CaloDREAM: Detector Response Emulation via Attentive Flow Matching

Luigi Favaro
in collaboration with: Ayodele Ore, Sofia Palacios Schweitzer, and Tilman Plehn
based on arXiv:2405.09629

ML4Jets 2024
Paris - 05.11.2024

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

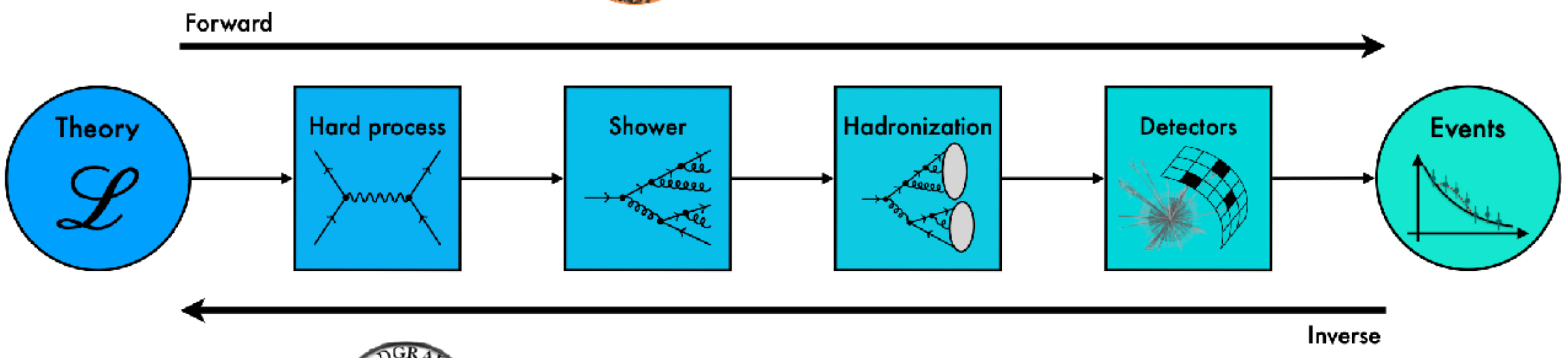
Collaborative Research Center TRR 257
Particle Physics Phenomenology after the Higgs Discovery

UCLouvain

DFG Deutsche Forschungsgemeinschaft
German Research Foundation

Simulation Chain

Pythia/Sherpa/Herwig



Madgraph

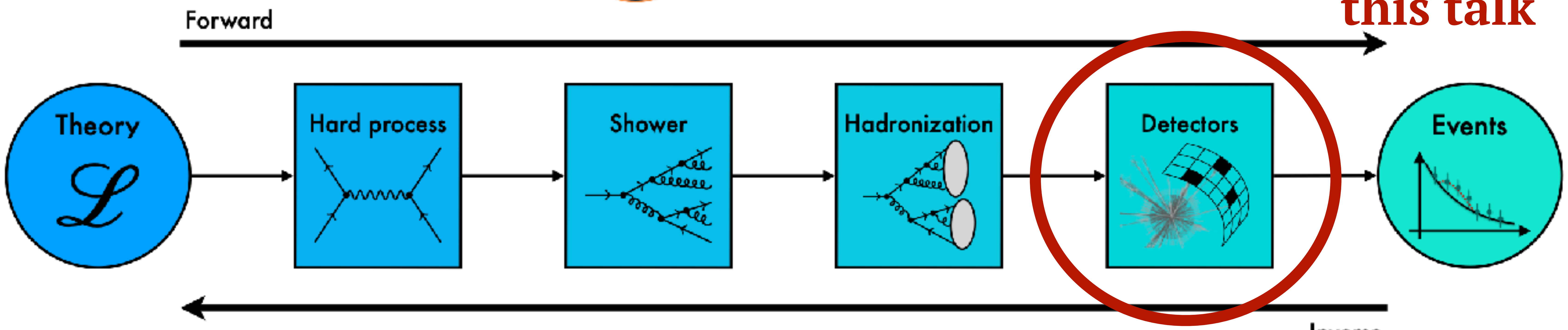
Geant4/Delphes

- First-principled simulations, from QFT to events

Simulation Chain

Pythia/Sherpa/Herwig

this talk



Madgraph

Geant4/Delphes

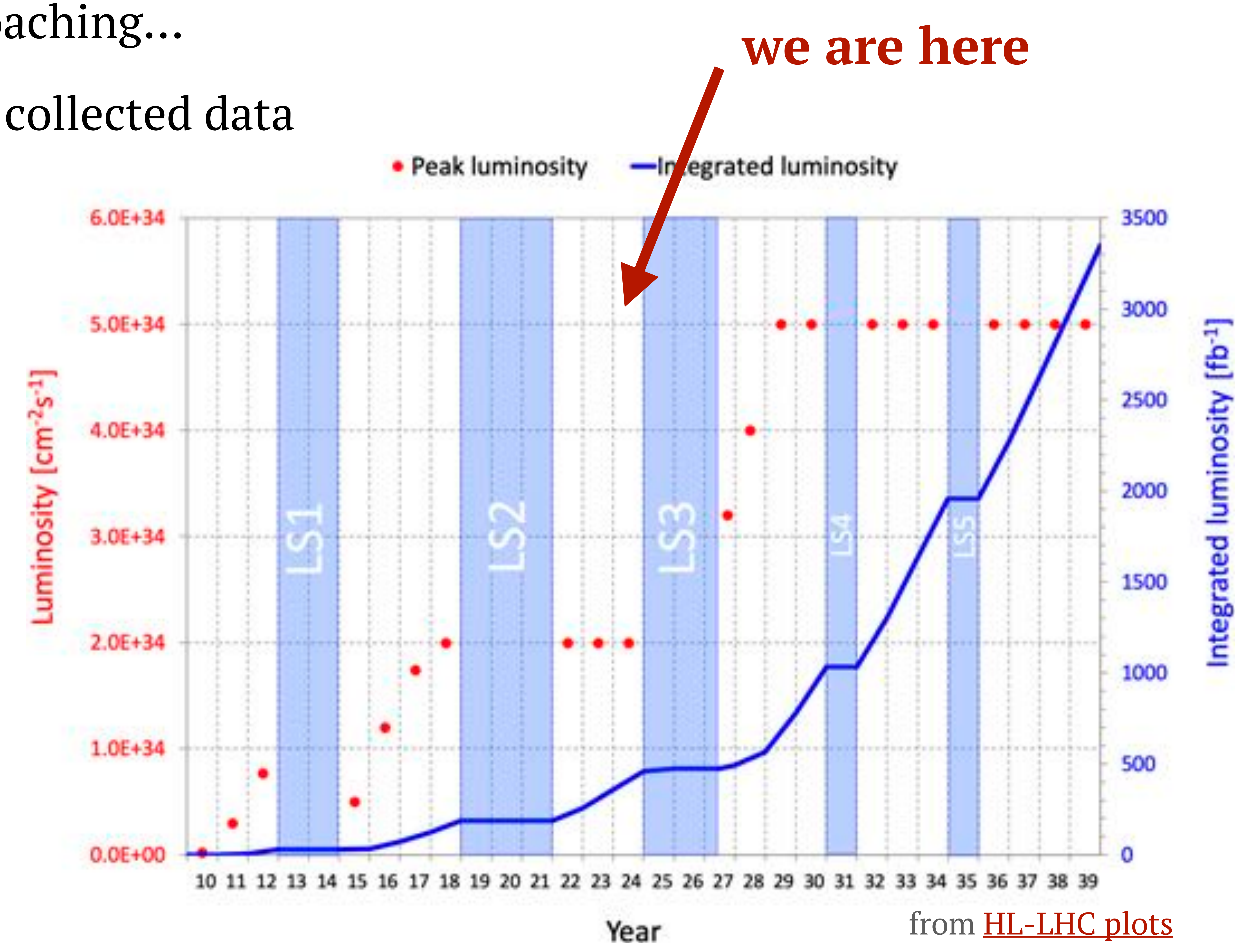
- First-principled simulations, from QFT to events

LHC future plan

- The high-luminosity data taking phase is approaching…
- Simulations will have to match the statistics of collected data

Need for fast generators…

… which are still (more) accurate and precise



from HL-LHC plots

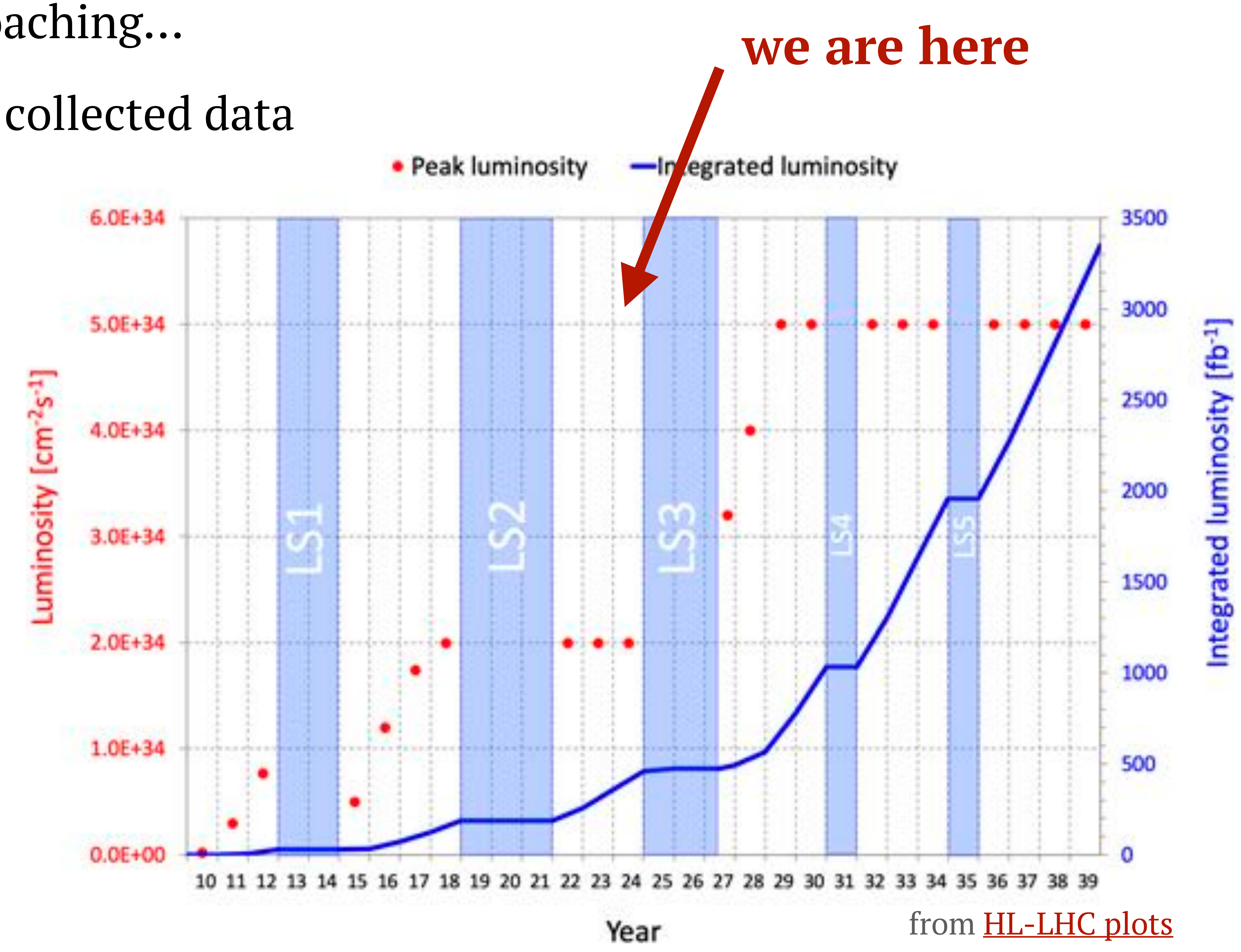
LHC future plan

- The high-luminosity data taking phase is approaching…
- Simulations will have to match the statistics of collected data

Need for fast generators…

… which are still (more) accurate and precise

find new physics!
(or rather understand LHC data)



from HL-LHC plots

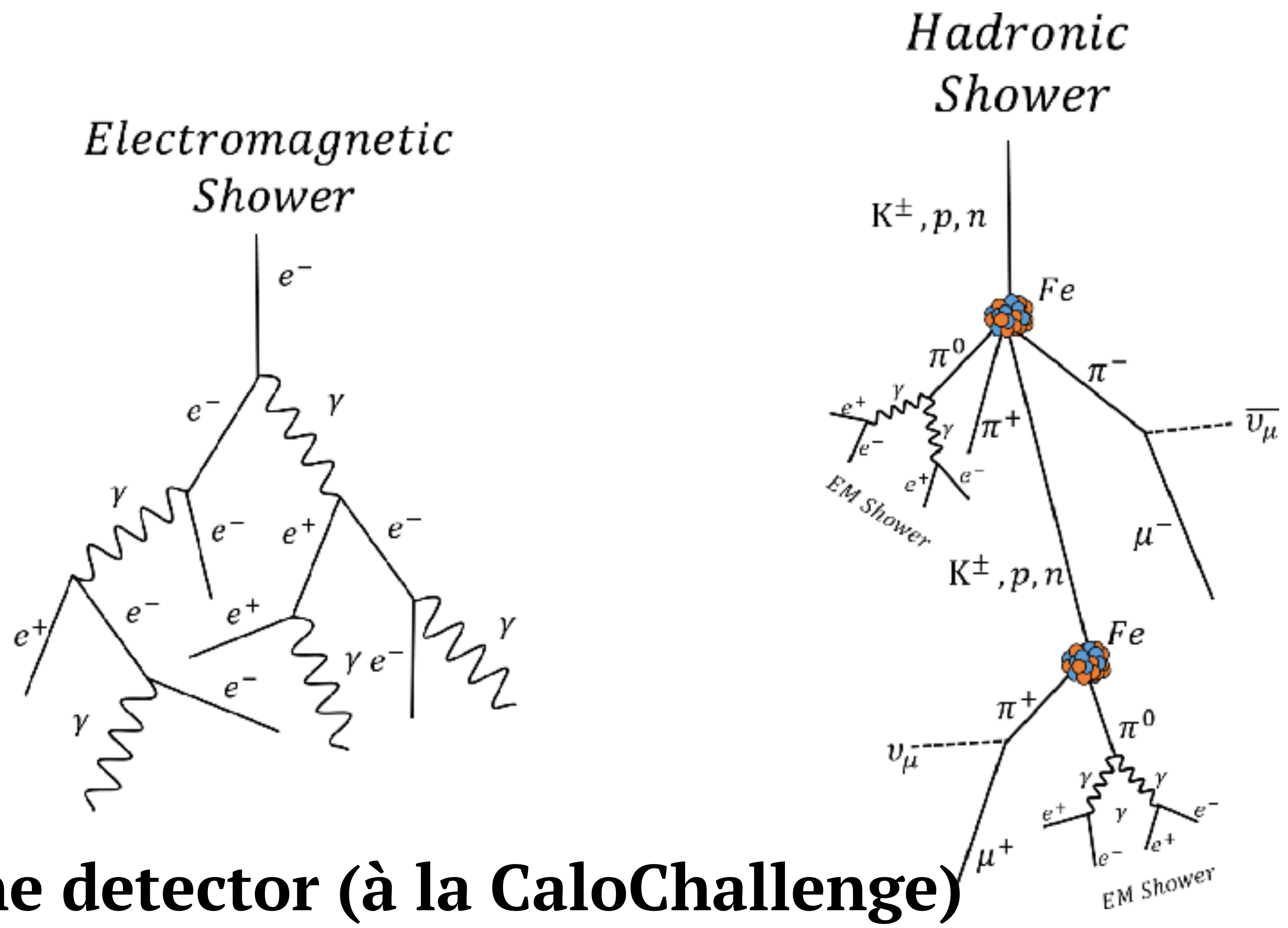
What are we simulating?

The leading speed bottleneck is the simulation of calorimeter showers

from here

Incident particle drastically changes the shower:

- $\gamma/e^{+/-}$: electromagnetic showers
  $\longrightarrow$ mostly Bremsstrahlung and pair-production
- hadrons: hadronic showers
  $\longrightarrow$ complex, non-perturbative phenomenology

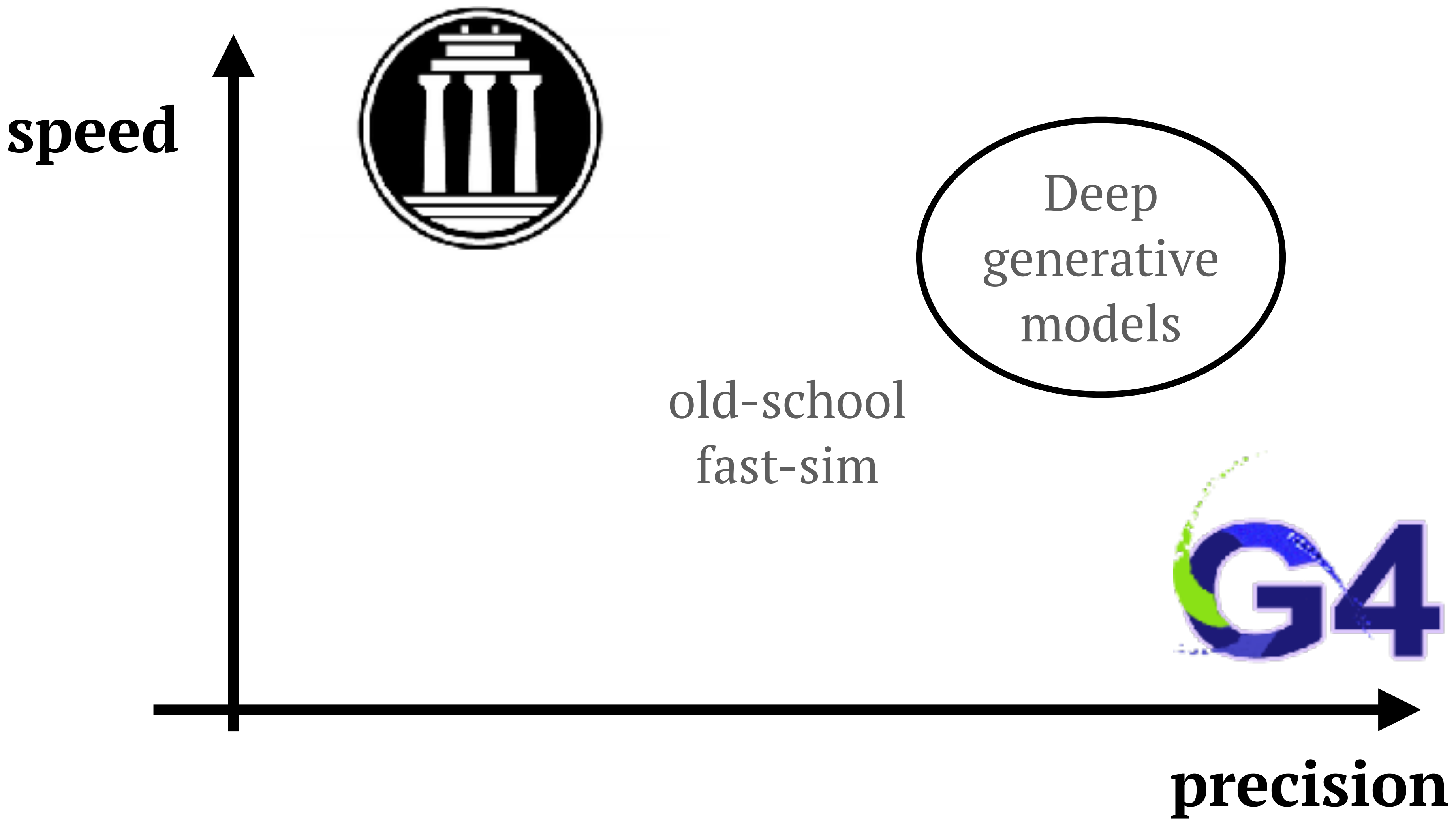


Calorimeter shower represented by the energy deposition in the detector (à la CaloChallenge)

Generative networks

Modern generative networks:

- Complex architectures but still fitting functions
- provided data, approximate Geant4
- speed and precision are key
- tradeoff



