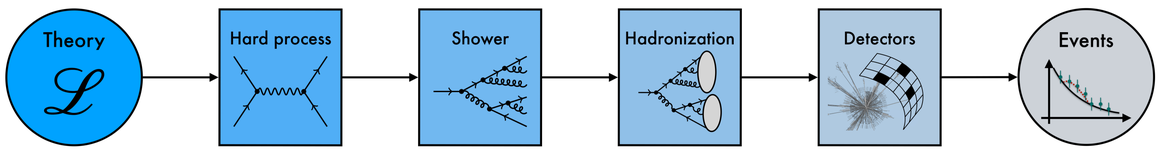
Flow-based generative models for particle calorimeter simulation

Claudius Krause
Claudius.Krause@oeaw.ac.at

HEPHY INSTITUTE OF HIGH ENERGY PHYSICS
ÖAW AUSTRIAN ACADEMY OF SCIENCES

Motivation

Particle physicists simulate collisions, allowing for a proper statistical interpretation of experimental data. The energy depositions of particles inside a calorimeter (as part of the detector simulation) are a crucial aspect of the simulation chain.



The state-of-the-art simulation code for detector simulation is GEANT4. It tracks each particle, as well as secondary particles that are produced in showers, through the entire detector volume. It is accurate, but slow. Deep generative models provide fast surrogates which could speed up detector simulation. We present different normalizing-flow-based approaches.

Normalizing Flows

Normalizing Flows learn a bijective mapping $f(x) = z$ between a high-dimensional prior distribution $\pi(z)$ and a complicated data distribution $p(x)$ via the change of coordinates formula:

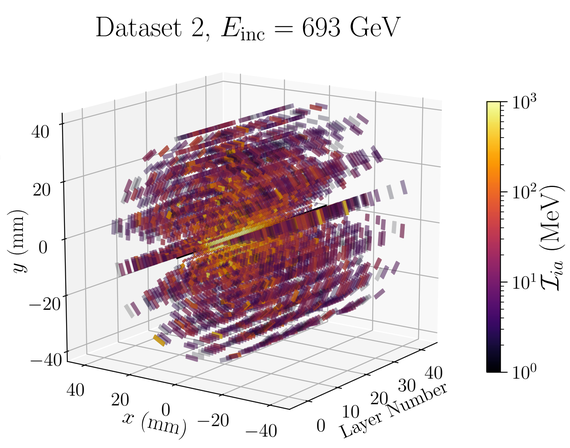
$$p(x) = \pi(f(x)) \left| \det \frac{\partial f(x)}{\partial x} \right| = \pi(z) \left| \det \frac{\partial f^{-1}(z)}{\partial z} \right|^{-1}$$

They can be trained by maximizing the log-likelihood $p(x)$ of the training data and act as generative model when run in the inverse direction $x = f^{-1}(z)$.

The Fast Calorimeter Simulation Challenge 2022

The fast Calorimeter Simulation Challenge 2022 [1] has provided 4 datasets of simulated calorimeter showers in increasing dimensionality.

1. dataset 1 — γ: 121k showers, 368 voxels
2. dataset 1 — π^+: 120.8k showers, 533 voxels
3. dataset 2 — e^-: 100k showers, 6,480 voxels
4. dataset 3 — e^-: 100k showers, 40,500 voxels



Evaluation Metrics

Shower quality is measured by training a neural classifier to distinguish generated samples from GEANT4 simulation. A lower AUC, i.e. a more confused classifier, indicates a generative model that better reproduces the distribution of GEANT4 showers. We investigate classifiers based on low-level features (calorimeter voxel) and high-level feature (observables like energies and shower shapes).

Generation speed is measured by generating a sample of the size of the training data with batch size 100 on a NVIDIA A100-SXM4-40GB.

1: Direct Approach: CaloFlow & CaloINN

All of these approaches split learning $p(\mathcal{I}|E_{\text{inc}})$ into 2 steps.

1. learns $p_1(E_1, E_2, \dots, E_n|E_{\text{inc}})$ $\rightarrow$ the energy split among layers
2. learns $p_2(\widehat{\mathcal{I}}|E_{1:n}, E_{\text{inc}})$ $\rightarrow$ the normalized shower in all layers.

CaloFlow [2, 3] is based on autoregressive flows, which have a fast and slow direction of evaluation. When using the Masked Autoregressive Flow (MAF), it can be trained by maximizing the log-likelihood but it is slow in sampling. When using the Inverse Autoregressive Flow (IAF), it is faster in sampling, but can only be trained via Probability Density Distillation from the MAF version.

CaloINN [4] is based on coupling-layers, which are equally fast in both directions, albeit need more bijector building blocks to capture all correlations.

