Joint Variational Auto-Encoder for Anomaly Detection in HEP

Lorenzo Valente [1*], Luca Anzalone [1,2], Marco Lorusso [1,2], Daniele Bonacorsi [1,2]

[1] University of Bologna
[2] INFN Bologna
[*]Speaker

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Introduction

Problem & Data

Anomaly Detection

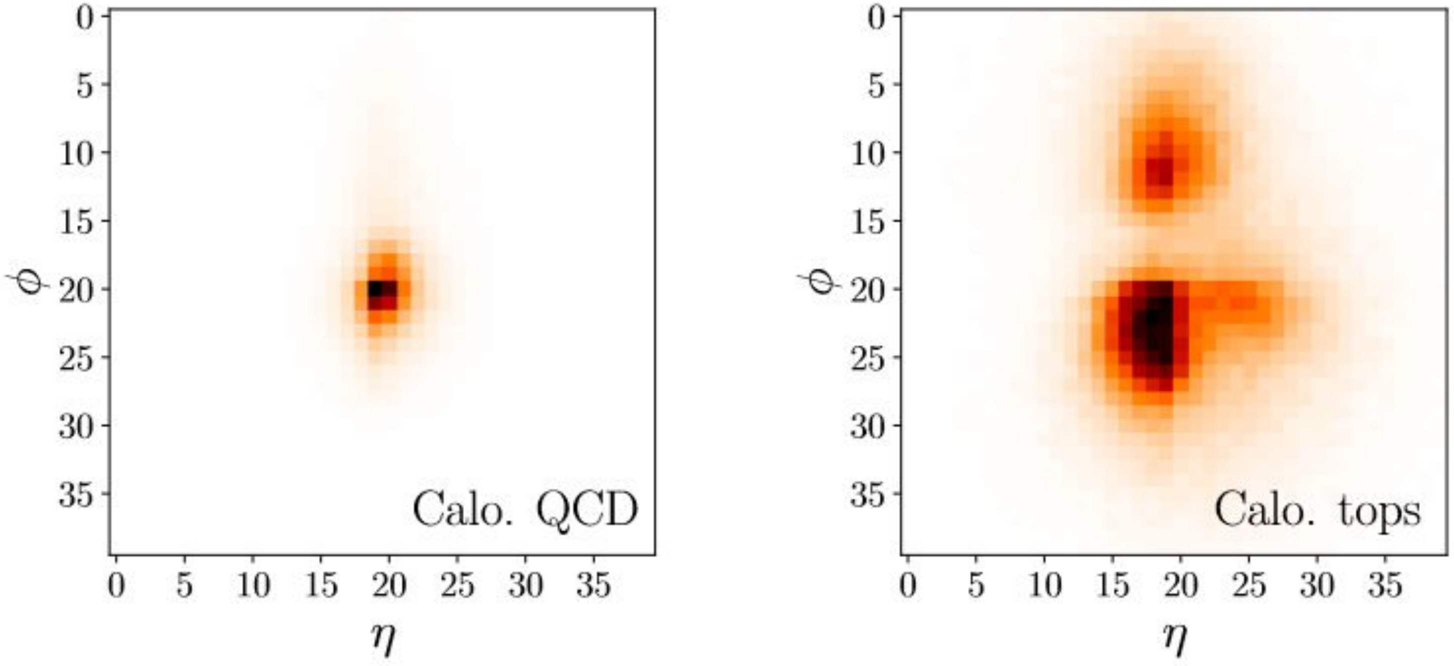
Auto-Encoders

Problem: Classification of QCD Jets

The problem is framed as **anomaly detection**: no assumption of signal form!

- Knowledge about the QCD background is only necessary.
- The model learns the QCD features, instead of adapting to the signal (as a classifier would do.)
- Must define an AD score.
- The signal samples are only used for evaluating the final performance.

⇒ The AD model is **signal agnostic**!