Conditional Flow Matching

Promote the discrete transformation to a continuous one:

$$\frac{dx(t)}{dt} = v(x(t), t) \quad \text{with} \quad x \in \mathbb{R}^d \qquad\qquad \frac{\partial p(x,t)}{\partial t} + \nabla_x \left[p(x,t) v(x,t) \right] = 0 \; .$$

We want to impose the boundary conditions for $p(x, t)$:

$$p(x,t) \to \begin{cases} \mathcal{N}(x; 0,1) & t \to 1 \\ p_{data}(x) & t \to 0 \; . \end{cases}$$

Need to define the training trajectories

$\longrightarrow$ linear, simplest choice

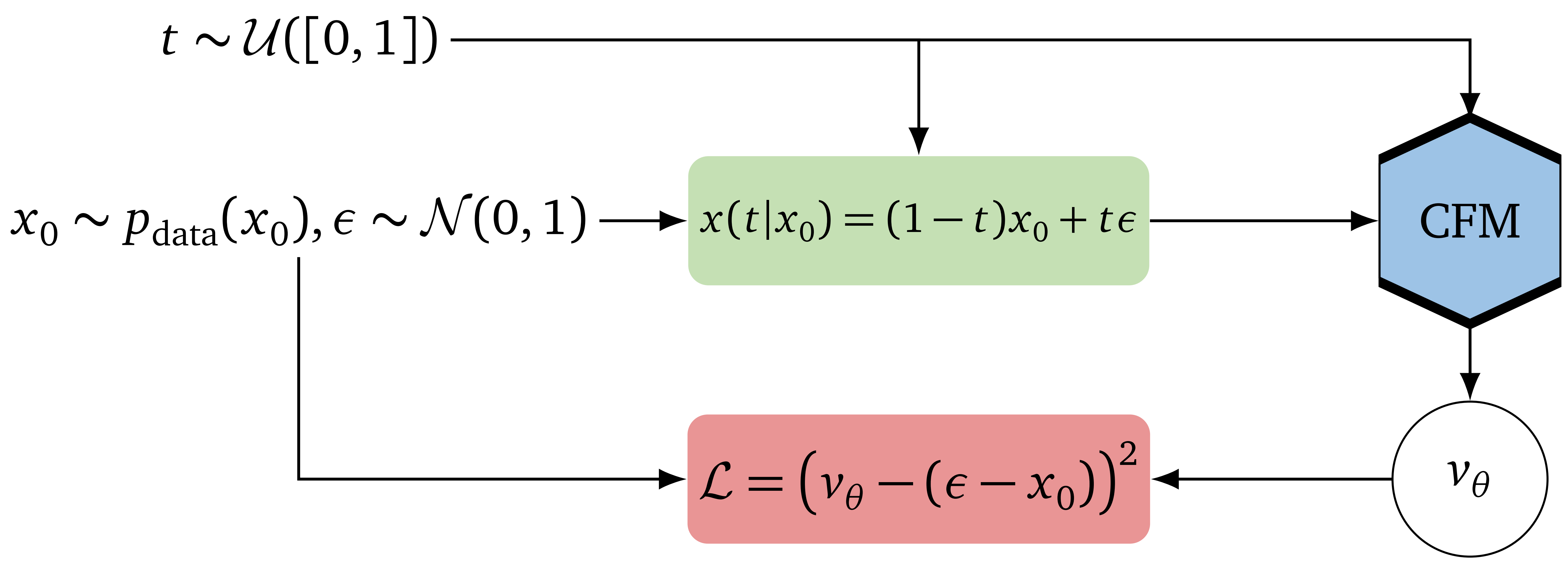
$$x(t \,|\, x_0) = (1-t)x_0 + t\epsilon \qquad \epsilon \sim \mathcal{N}(0,1)$$

Learn this velocity field with a NN:

$$\mathcal{L} = || v(x,t) - v_\phi(x,t) ||_{L_2}$$

Conditional Flow Matching

Jet diffusion versus JetGPT



$$\mathscr{L}_{\text{CFM}} = \left\langle \left[v_\phi((1-t)x_0 + t\epsilon, t) - (\epsilon - x_0) \right]^2 \right\rangle_{U(0,1), \mathcal{N}, p_{data}} .$$

Sampling $\longrightarrow$ solve the differential equation numerically: $\quad x(t=0) = x(t=1) - \int_0^1 v_\phi(x,t)dt$

Preprocessing

Preprocessing

normalized showers

$$u_0 = \frac{\sum_i E_i}{f E_{inc}} \qquad \text{and} \qquad u_i = \frac{E_i}{\sum_{j \geq i} E_j}\ ,$$

logit

$$x_\alpha = (1 - 2\alpha)x + \alpha \in [\alpha, 1 - \alpha] \qquad \text{with} \quad \alpha = 10^{-6}$$

$$x' = \log \frac{x_\alpha}{1 - x_\alpha}\ .$$

Factorise the problem into:

- learn the energy distribution, $p(u \mid E_{inc})$
- learn the normalised voxels $p(x \mid u, E_{inc})$

Preprocessing

normalized showers

$$u_0 = \frac{\sum_i E_i}{f E_{inc}} \qquad \text{and} \qquad u_i = \frac{E_i}{\sum_{j \geq i} E_j} ,$$

logit

$$x_\alpha = (1 - 2\alpha)x + \alpha \in [\alpha, 1 - \alpha] \qquad \text{with} \qquad \alpha = 10^{-6}$$

$$x' = \log \frac{x_\alpha}{1 - x_\alpha} .$$

Factorise the problem into:

- learn the energy distribution, $p(u \,|\, E_{inc})$
- learn the normalised voxels $p(x \,|\, u, E_{inc})$

SciPost Physics — Submission

arXiv:2405.09629

CaloDREAM —
Detector Response Emulation via Attentive flow Matching

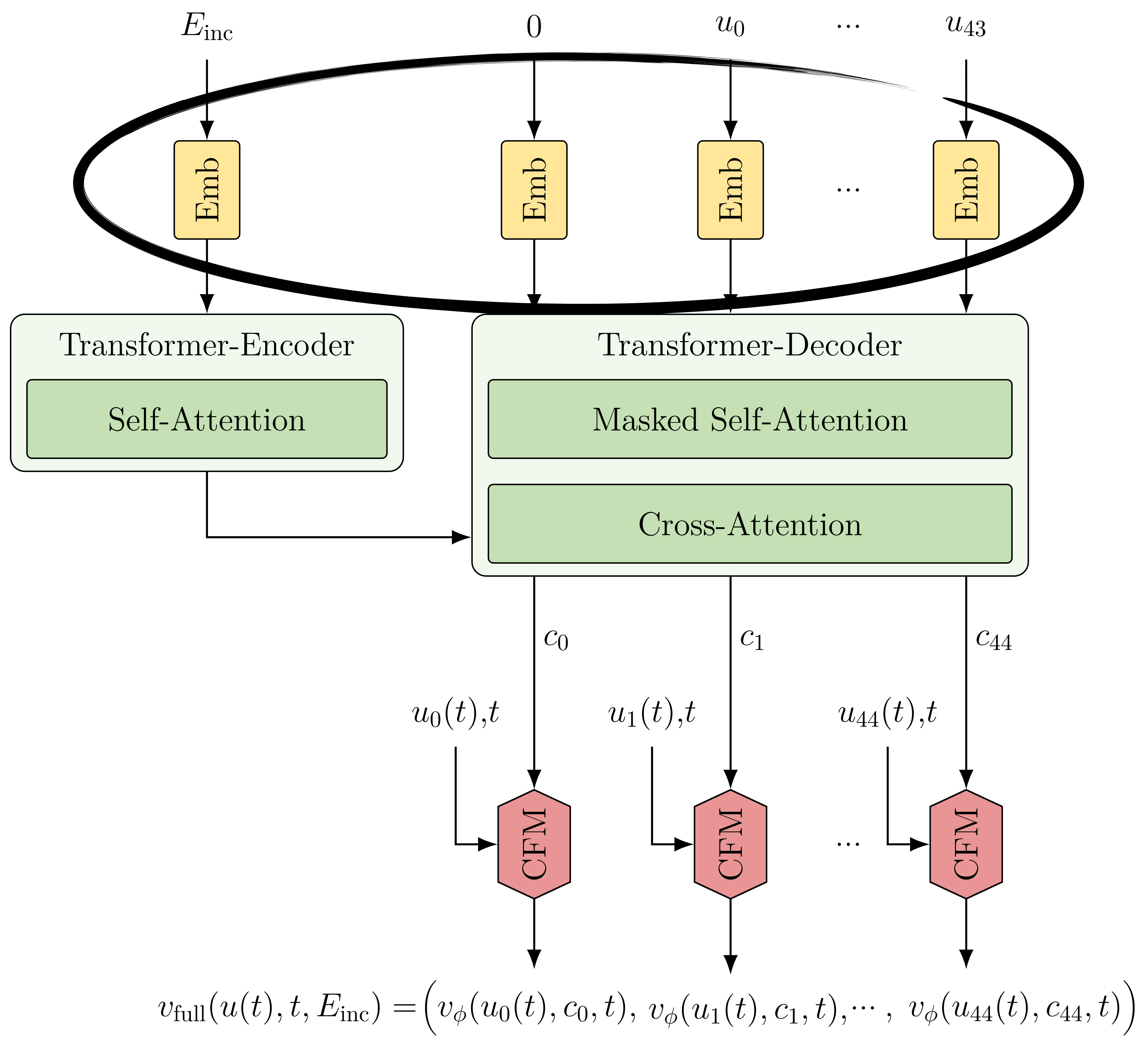
Luigi Favaro, Ayodele Ore, Sofia Palacios Schweitzer, and Tilman Plehn

Institut für Theoretische Physik, Universität Heidelberg, Germany

May 20, 2024

CaloDREAM: TraCFM

Energy network

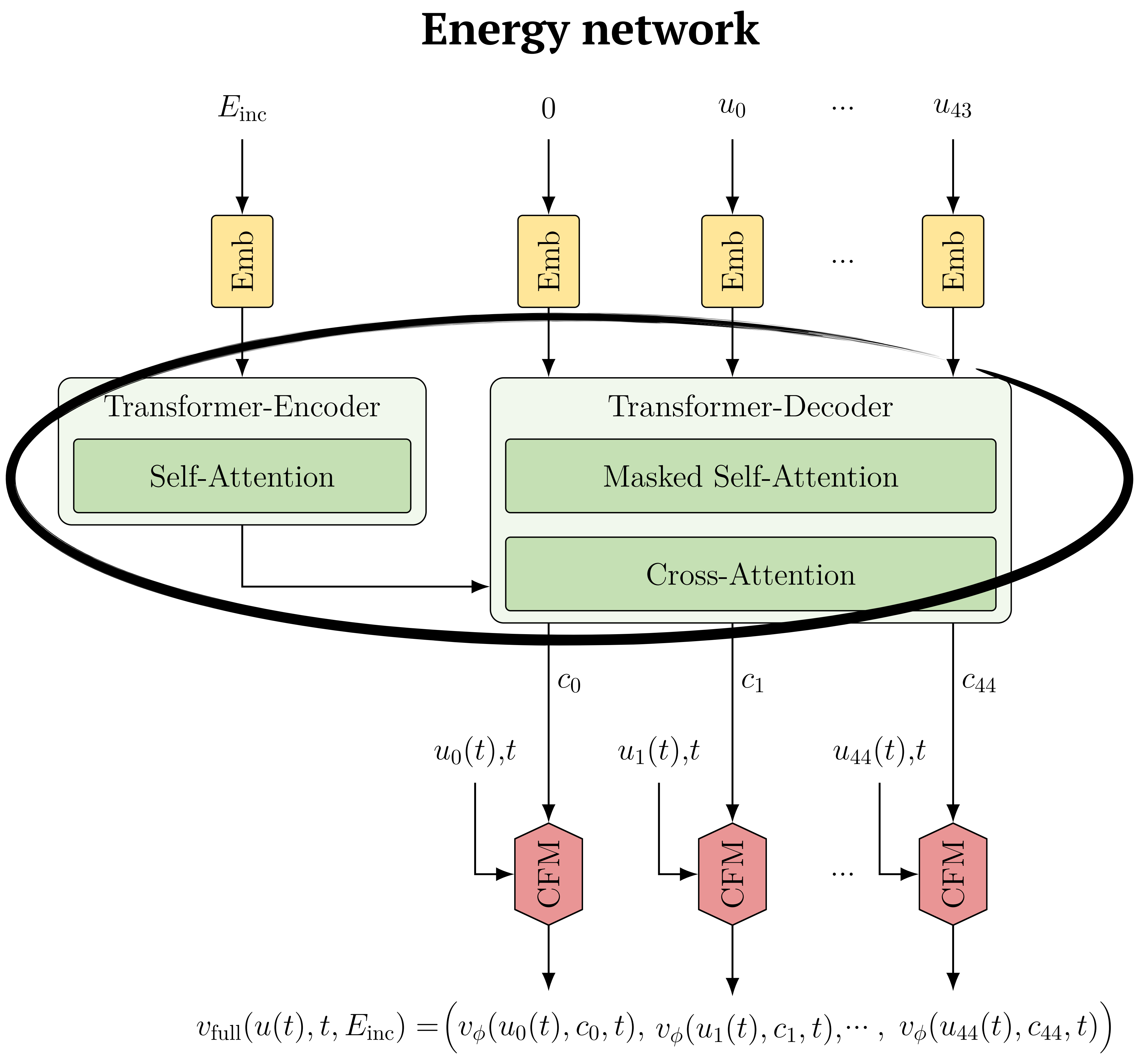


Autoregressive transformer:

- Embed each condition separately;

Networks

CaloDREAM: TraCFM



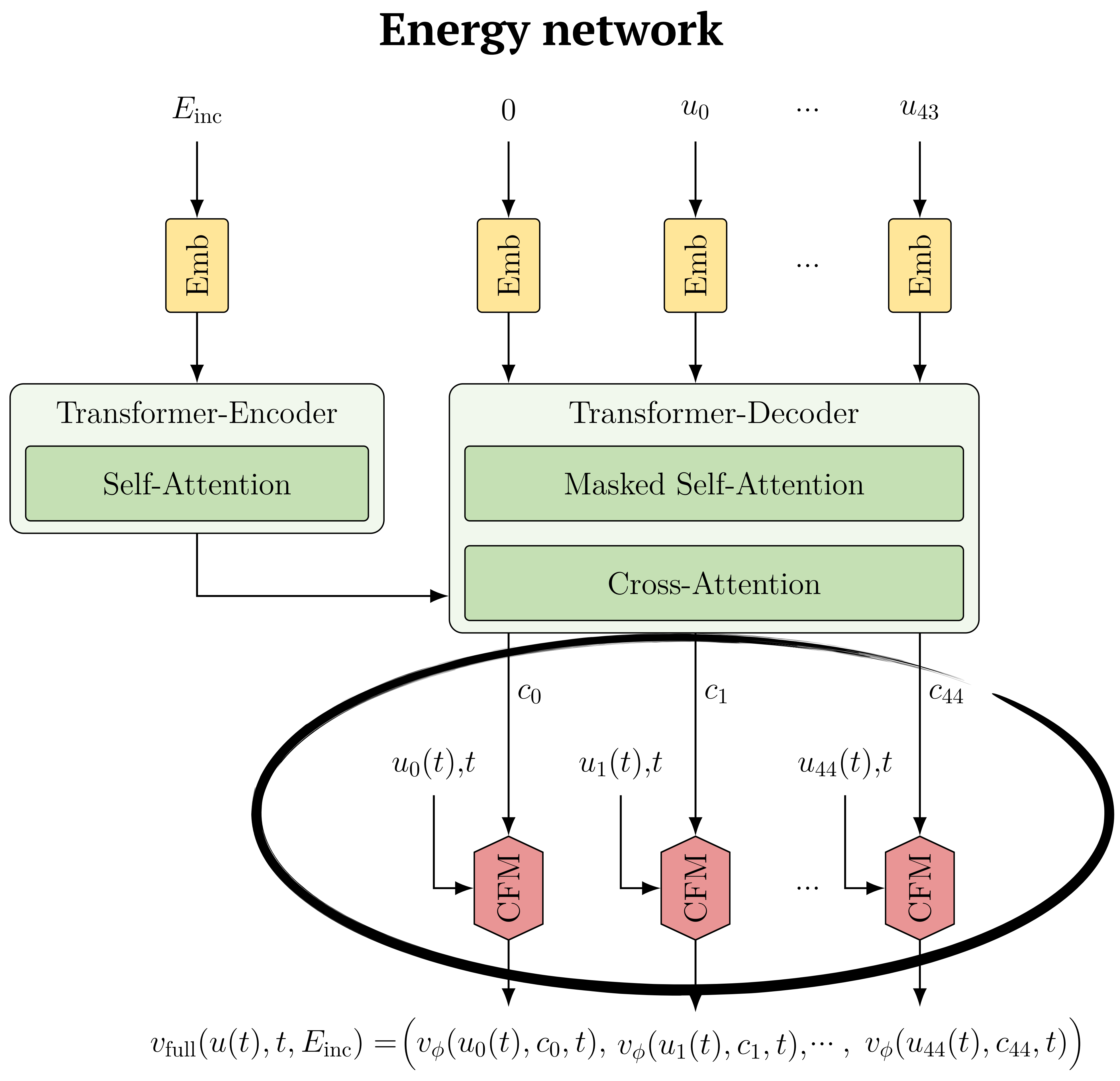
Autoregressive transformer:

- Embed each condition separately;
- Encode energy conditions ⟶ transformer backbone;
- Masked attention over previous layers:

$$c_i = c_i(u_0, \ldots, u_{i-1}, E_{inc});$$

Networks

CaloDREAM: TraCFM



Autoregressive transformer:

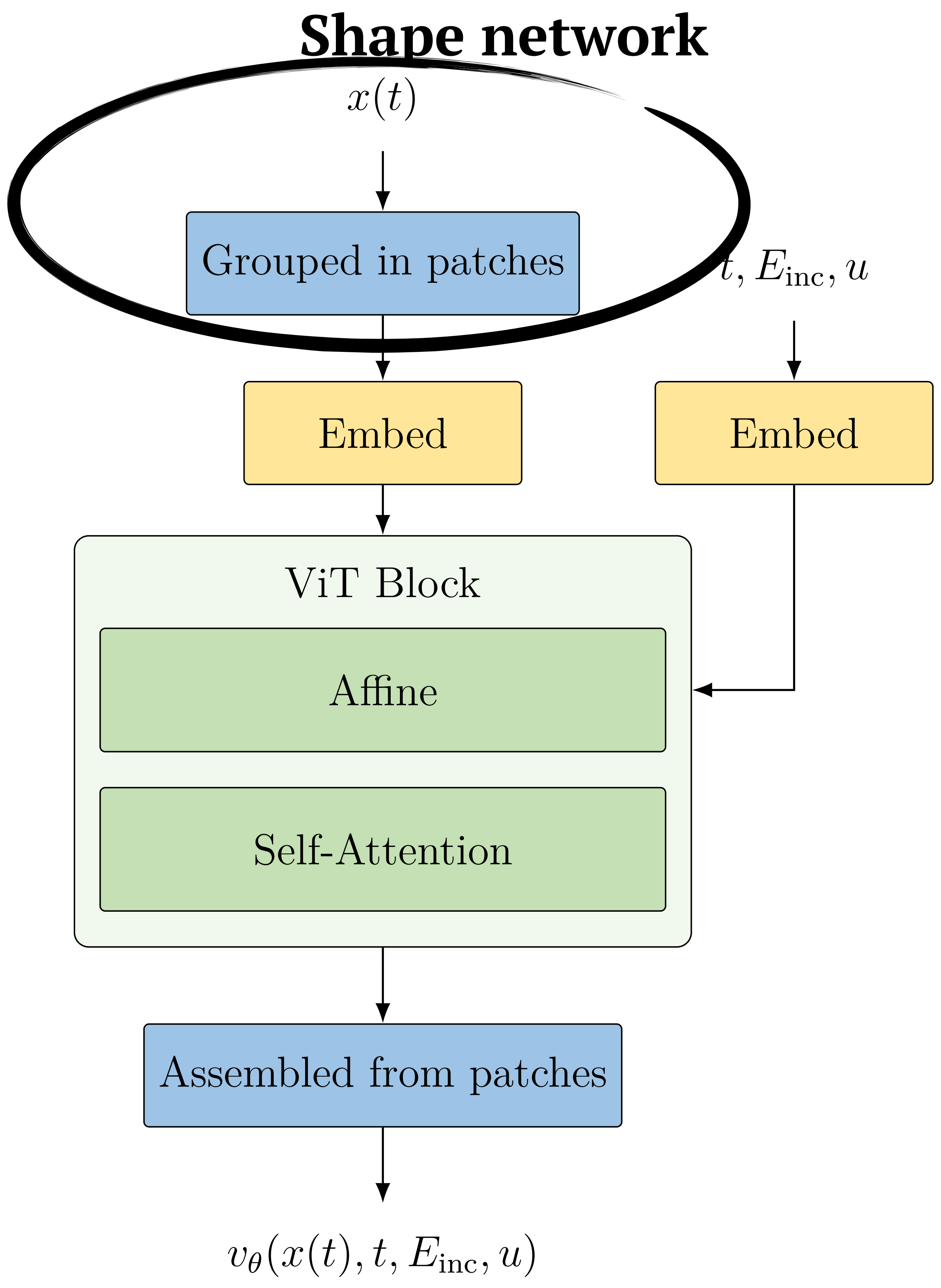
- Embed each condition separately;
- Encode energy conditions $\longrightarrow$ transformer backbone;
- Masked attention over previous layers:

$$c_i = c_i(u_0, \ldots, u_{i-1}, E_{inc});$$

- Conditions, energy, and time predict v_ϕ.