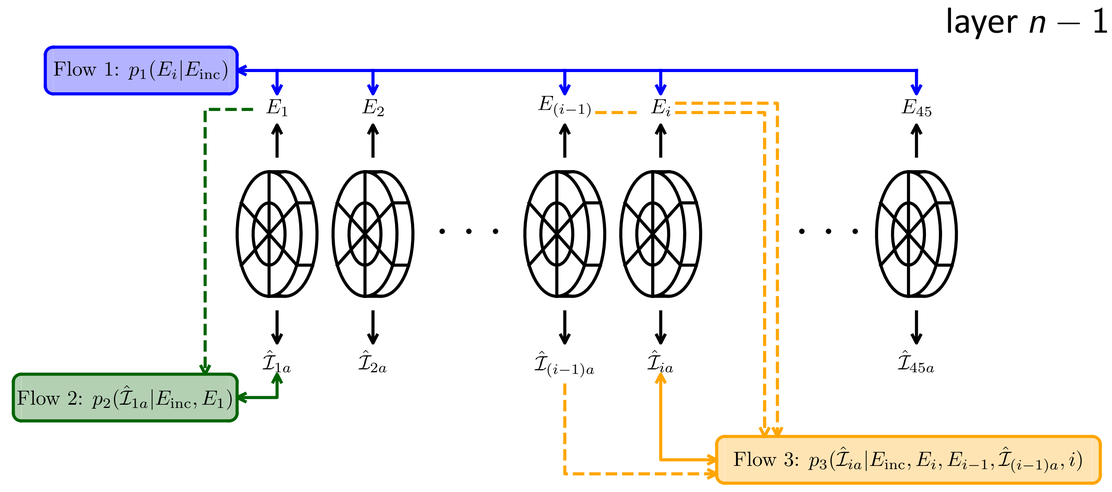
Dataset	Method	AUC ↓ low-level	AUC ↓ high-level	generation time per shower [ms] ↓
1: γ	GEANT4	0.499(2)	0.499(3)	$\mathcal{O}(10^4)$
	CaloFlow MAF [5]	0.733(3)	0.636(2)	45.5 ± 1.10
	CaloFlow IAF [5]	0.761(2)	0.667(4)	0.79 ± 0.01
	CaloINN [4]	0.626(4)	0.638(3)	0.51 ± 0.03
1: π^+	GEANT4	0.609(4)	0.558(2)	$\mathcal{O}(10^4)$
	CaloFlow MAF [5]	0.845(2)	0.797(2)	70.1 ± 1.00
	CaloFlow IAF [5]	0.884(2)	0.827(4)	1.00 ± 0.02
	CaloINN [4]	0.784(2)	0.732(2)	0.44 ± 0.01
2: e^-	GEANT4	0.500(2)	0.499(2)	$\mathcal{O}(10^5)$
	CaloINN [4]	0.743(2)	0.865(3)	1.18 ± 0.03

2: Autoregressive Decomposition: iCaloFlow [6]

Datasets 2 and 3 are too high-dimensional to be learned by a single normalizing flow.

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

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



Dataset	Method	AUC ↓ low-level	AUC ↓ high-level	generation time per shower [ms] ↓
2: e^-	GEANT4	0.500(2)	0.499(2)	$\mathcal{O}(10^5)$
	iCaloFlow MAF	0.763(4)	0.837(5)	829.4 ± 16.7
	iCaloFlow IAF	0.819(4)	0.886(3)	13.2 ± 0.5
3: e^-	GEANT4	0.498(2)	0.500(3)	$\mathcal{O}(10^5)$
	iCaloFlow MAF	0.911(3)	0.962(1)	5596.2 ± 56.0
	iCaloFlow IAF	0.891(3)	0.971(1)	16.7 ± 0.5

References

[1] M. Faucci Giannelli, G. Kasieczka, C. Krause, B. Nachman, D. Salamani, D. Shih et al., "Fast calorimeter simulation challenge 2022." https://github.com/CaloChallenge/homepage, 2022.

[2] C. Krause and D. Shih, *Fast and accurate simulations of calorimeter showers with normalizing flows*, *Phys. Rev. D* **107** (2023) 113003 [2106.05285].

[3] C. Krause and D. Shih, *Accelerating accurate simulations of calorimeter showers with normalizing flows and probability density distillation*, *Phys. Rev. D* **107** (2023) 113004 [2110.11377].

[4] F. Ernst, L. Favaro, C. Krause, T. Plehn and D. Shih, *Normalizing Flows for High-Dimensional Detector Simulations*, 2312.09290.

[5] C. Krause, I. Pang and D. Shih, *CaloFlow for CaloChallenge Dataset 1*, 2210.14245.

[6] M.R. Buckley, C. Krause, I. Pang and D. Shih, *Inductive simulation of calorimeter showers with normalizing flows*, *Phys. Rev. D* **109** (2024) 033006 [2305.11934].