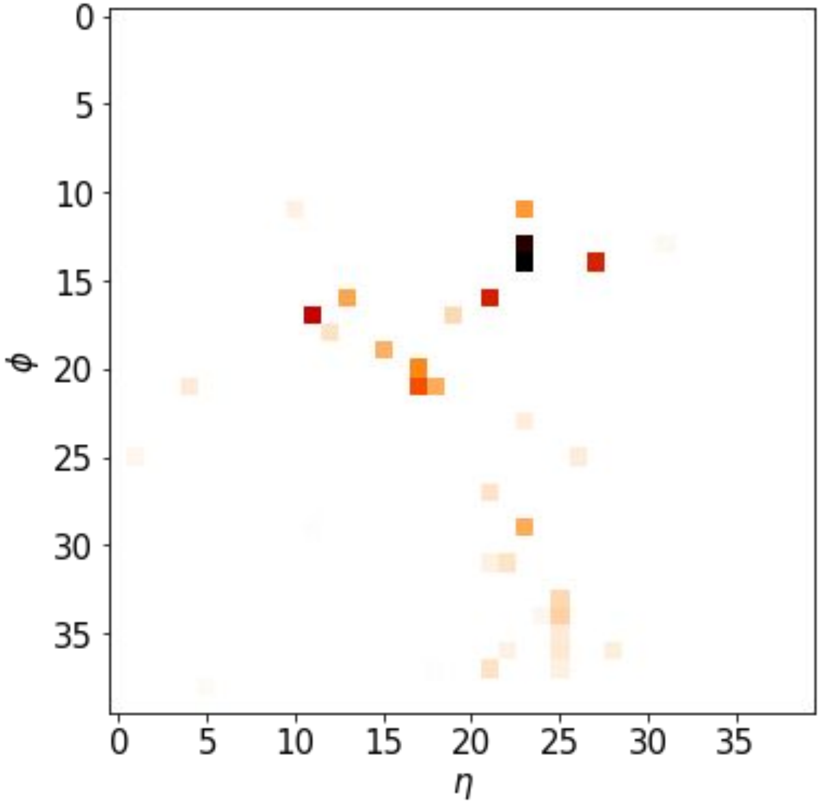


Average of 200k images each.

Dataset: QCD vs Top Jets

Data from the **Top-tagging challenge**: 200k QCD and 200k Top jets.

- Simulated jets with Pythia8 at 14TeV.
- **Selection:** jets with $p_T \in [550{,}650]$ GeV and $|\eta| < 2$.
- **Pre-processing:** each jet has at most 200 4-vectors; these are *centered*, *rotated*, and *flipped* in both axes. Then pixelization occurs with a slight *crop*, yielding 40×40 images. Finally, the pixels are *normalized* to sum to one.
- Pixel size is $[\Delta\eta, \Delta\varphi] = [0.029{,}0.035]$.
- Train-test split: 75/25 for QCD, 0/100 Tops.



A random sample.

Task: Anomaly Detection

Is about **distinguishing anomalies** (e.g. Top jets) from normal data (e.g. QCD Jets):

- Idea: normal samples have either low error or high-likelihood.
- Anomalies can be either *erroneous*, *rare*, or *interesting* events.
- Can be solved either by: estimating data density, thresholding distances or errors, clustering, or classification.
- Our approach is *self-supervised* and assumes a normal-only (N) set of samples: QCD jets.

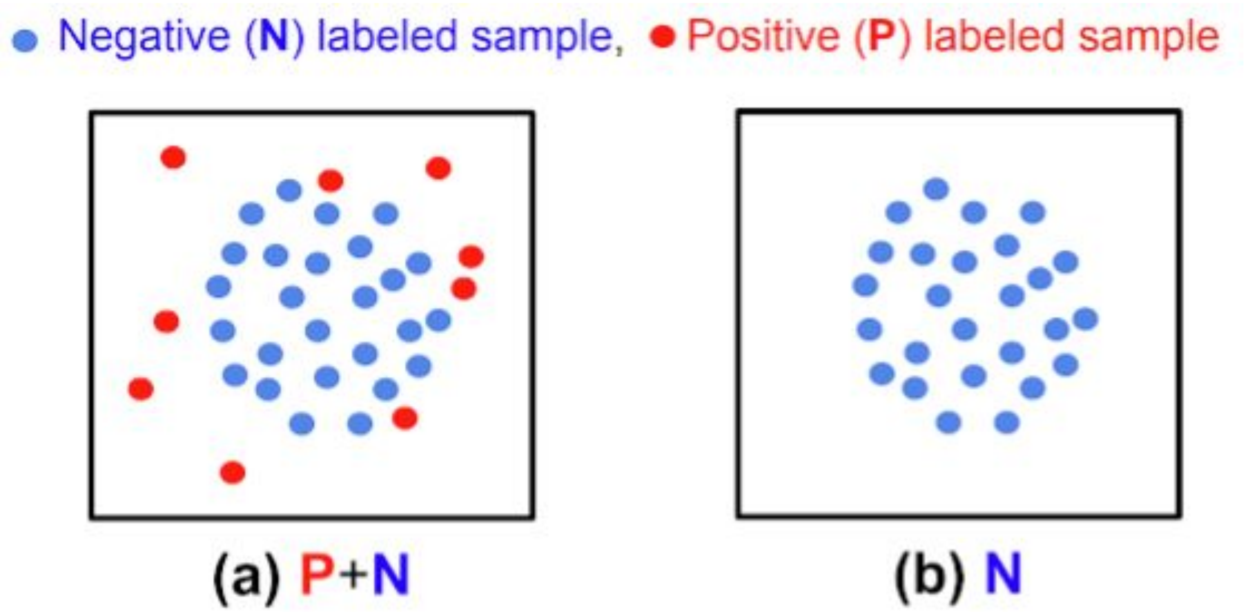
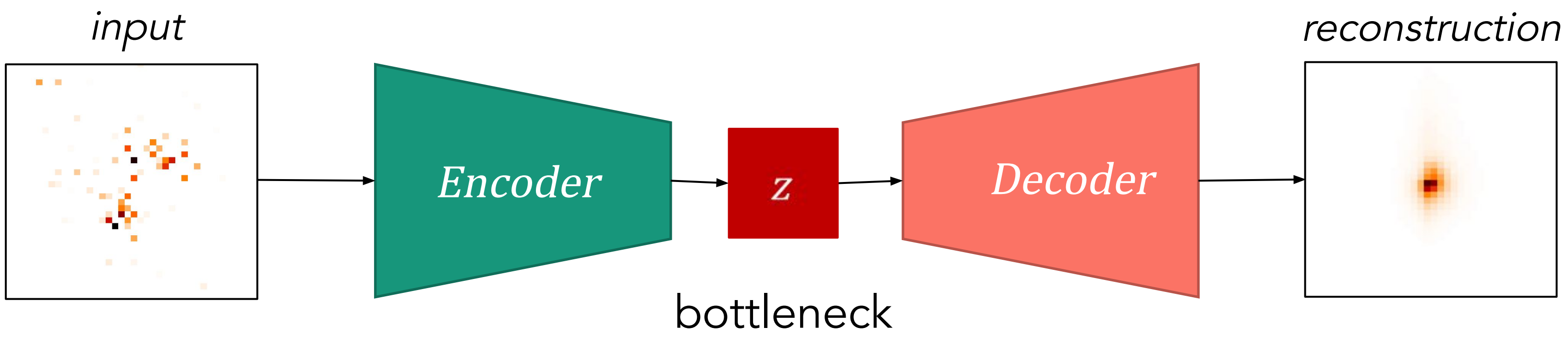