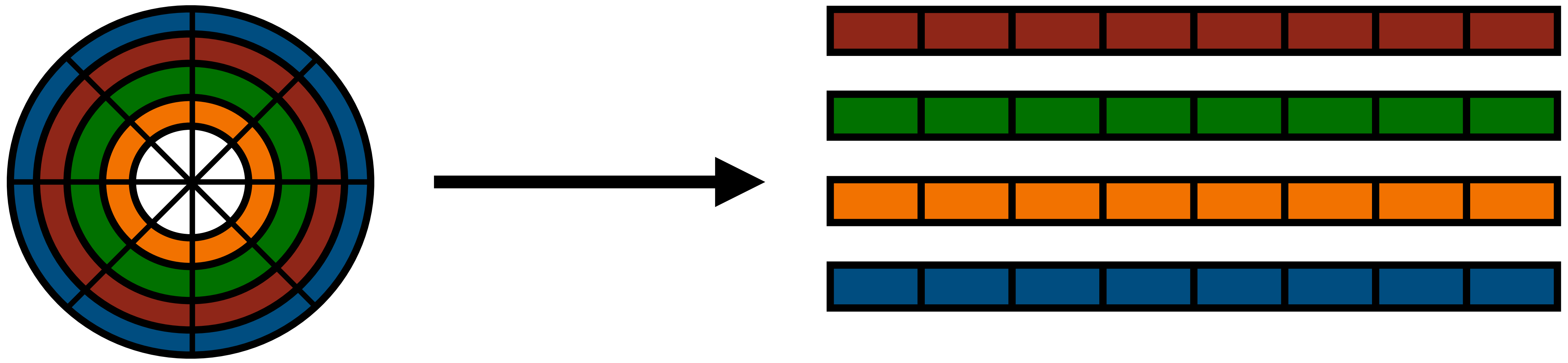
Networks

CaloDREAM: ViT



Vision transformer:

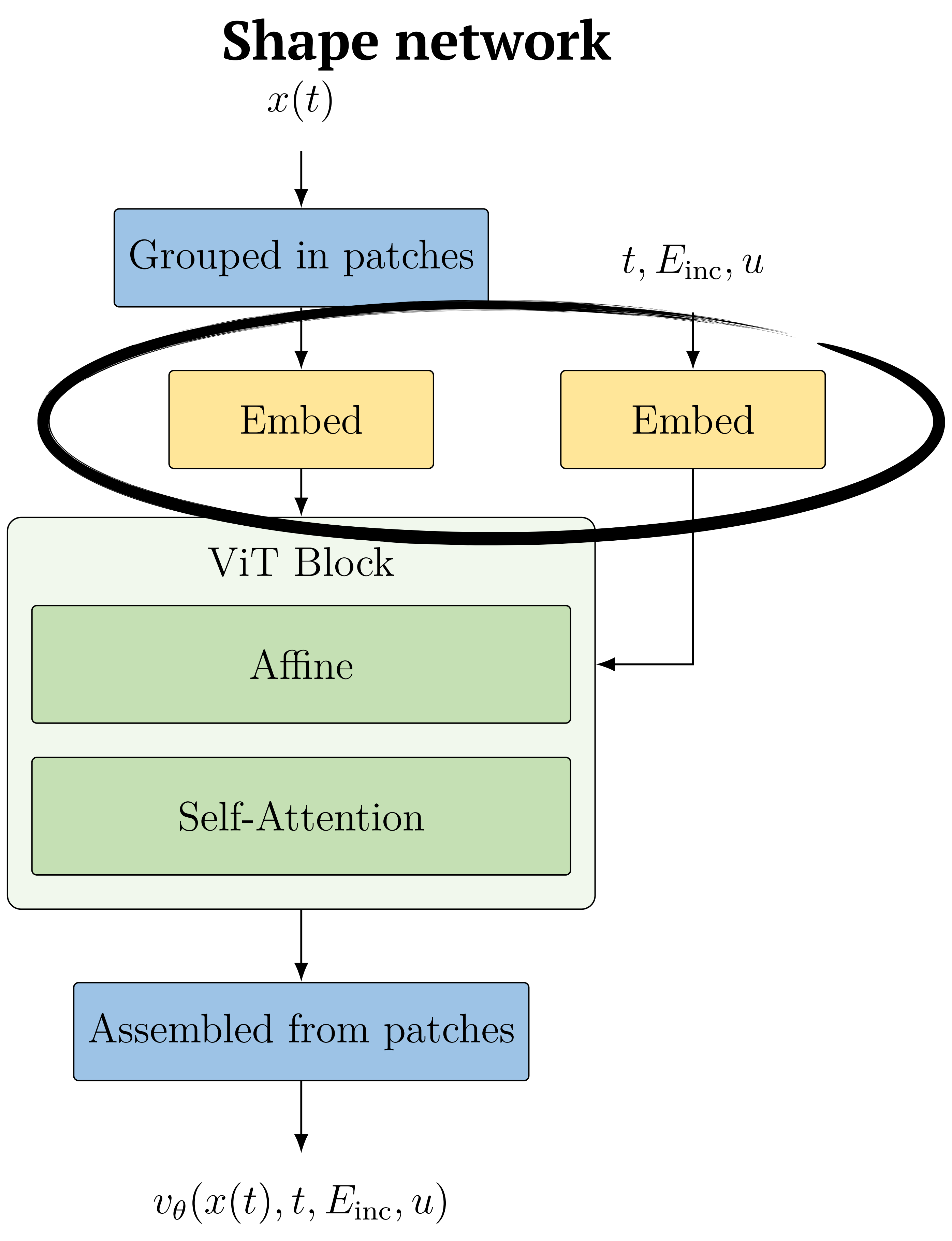
- Split detector into patches;



e.g. for dataset-2: $(r, \alpha, z) = (1,16,3)$

Networks

CaloDREAM: ViT

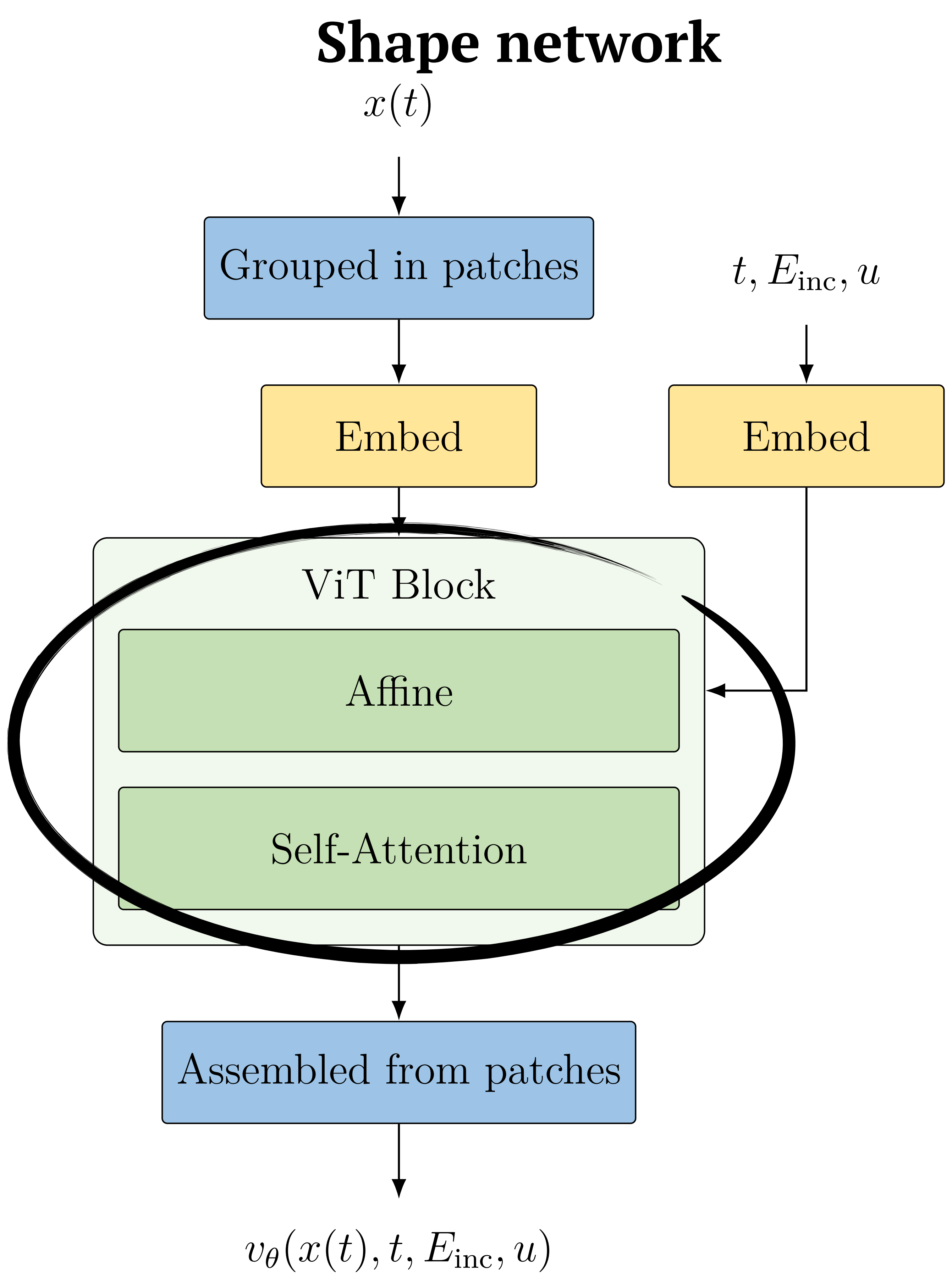


Vision transformer:

- Split detector into patches;
- Embed patches and conditions;

Networks

CaloDREAM: ViT



Vision transformer:

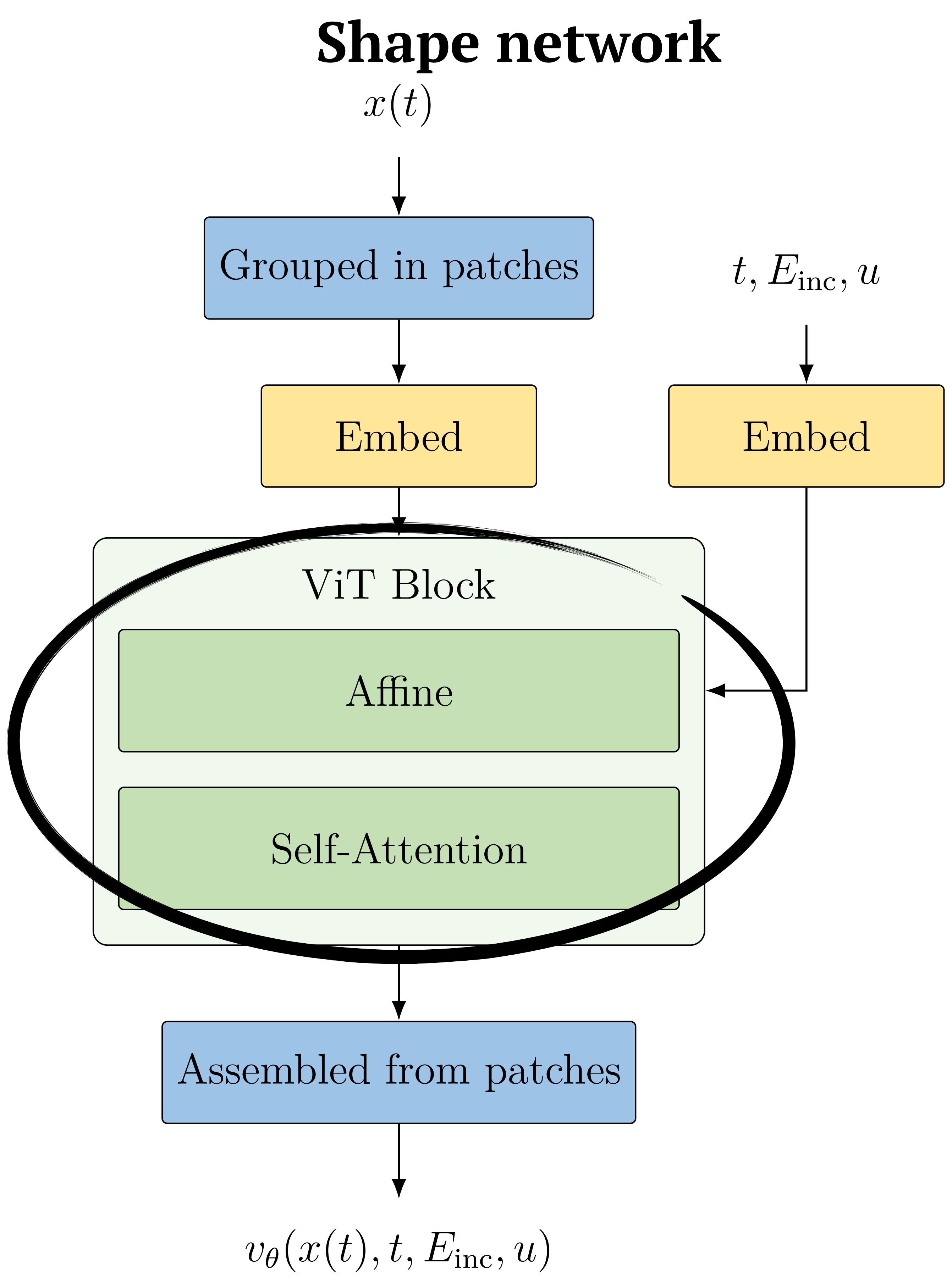
- Split detector into patches;
- Embed patches and conditions;
- Apply a residual transformation to the inputs:
  - Multi-head self-attention

$$x_h = x + \gamma_h \cdot g_h(a_h x + b_h)$$

γ_h, a_h, b_h learnable, conditioned on t, E_{inc}, u

Networks

CaloDREAM: ViT



Vision transformer:

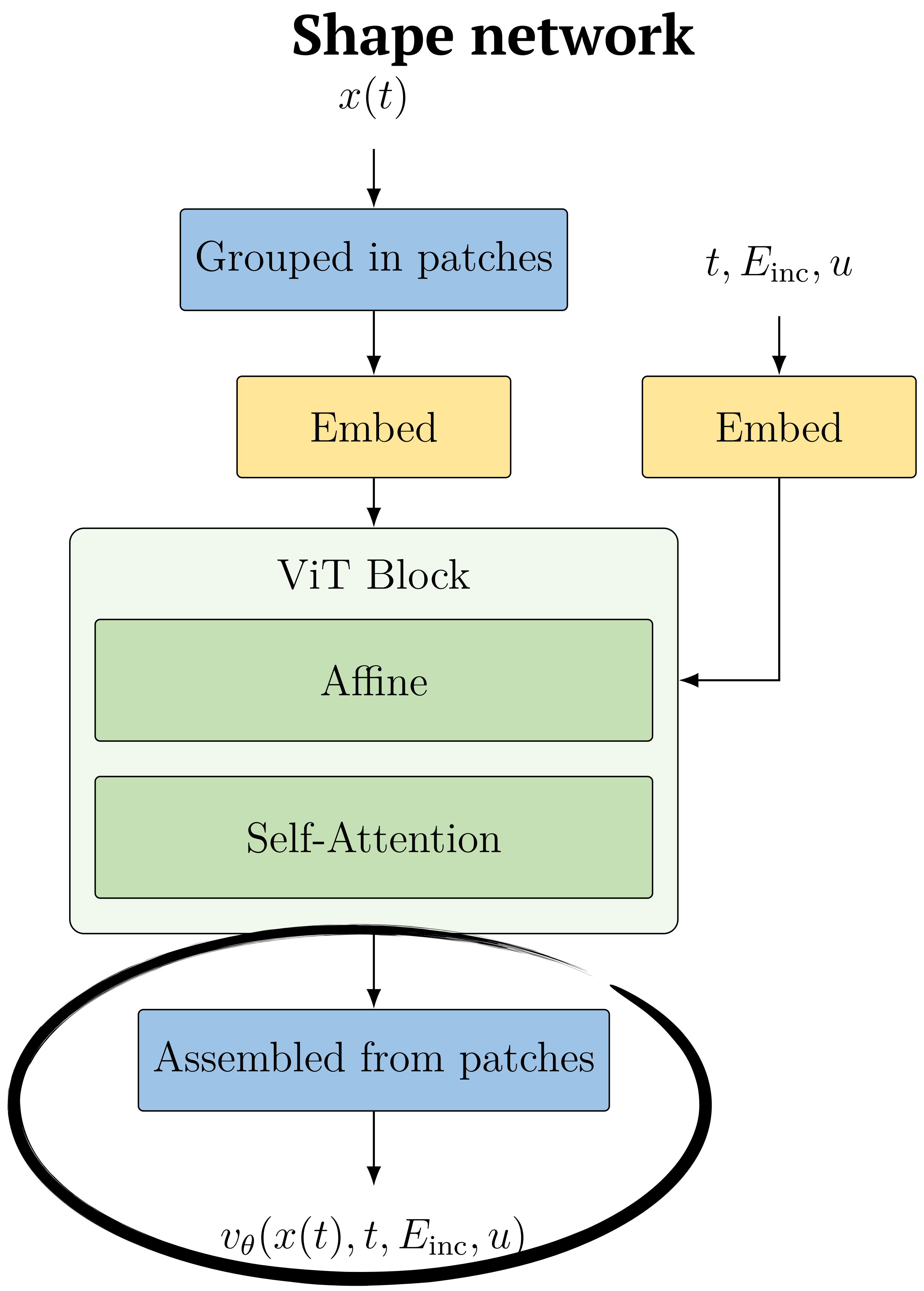
- Split detector into patches;
- Embed patches and conditions;
- Apply a residual transformation to the inputs:
  - Multi-head self-attention;
  - Fully-connected network.

$$x_l = x_h + \gamma_l \cdot g_l(a_l x_h + b_l)$$

γ_l, a_l, b_l learnable, conditioned on t, E_{inc}, u

Networks

CaloDREAM: ViT



Vision transformer:

- Split detector into patches;
- Embed patches and conditions;
- Apply a residual transformation to the inputs:
  - Multi-head self-attention;
  - Fully-connected network.

