ $\rightarrow$ how energy is distributed among layers.
2. learns $p_i(\mathcal{I}_i|\mathcal{I}_{1:i-1}, E_{1:45}, E_{\text{inc}})$ $\rightarrow$ how the shower in the ith layer looks like.

AUC	low-level	high-level
dataset 2	0.708(4)	0.737(2)
dataset 3	0.588(4)	0.733(6)

preliminary



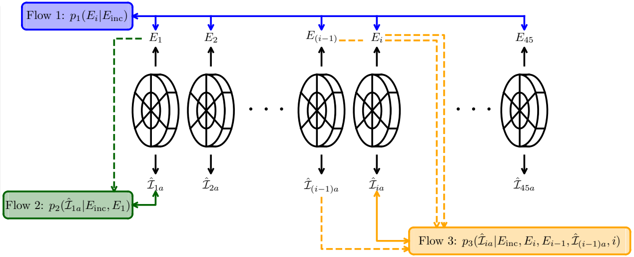
Buss/Diefenbacher/Gaede/Kasieczka/CK/Shih [in preparation];
on a different dataset: Diefenbacher/Eren/Gaede/Kasieczka/CK/Shekhzadeh/Shih [2302.11594, JINST]

II: Targeting higher dimensions via autoregressive generation 2

iCaloFlow: Split learning $p(\vec{\mathcal{I}}|E_{\text{inc}})$ into 3 steps, leveraging the detector geometry.

1. learns $p_1(E_1, E_2, E_3, \ldots, E_{45}|E_{\text{inc}})$ → how energy is distributed among layers.
2. learns $p_2(\mathcal{I}_1|E_1, E_{\text{inc}})$ → how the shower in the first layer looks like.
3. learns $p_3(\mathcal{I}_n|\mathcal{I}_{n-1}, n, E_n, E_{n-1}, E_{\text{inc}})$ → how the shower in layer n looks like, given layer $n-1$

AUC	low-level	high-level
dataset 2 teacher	0.797(5)	0.798(3)
dataset 2 student	0.840(3)	0.838(2)
dataset 3 teacher	0.911(3)	0.941(1)
dataset 3 student	0.910(8)	0.951(1)



M. Buckley, CK, I. Pang, D. Shih [2305.11934, PRD]

Normalizing Flows for Calorimeter Simulation

- Normalizing Flows have shown great performance as surrogate models in calorimeter simulation.
- Especially models based on coupling-layers show good sample quality at high generation speed.

Normalizing Flows for Calorimeter Simulation

- Normalizing Flows have shown great performance as surrogate models in calorimeter simulation.
- Especially models based on coupling-layers show good sample quality at high generation speed.

- For low-dimensional datasets ($\lesssim \mathcal{O}(500)$), flows are state-of-the-art in terms of quality & speed.
- For higher dimensional datasets ($\gtrsim \mathcal{O}(10^3)$) we need to investigate modifications / alternatives.
- The CaloChallenge allowed us to study several of them in detail. In our studies with VAEs and autoregressive models, we saw a trade-off in speed vs quality.
- The final evaluation of the CaloChallenge will quantify those statements.