Figure from Google AI Blog

Method: Auto-Encoders

Auto-Encoders (AEs) are a neural network model trained to **reconstruct** its inputs:

- The **encoder** has to *compress* the input into a meaningful **latent space** Z (bottleneck)
- The **decoder** reconstructs from the compressed representation.



- AEs and variants are suitable for anomaly detection, since the training doesn't require labels!
- In context of AD, the AE have to capture the background peculiarities.

Joint-VAE for Anomaly Detection

Variational AE

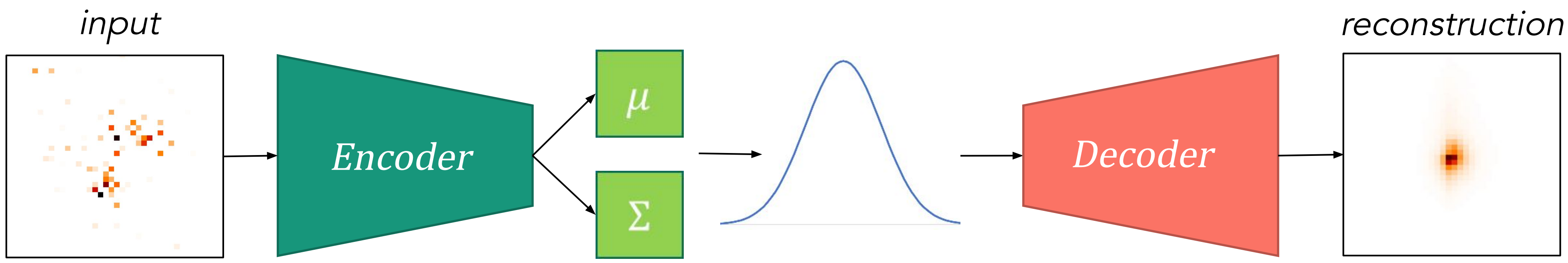
Discrete VAE

Joint-VAE

Variational Auto-Encoders

Variational Auto-Encoders (VAEs) are *probabilistic* models:

- The **encoder** q_ϕ, parameterizes a **Gaussian distribution** $z \sim N(\mu_\phi(x), \Sigma_\phi(x))$.
- The **decoder** p_θ, reconstructs from the sampled latent vectors.