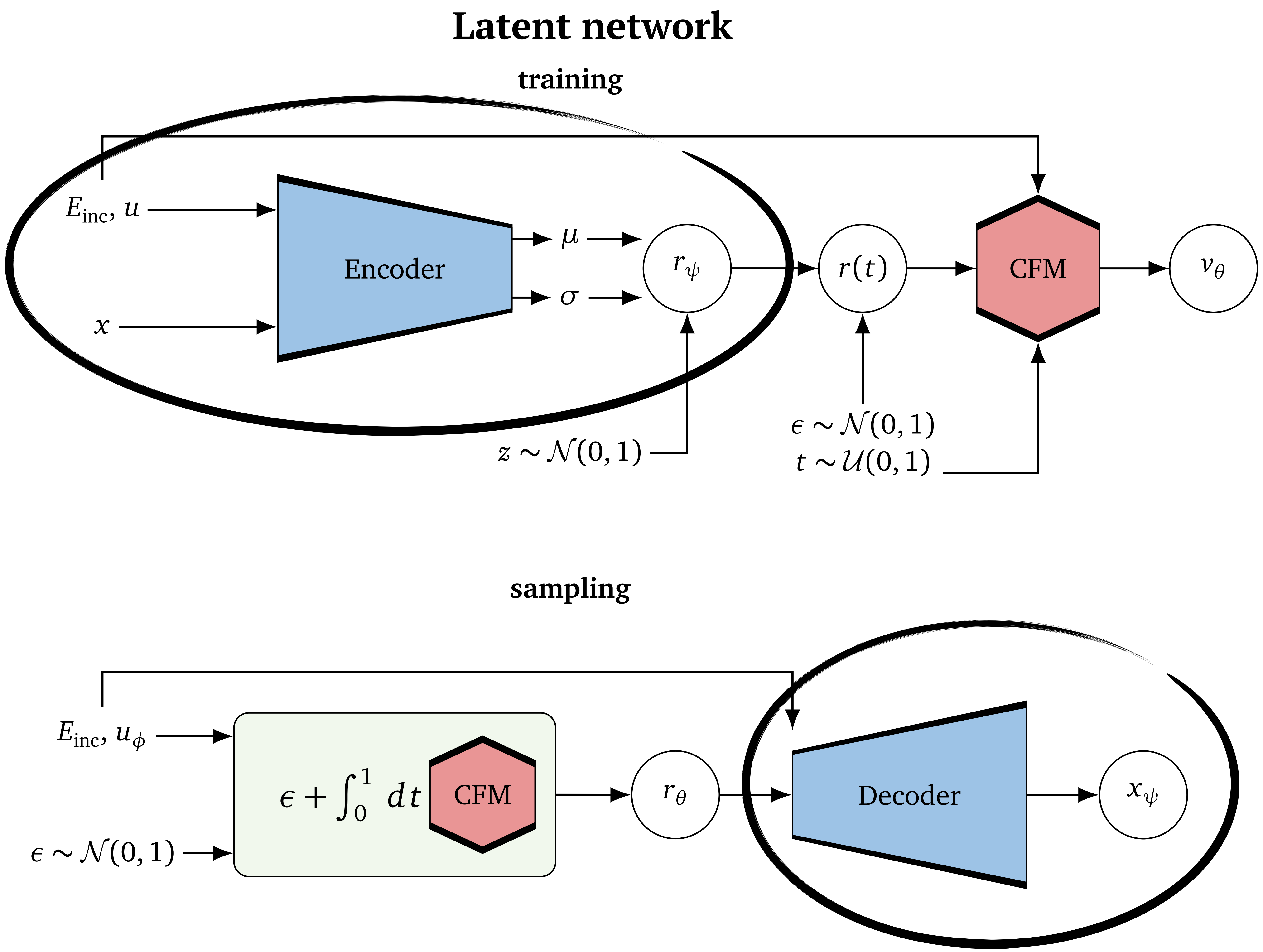
$$x_l = x_h + \gamma_l \cdot g_l(a_l x_h + b_l)$$

γ_l, a_l, b_l learnable, conditioned on t, E_{inc}, u

- Predict a v_θ for each voxel.

Networks

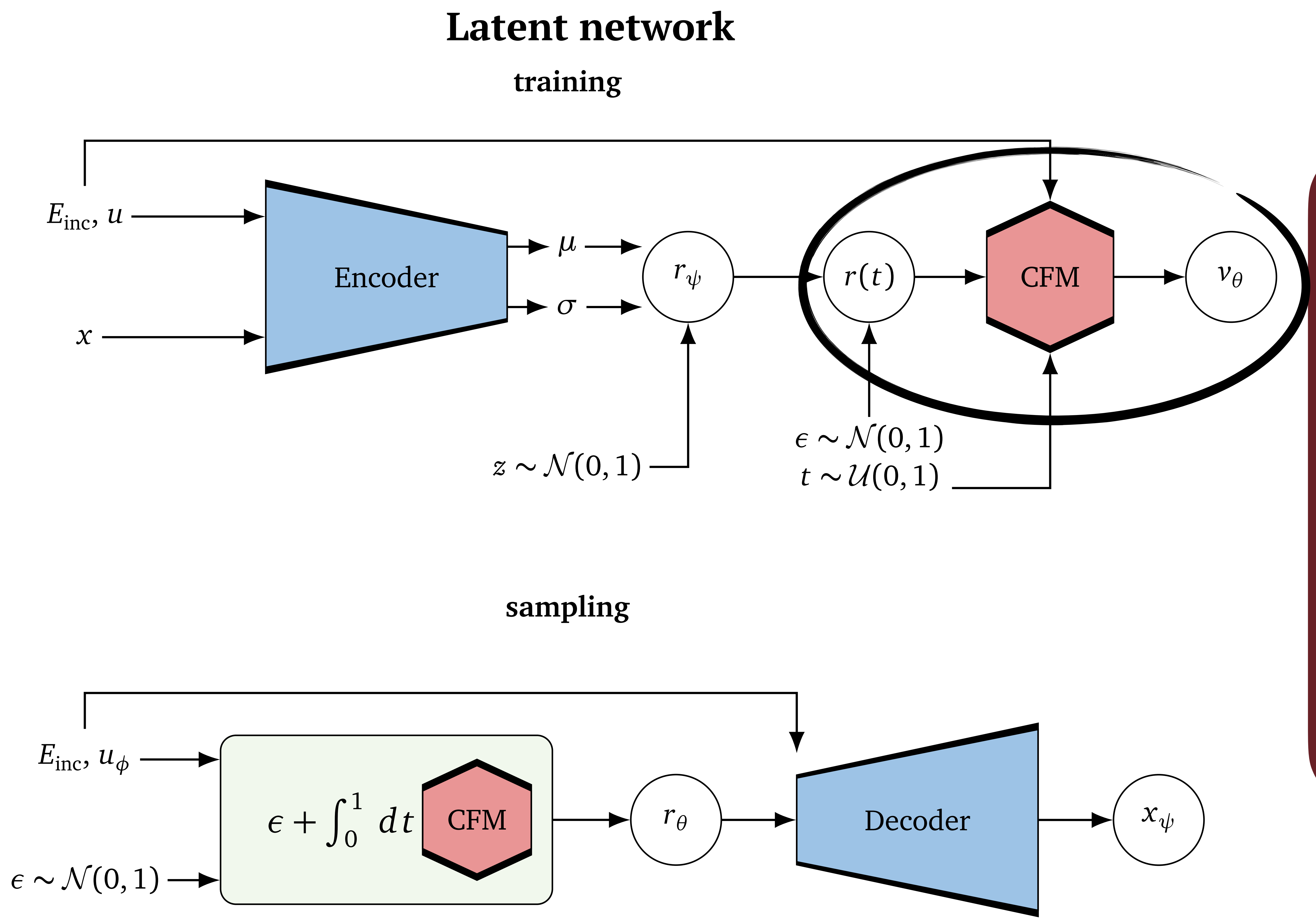
CaloDREAM: laViT



CNN autoencoder

- VAE with Bernoulli decoder;

CaloDREAM: laViT

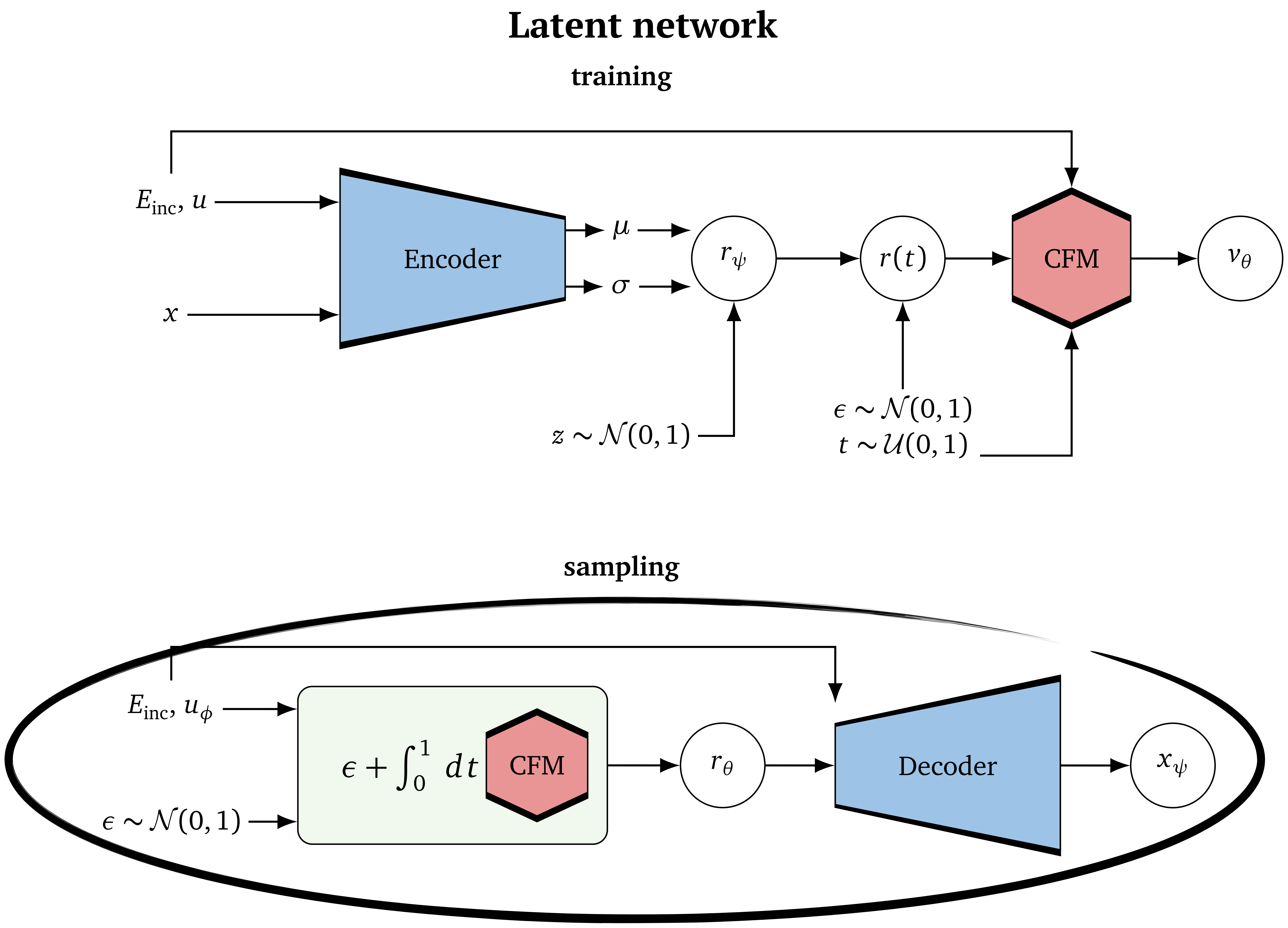


Conv. autoencoder

- VAE with Bernoulli decoder;
- train a ViT in the latent space;

Networks

CaloDREAM: laViT



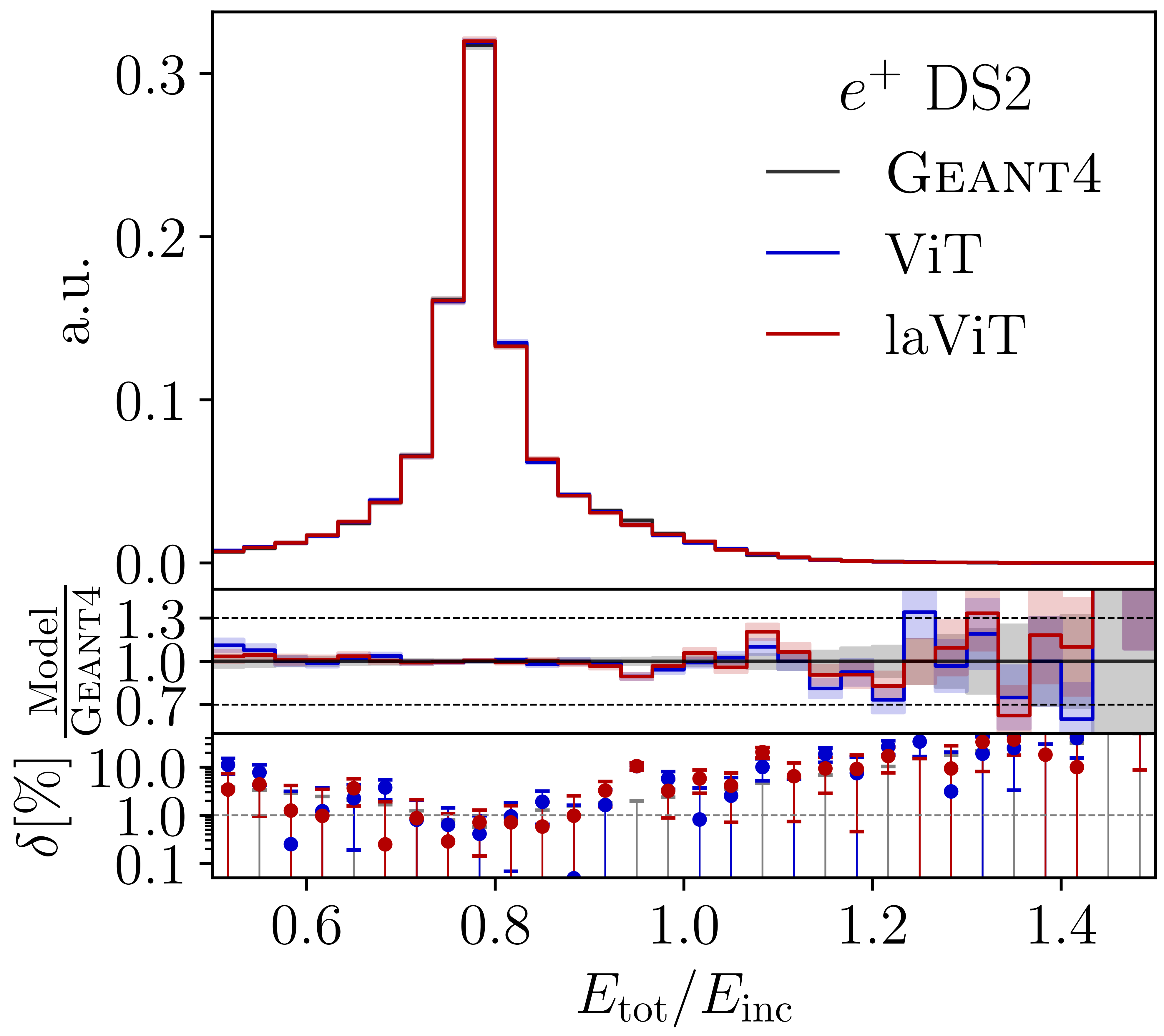
Conv. autoencoder

- VAE with Bernoulli decoder;
- train a ViT in the latent space;
- sampling done in z space:

$$u \sim p_\phi(u \,|\, E_{inc})$$
$$r \sim p_\theta(r, 1 \,|\, u, E_{inc})$$
$$x = D_\psi(r, u, E_{inc})$$

DS2 - histograms

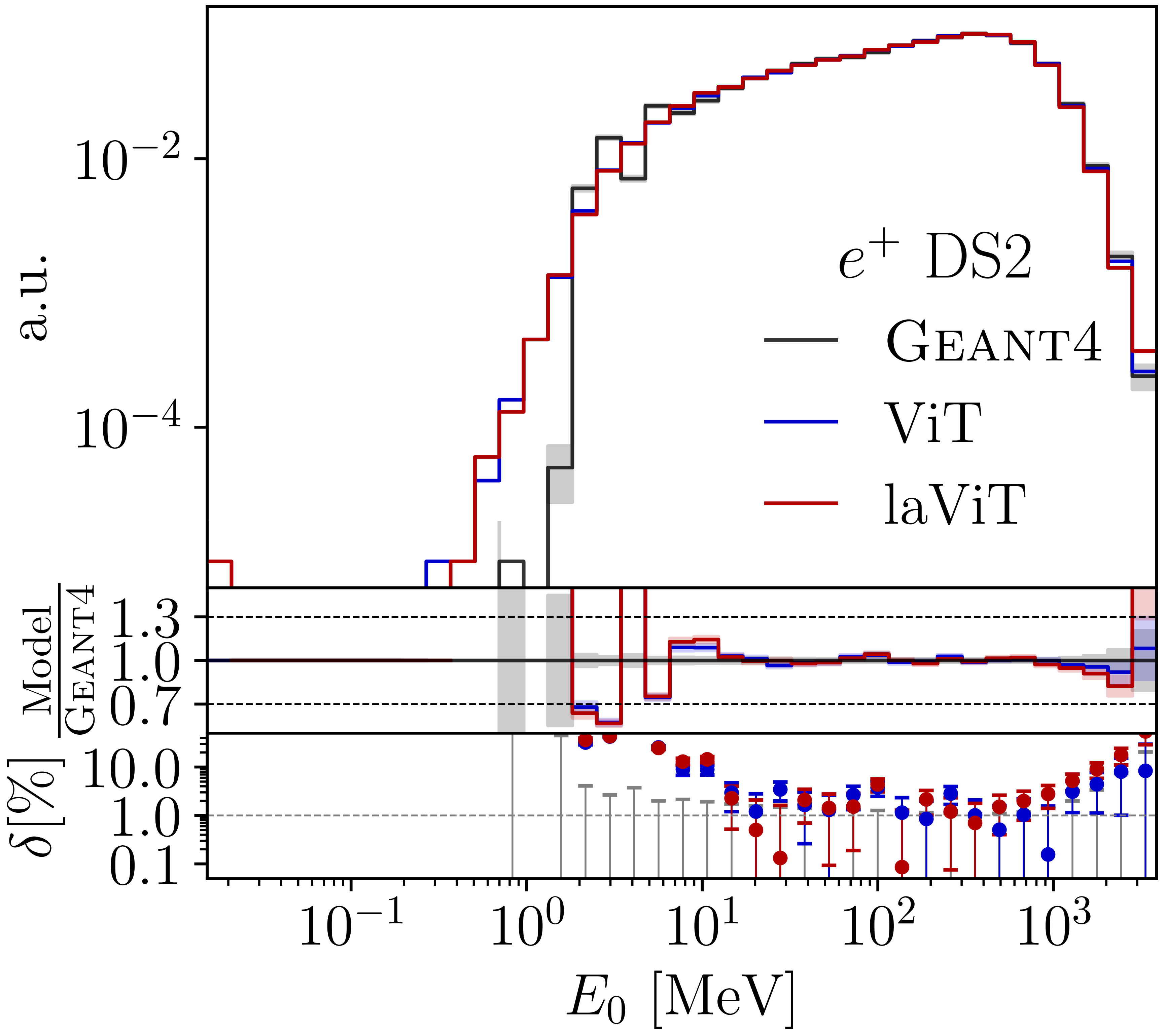
Energy ratio ✓

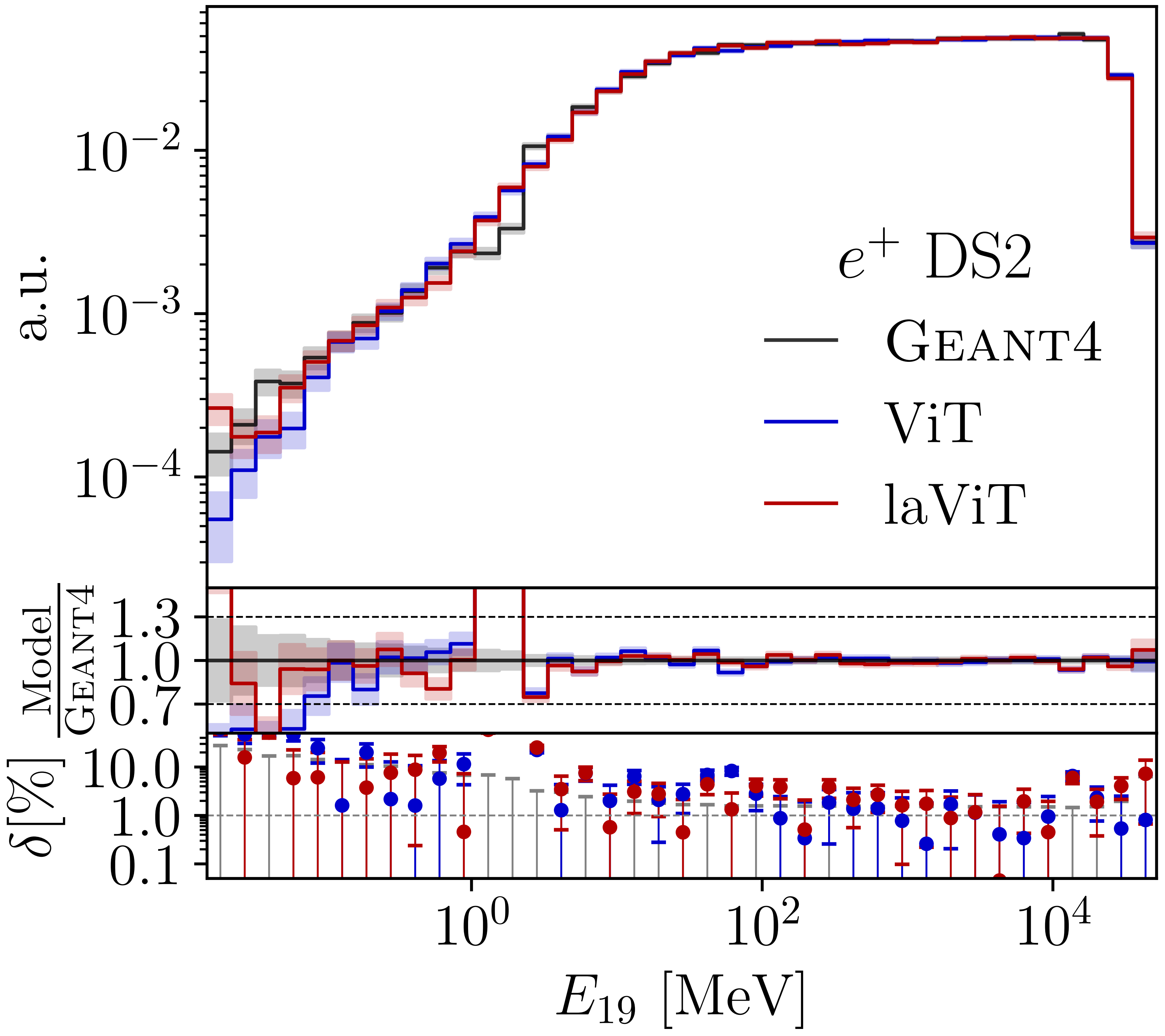


DS2 - histograms

Energy ratio ✓

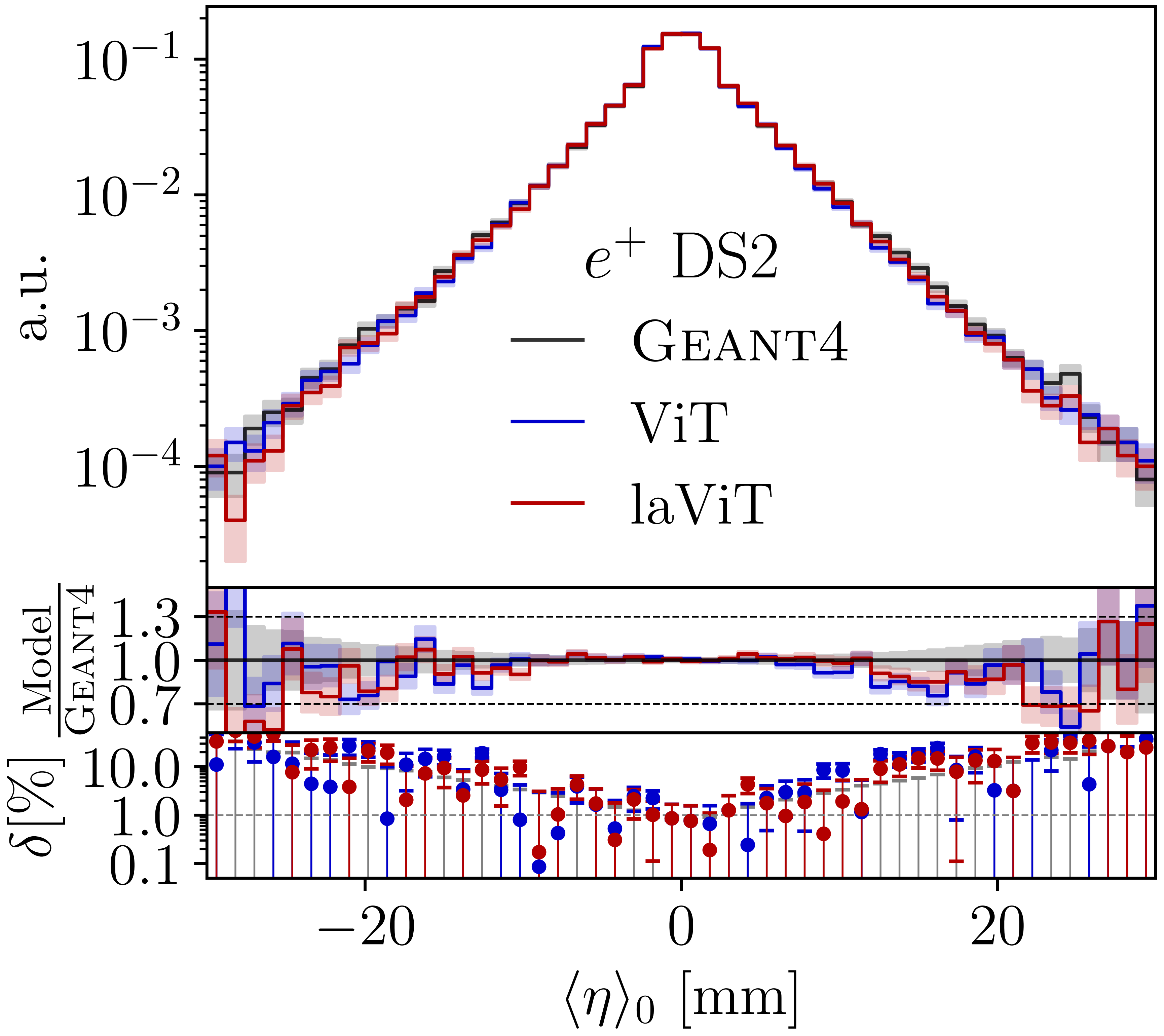
Layer energy ✓

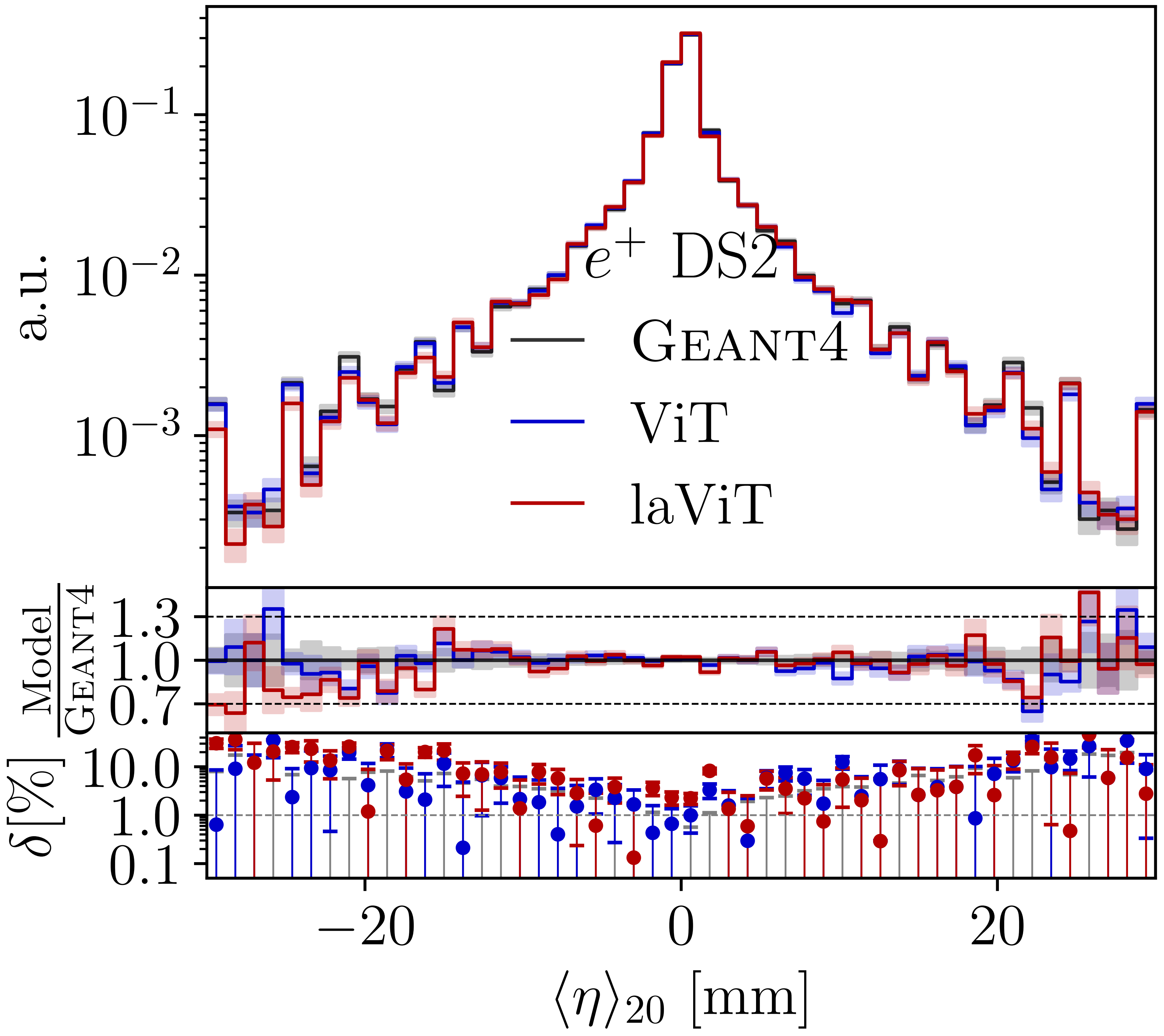




DS2 - histograms

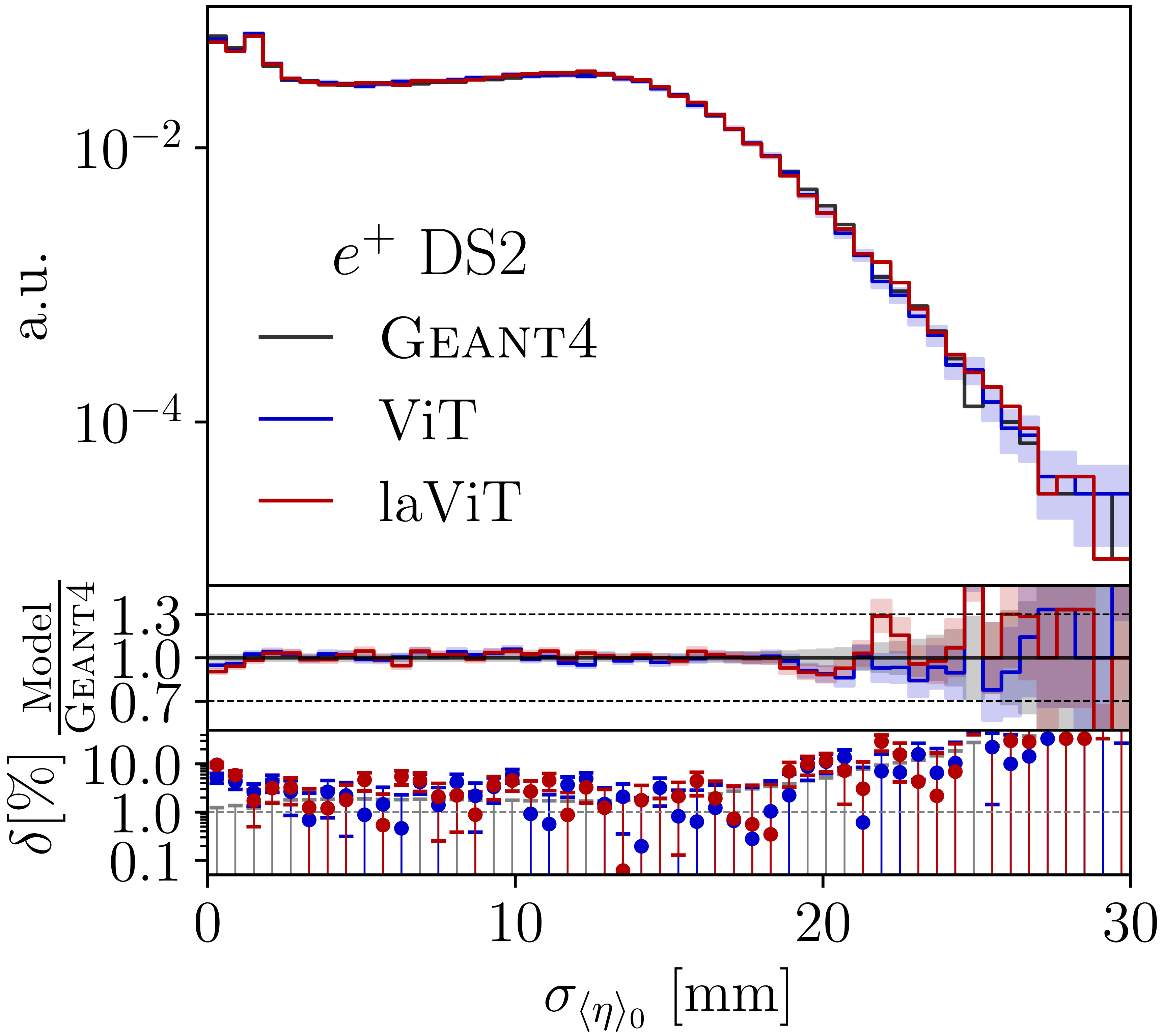
- Energy ratio ✓
- Layer energy ✓
- Center of energy ✓

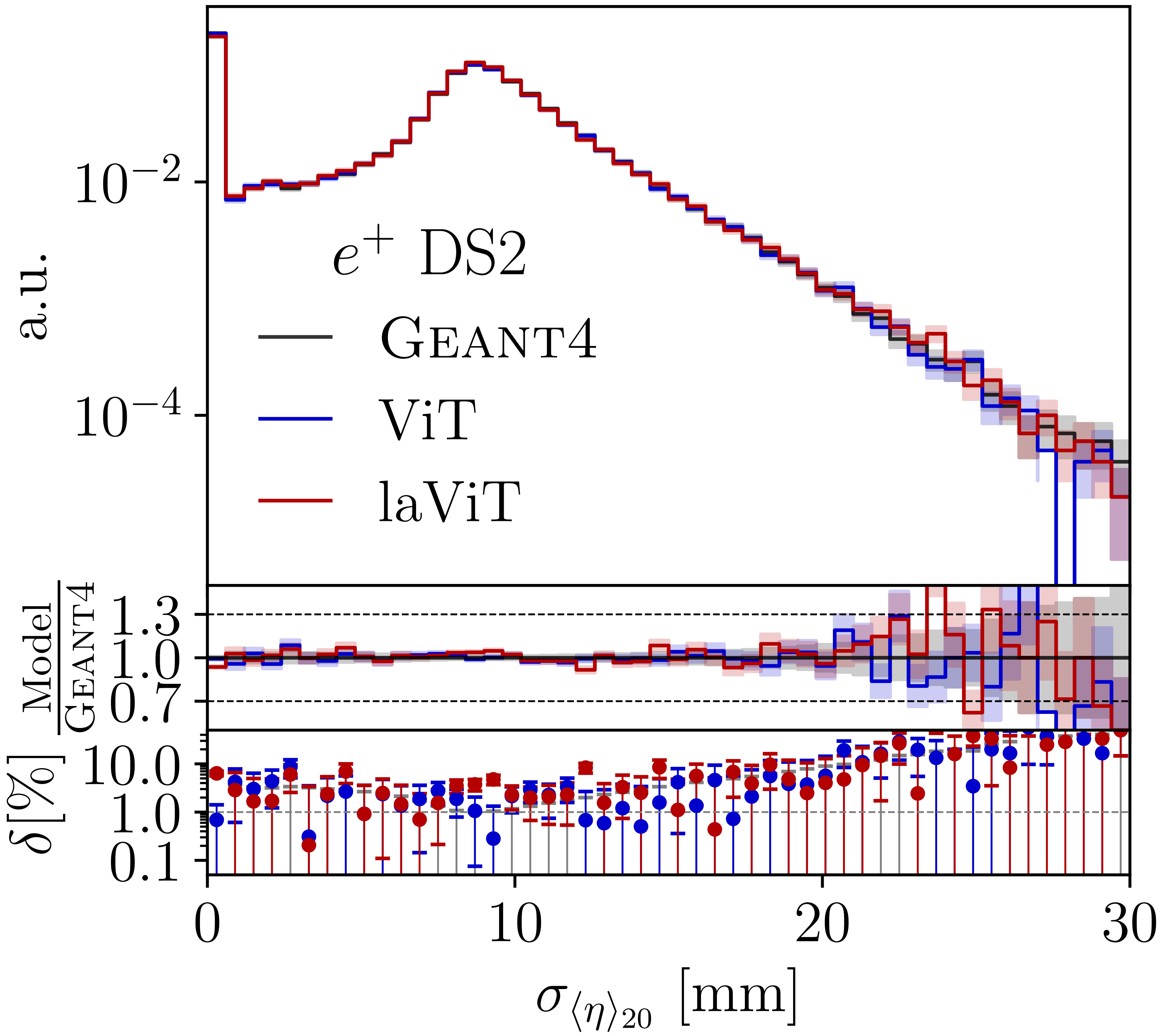




DS2 - histograms

- Energy ratio ✓
- Layer energy ✓
- Center of energy ✓
- Width of the center ✓





DS2 - histograms

Energy ratio ✓

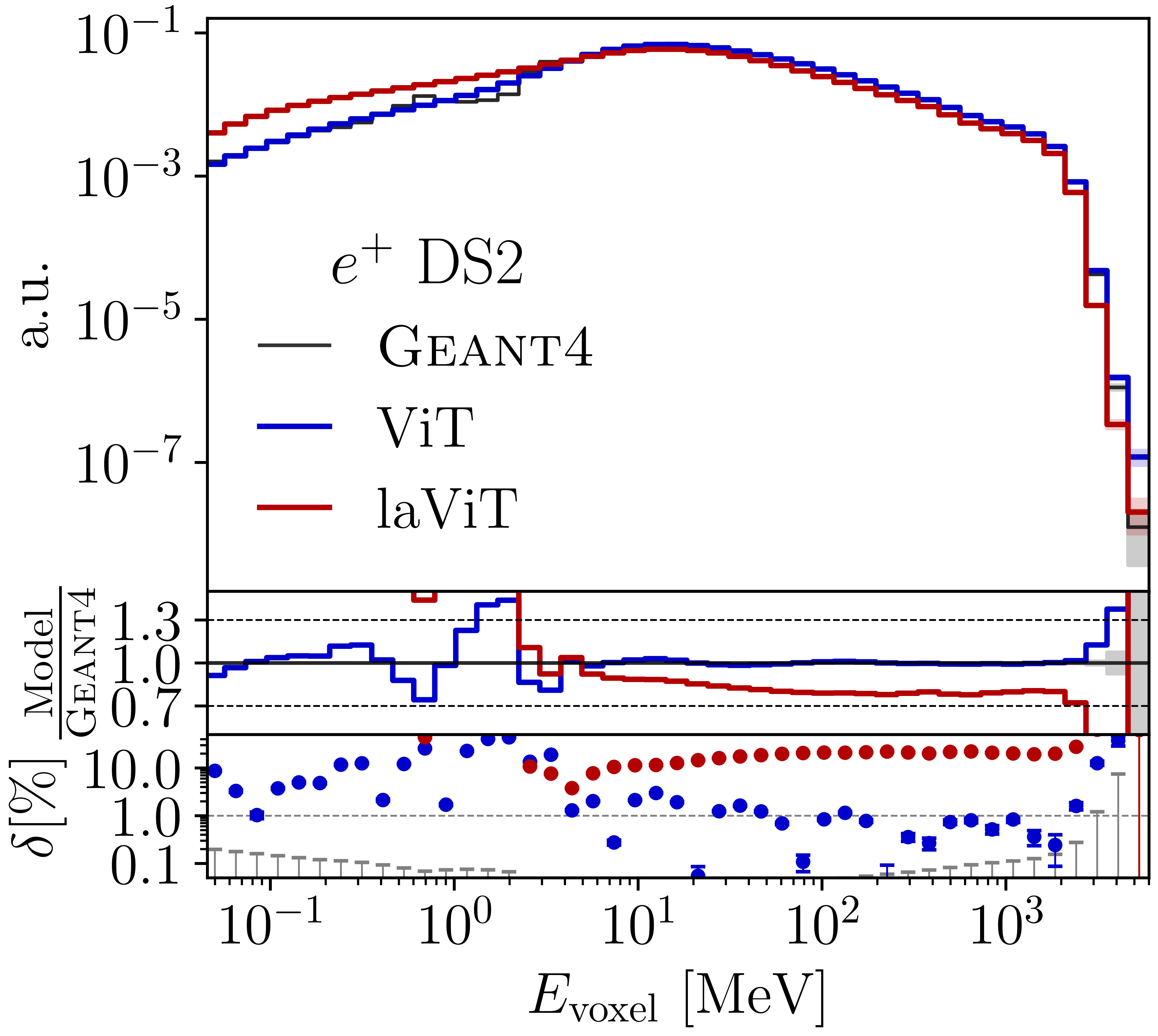
Layer energy ✓

Center of energy ✓

Width of the center ✓

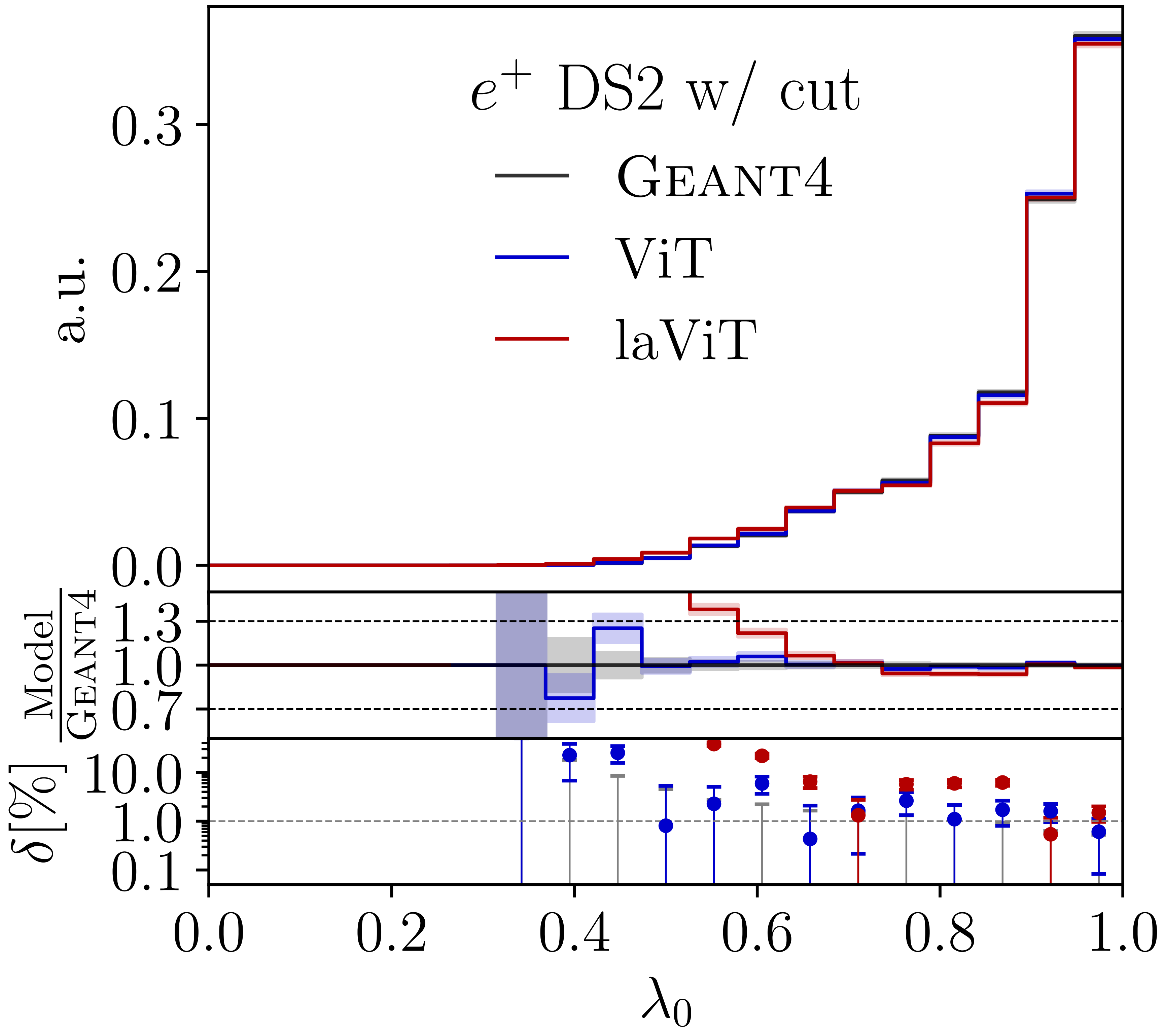
Voxel ✓

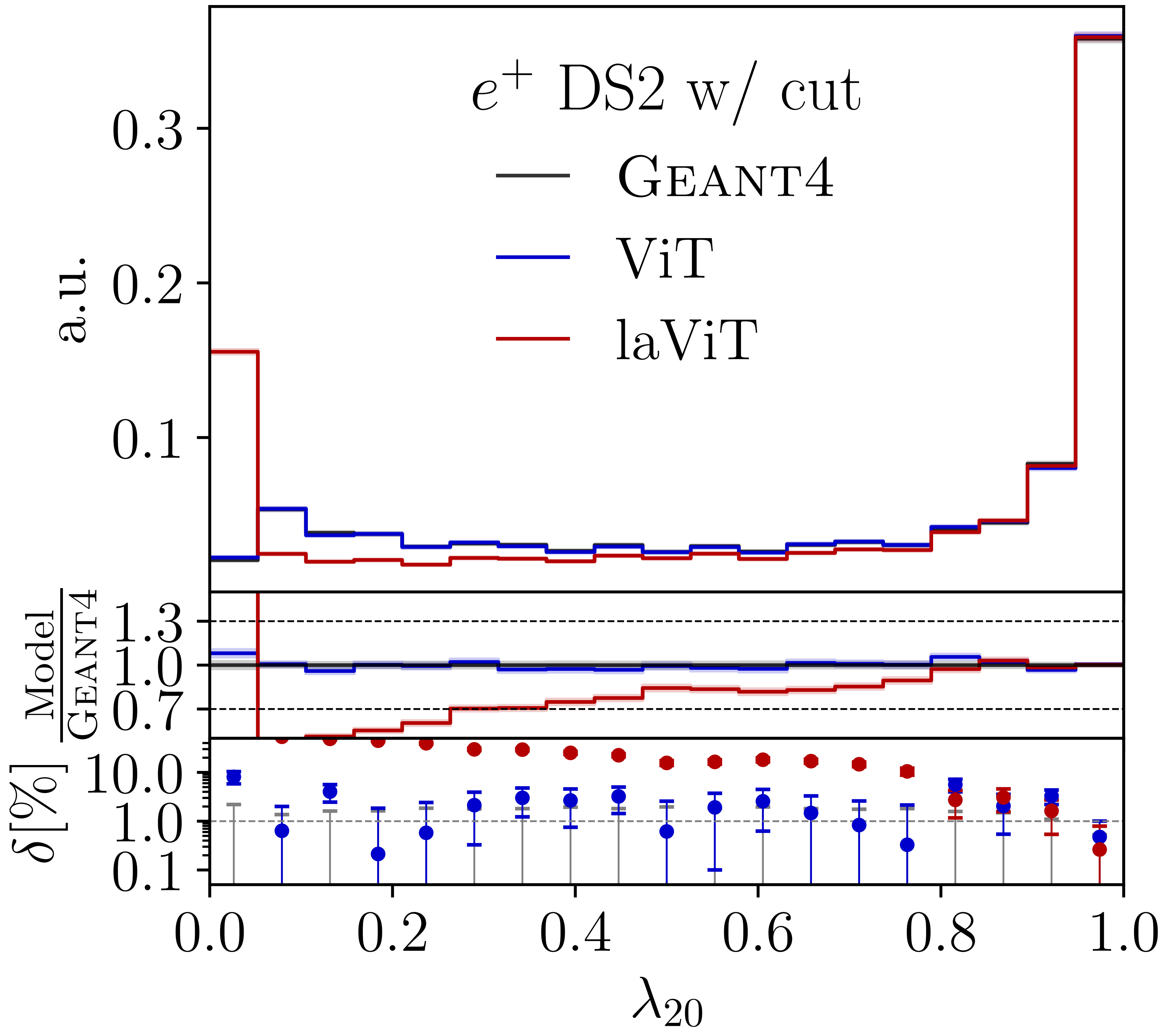
- Autoencoder is not able to reconstruct zero voxels



DS2 - histograms

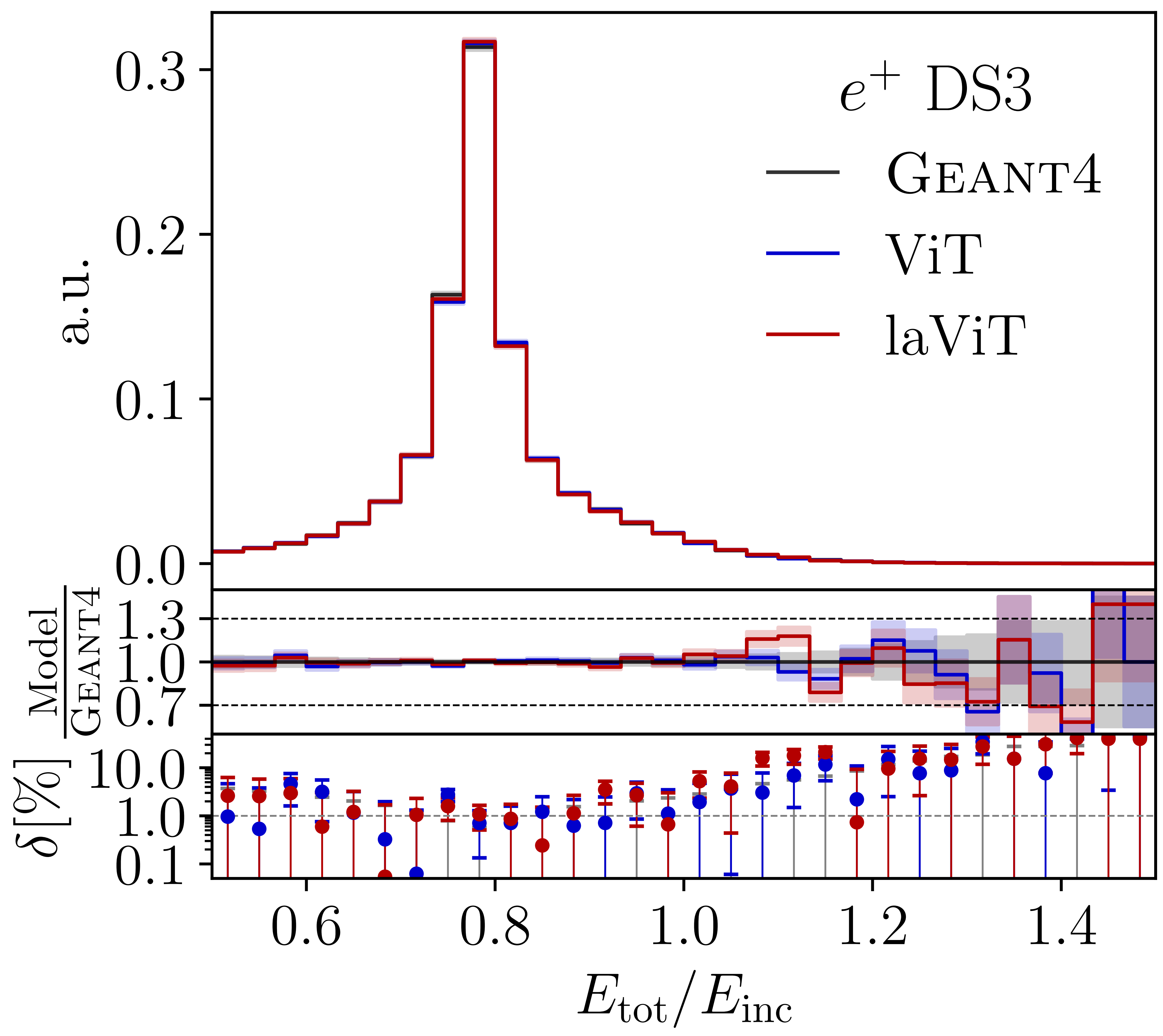
- Energy ratio ✓
- Layer energy ✓
- Center of energy ✓
- Width of the center ✓
- Voxel ✓
- Sparsity ✓





DS3 - histograms

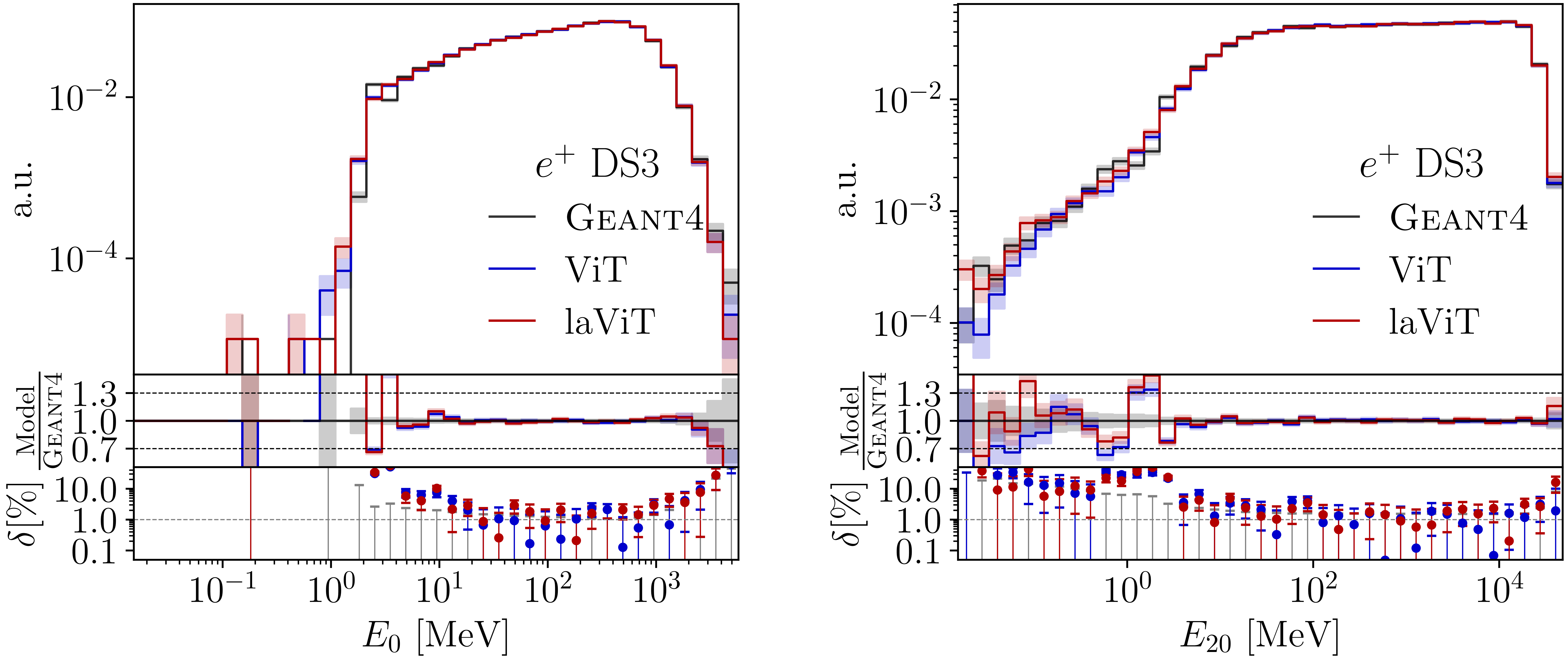
Energy ratio ✓



DS3 - histograms

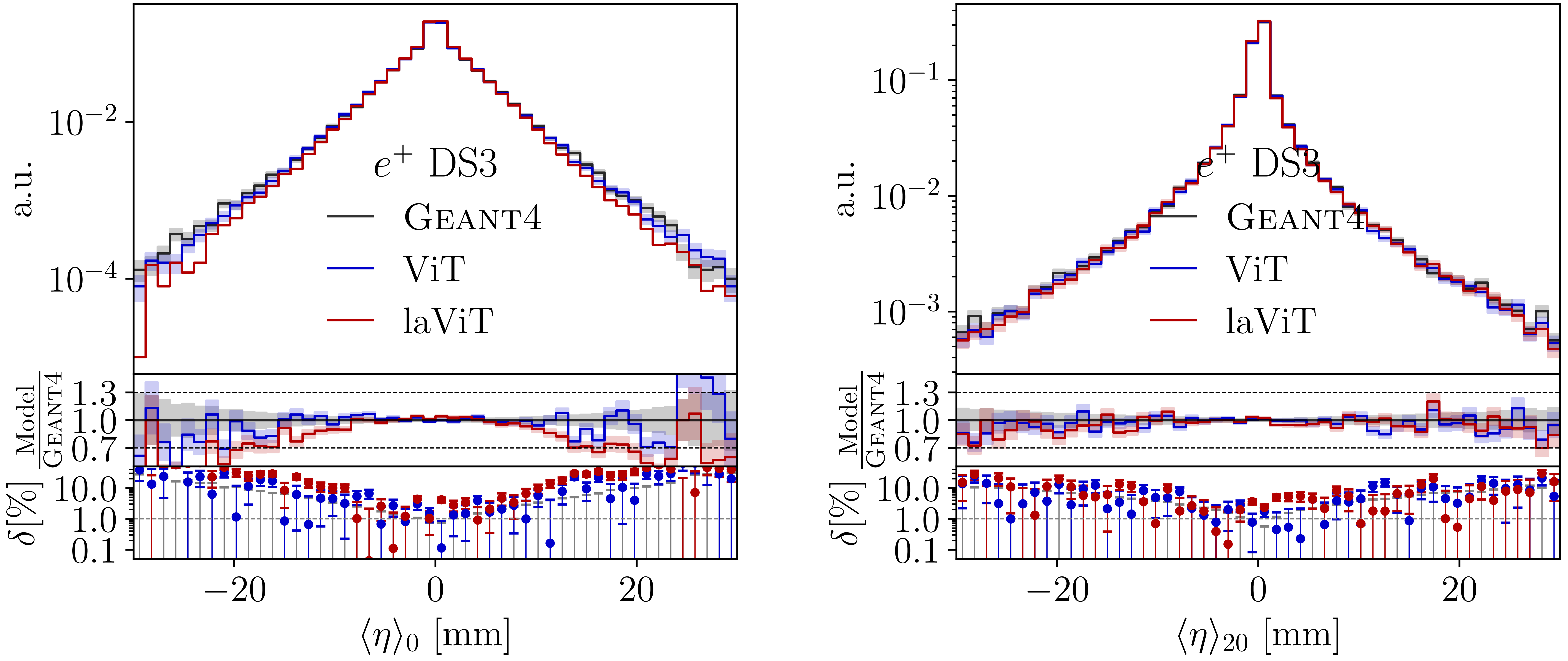
Energy ratio ✓

Layer energy ✓



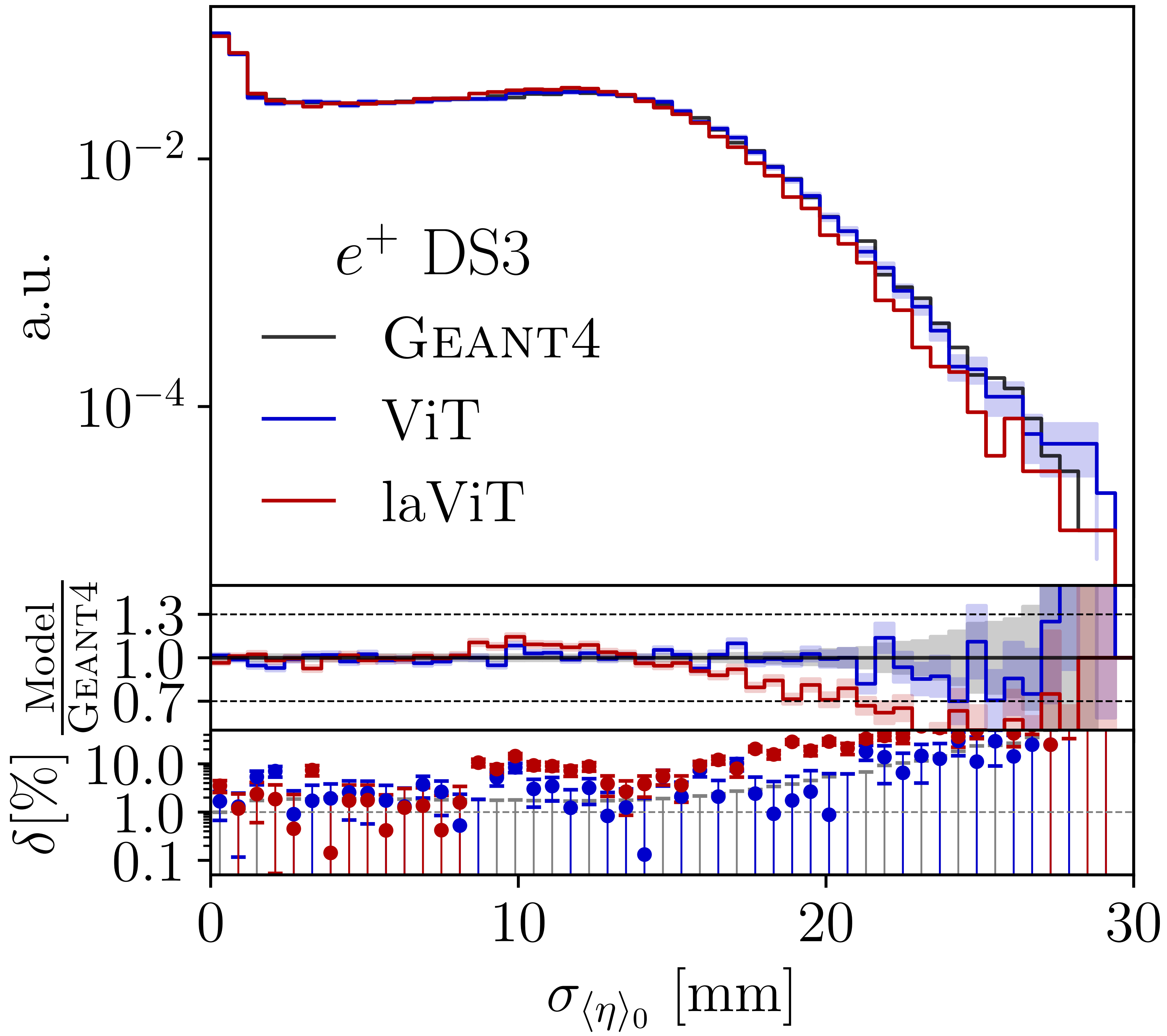
DS3 - histograms

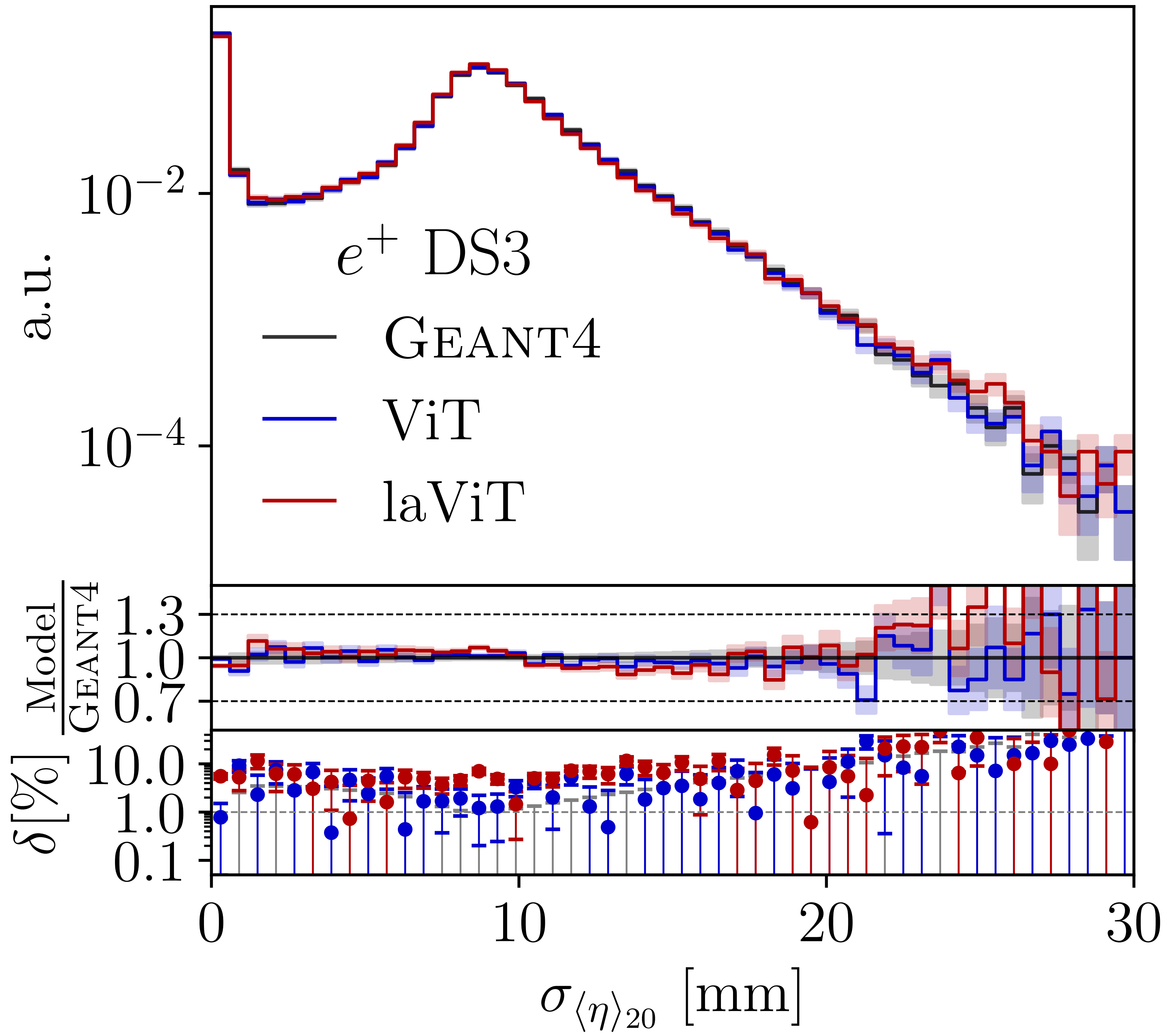
- Energy ratio ✓
- Layer energy ✓
- Center of energy ✓



DS3 - histograms

- Energy ratio ✔
- Layer energy ✔
- Center of energy ✔
- Width of the center ✔





DS3 - histograms

Energy ratio ✓

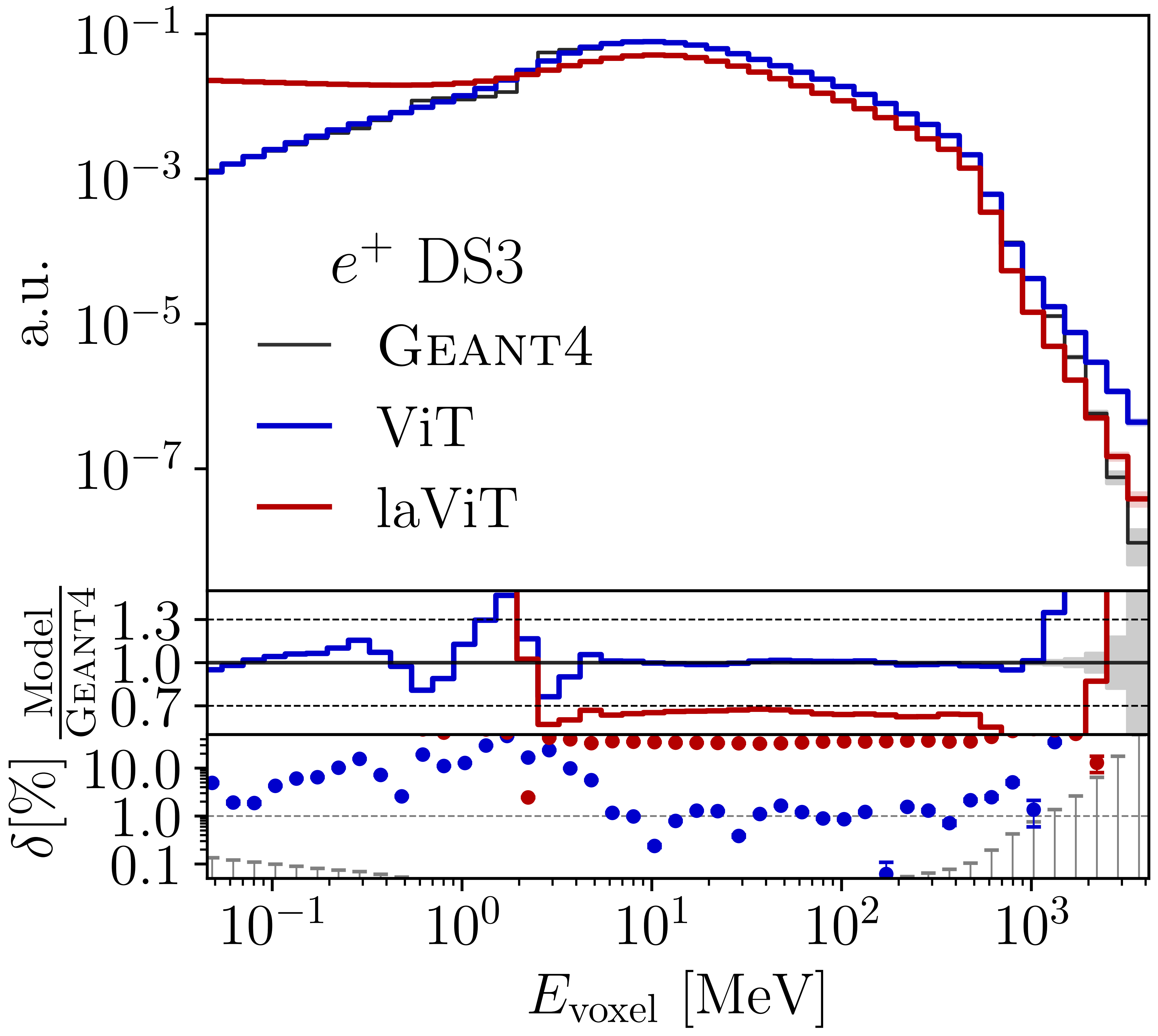
Layer energy ✓

Center of energy ✓

Width of the center ✓

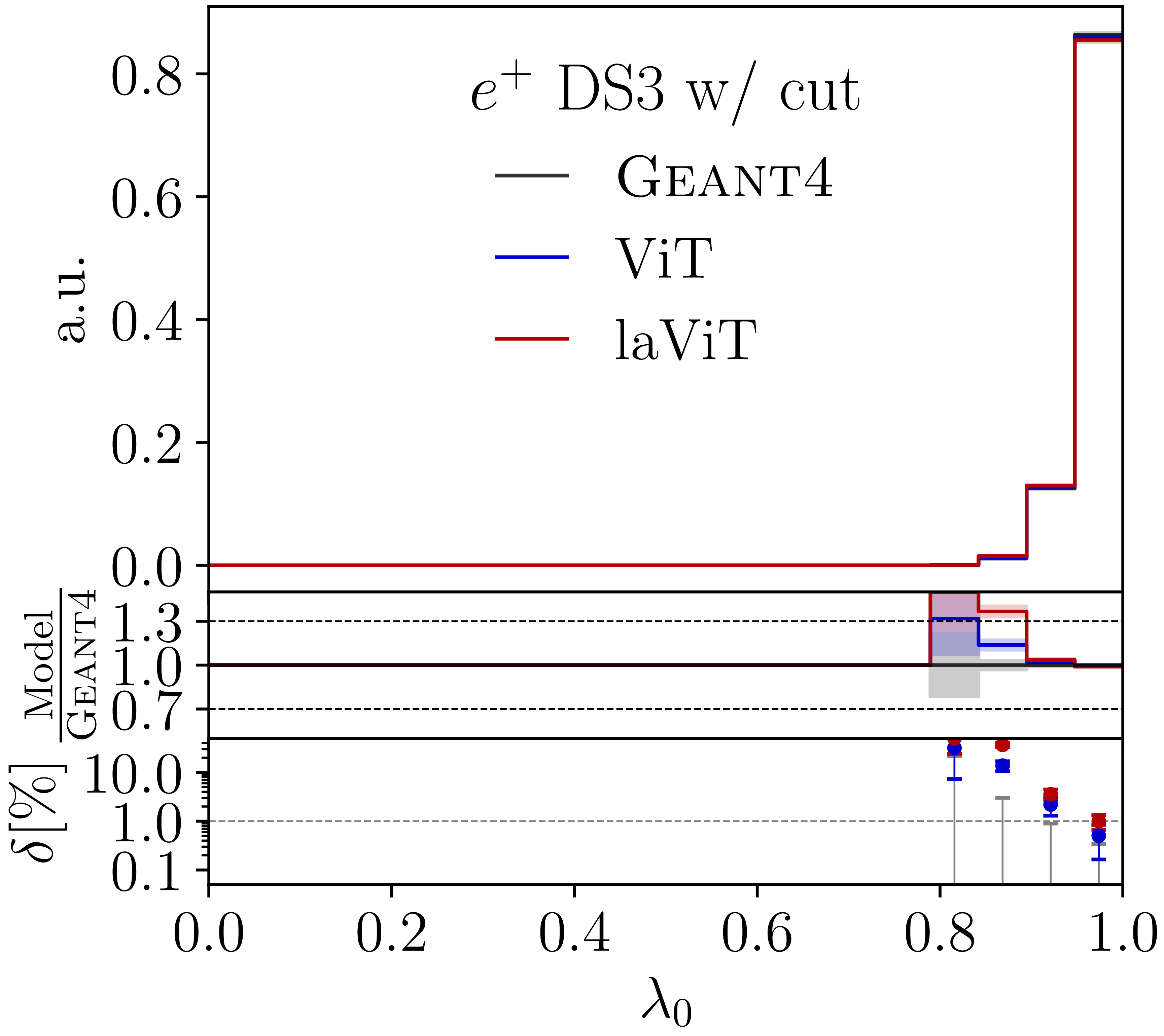
Voxel ✓

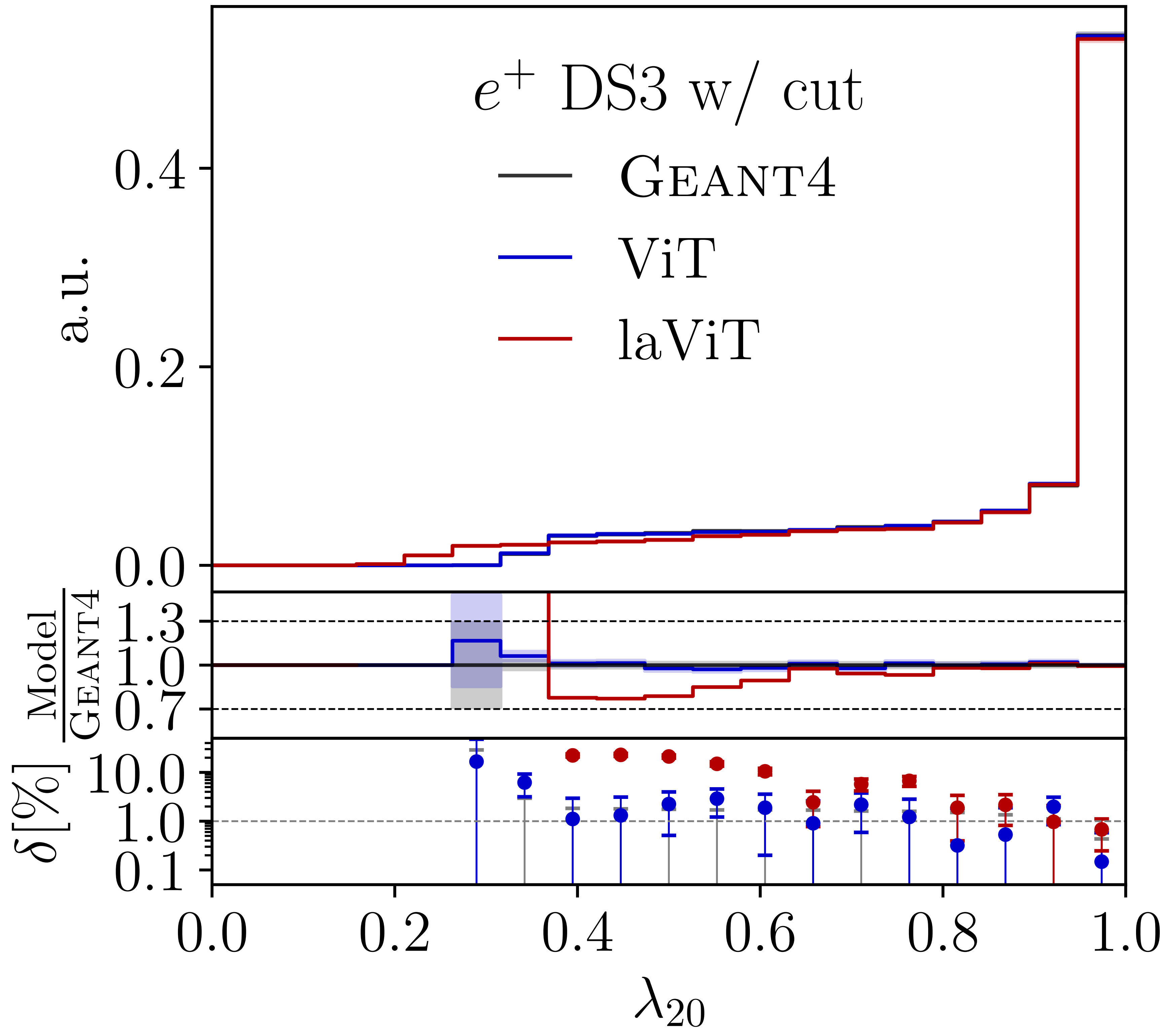
- Autoencoder is not able to reconstruct zero voxels



DS3 - histograms

- Energy ratio ✓
- Layer energy ✓
- Center of energy ✓
- Width of the center ✓
- Voxel ✓
- Sparsity ✓





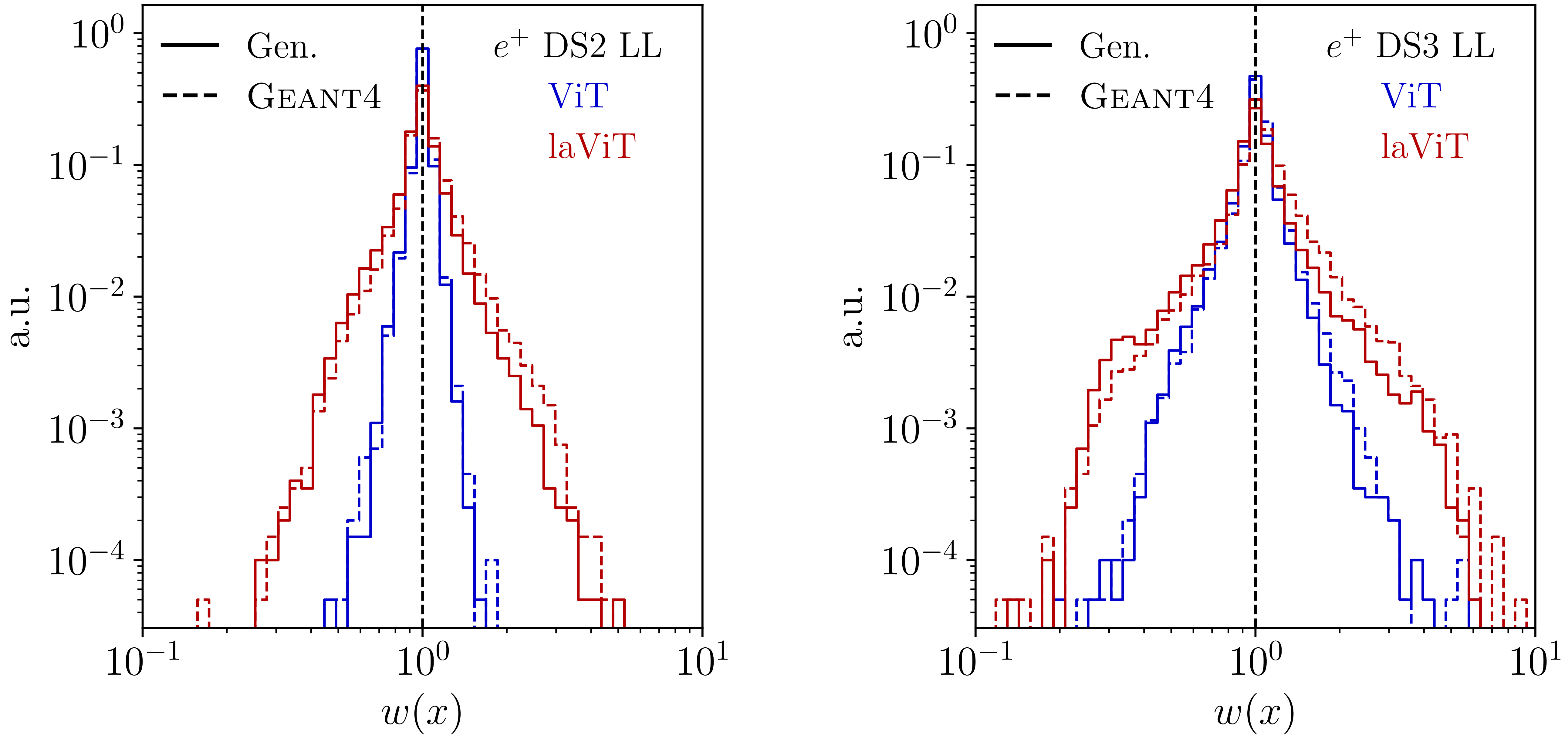
The ultimate metric

- Evaluation done in terms of the area-under-the-ROC (AUC) curve

  ⟶ indistinguishable samples if AUC=0.5

- Better to look at the weight distribution

	AUC (LL/HL)	
	DS2	DS3
ViT	0.54/0.52	0.63/0.53
laViT	0.58/0.53	0.62/0.59



Bespoke samplers

- Sampling requires multiple evaluation of the neural network;
- BNS $\longrightarrow$ keep the model fixed and learn a model specific solver;
- Take a general expression for a non-stationary solver:

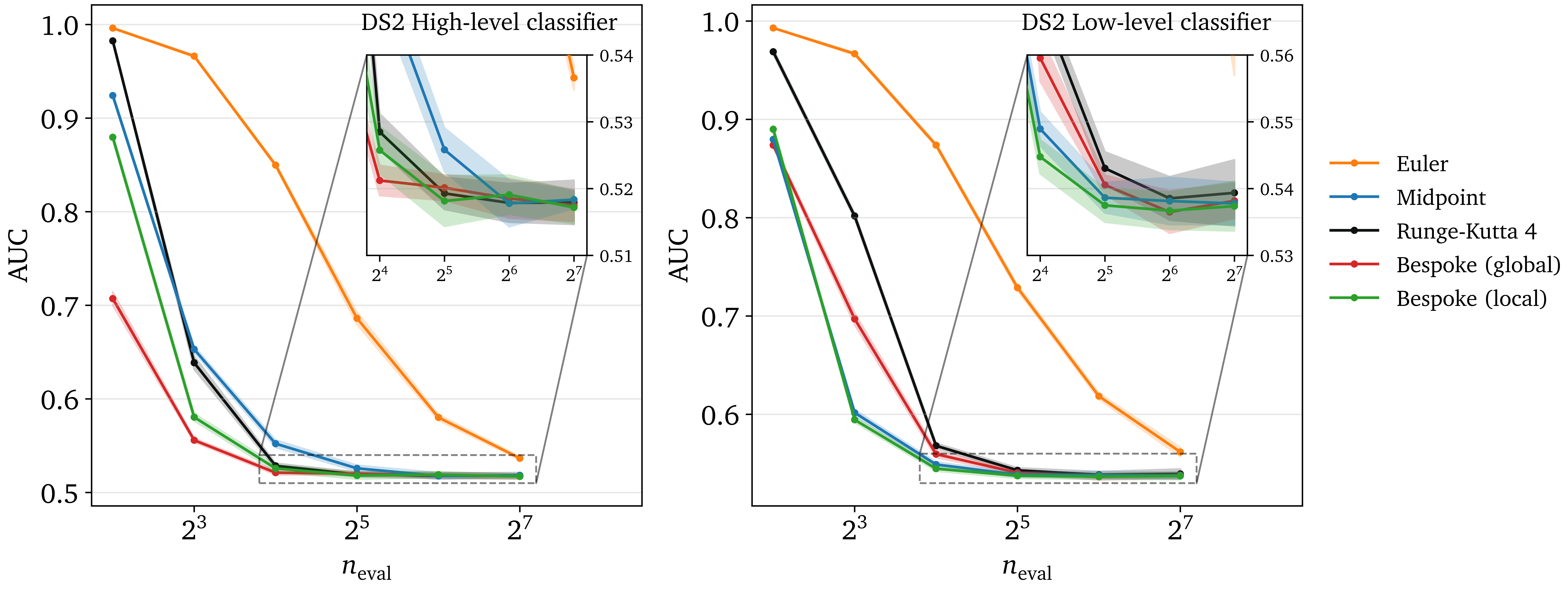
$$(t_i, x_i)_{i=0}^{N} \qquad x_{i+1} = a_i\, x_0 + b_i \cdot V_i$$

x_0 noise point
V_i vector fields
a_i, b_i, t_i learnable parameters

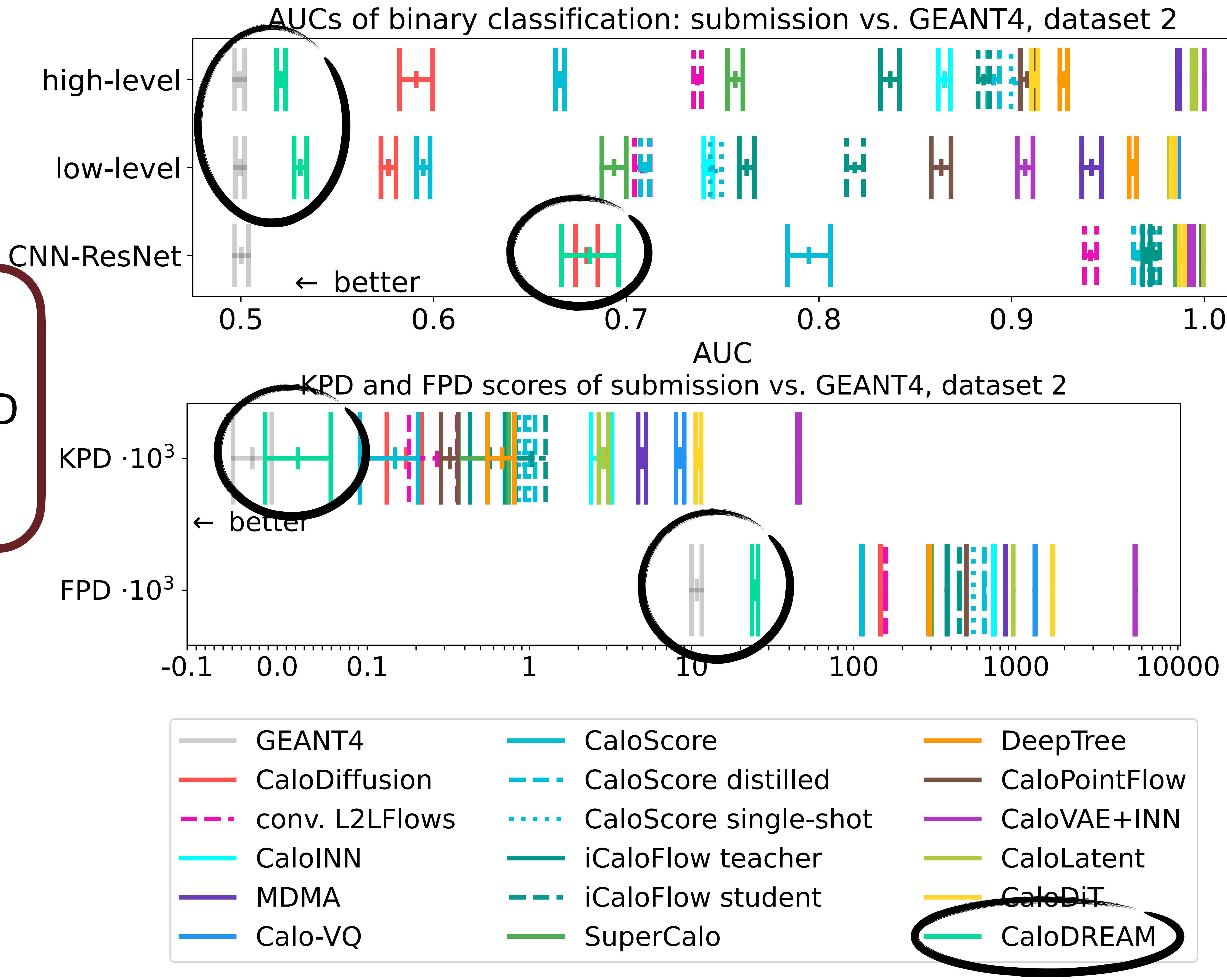
- Minimize either the global or the local truncation error:

$$\mathscr{L}_{GTE} = \langle [x_{ref}(1) - x_N]^2 \rangle_{x_0 \sim \mathcal{N}} \ , \qquad \mathscr{L}_{LTE} = \langle \sum_{i=0}^{N-1} \left[x_{ref}(t_{i+1}) - (a_i x_0 + b_i \cdot V_{ref,i}) \right]^2 \rangle_{x_0 \sim \mathcal{N}}$$

Bespoke samplers



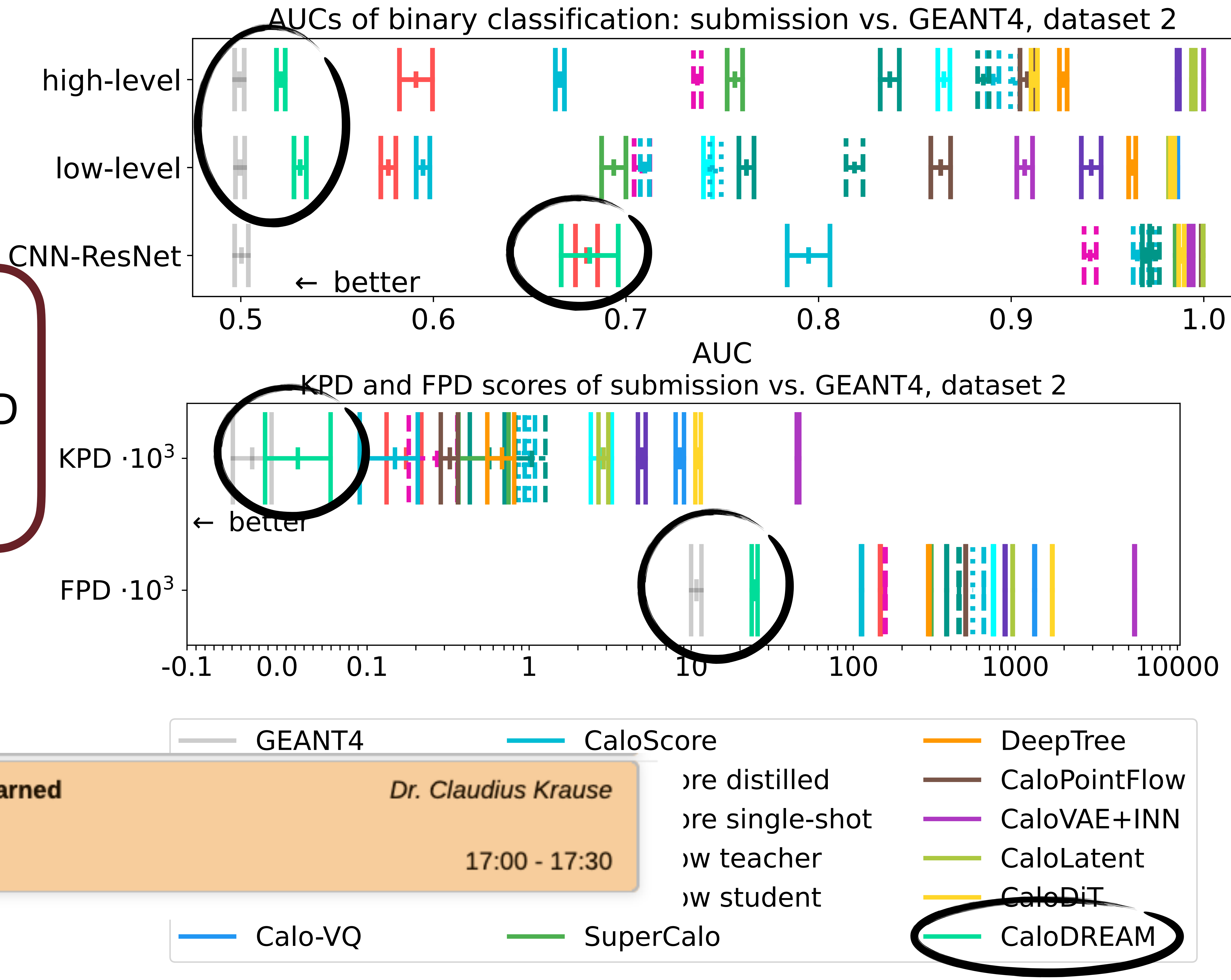
From the CaloChallenge



Metrics from the CaloChallenge:

- Great performance over classifiers and KPD/FPD
- Also faster than other diffusion models

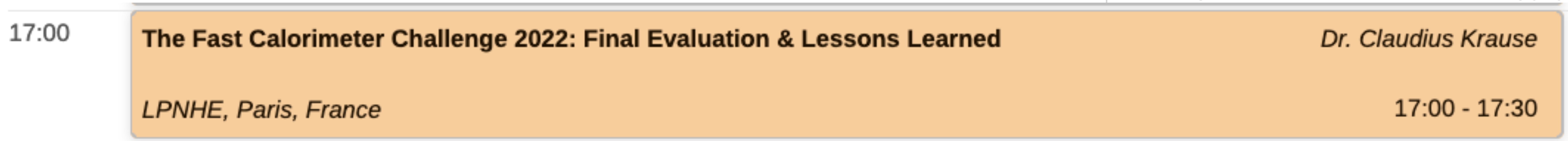
From the CaloChallenge



Metrics from the CaloChallenge:

- Great performance over classifiers and KPD/FPD
- Also faster than other diffusion models

More in the writeup: arXiv:2410.21611
and in Claudius' talk!



17:00

The Fast Calorimeter Challenge 2022: Final Evaluation & Lessons Learned *Dr. Claudius Krause*

LPNHE, Paris, France 17:00 - 17:30

Conclusions

- Diffusion models are state-of-the-art for fastsim;
- CaloDREAM:
  - awesome generation quality for both DS2/DS3;
  - reduce number of function evaluation with BNS;
- Training can get expensive:
  - size of DS3 network limited by our available resources;
- Now looking at making the model usable for the community.

Conclusions

- Diffusion models are state-of-the-art for fastsim;
- CaloDREAM:
  - awesome generation quality for both DS2/DS3;
  - reduce number of function evaluation with BNS;
- Training can get expensive:
  - size of DS3 network limited by our available resources;
- Now looking at making the model usable for the community.

Thank you for your attention!

Backup

The ultimate metric

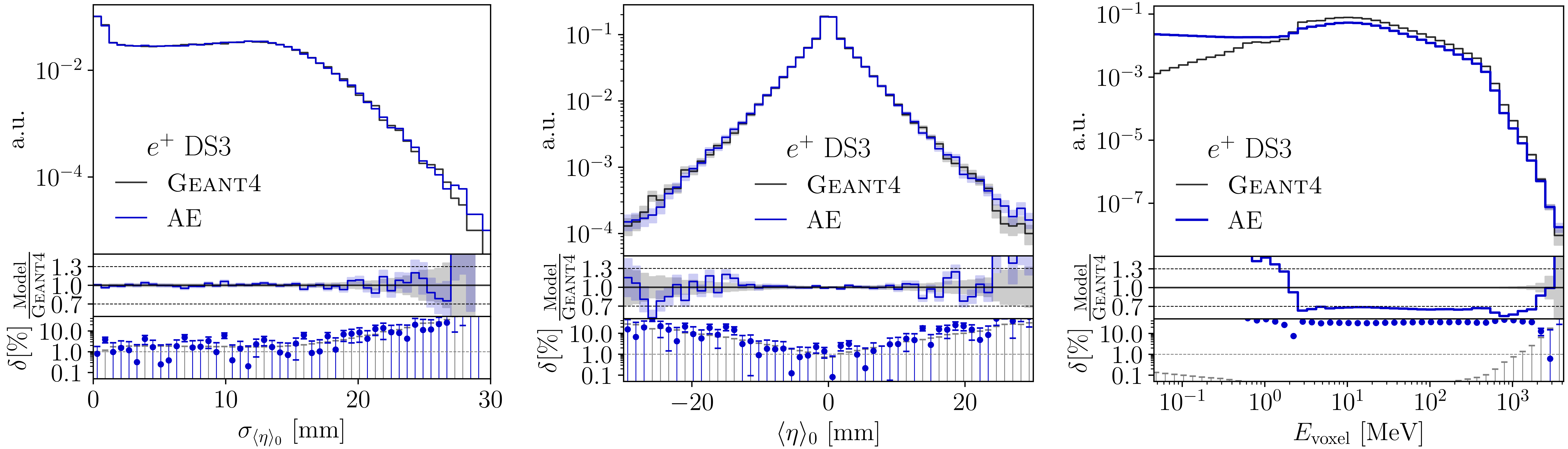
from arXiv:2305.16774

- Classifiers are the best tools we have to test our generative networks;
- the output approximates the quantity:

$$C(x) = \frac{p_{data}}{p_{data} + p_\theta} \qquad \frac{p_{data}}{p_\theta} = \frac{C(x)}{1 - C(x)}$$

- Optimal observable for a two hypothesis test according to the Neyman-Pearson lemma
- Proper training is essential: architecture, over-fitting, calibration,...
- we can easily extract weights from properly trained classifiers $\longrightarrow w(x) \approx \frac{p_{data}}{p_\theta}(x)$

AE reco.



a.u.
e^+ DS3
GEANT4
AE
$\frac{\text{Model}}{\text{GEANT4}}$
$\delta[\%]$
$\sigma_{\langle\eta\rangle_0}$ [mm]
$\langle\eta\rangle_0$ [mm]
E_{voxel} [MeV]