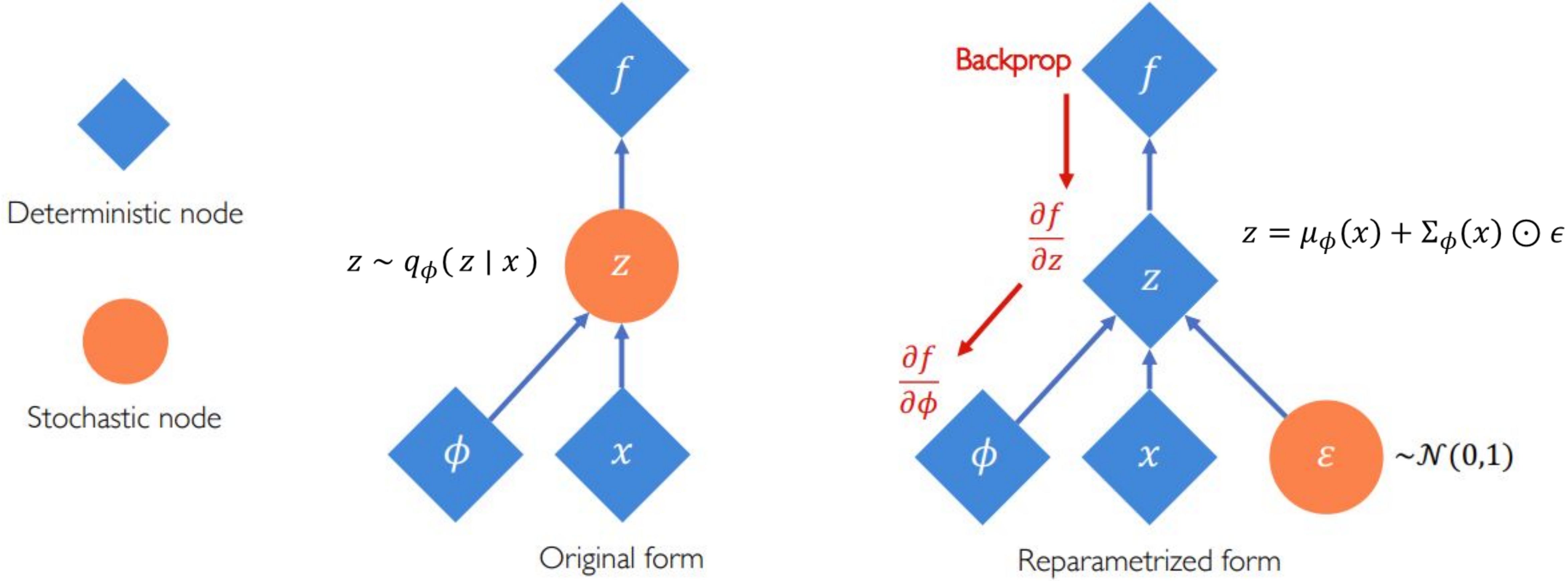


$$q_\phi(z \mid x) = N(\mu(x), \Sigma(x))$$

- The latent space encodes **continuous features**.
- VAEs can generate new samples that look like the inputs.

Reparameterization Trick

Issue: cannot backpropagate through stochastic (sampling) nodes



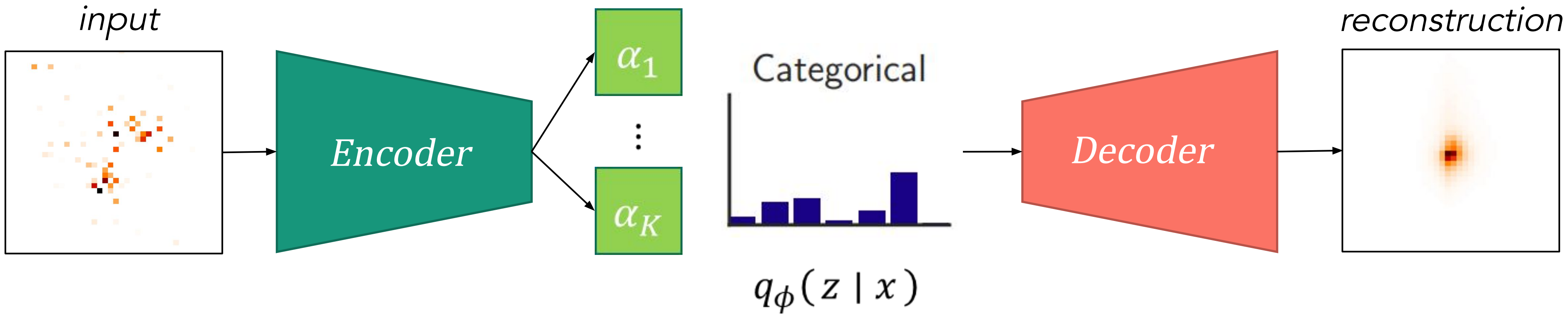
Categorical VAE

[Categorical Reparameterization with Gumbel-Softmax](#)

[The Concrete Distribution: continuous relaxation of discrete random variables](#)

Categorical VAEs are *discrete latent variable* models:

- The **encoder** q_ϕ, parameterizes up to K **Categorical distributions** with C classes each.
- The **decoder** p_θ, reconstructs from the sampled latent vectors.



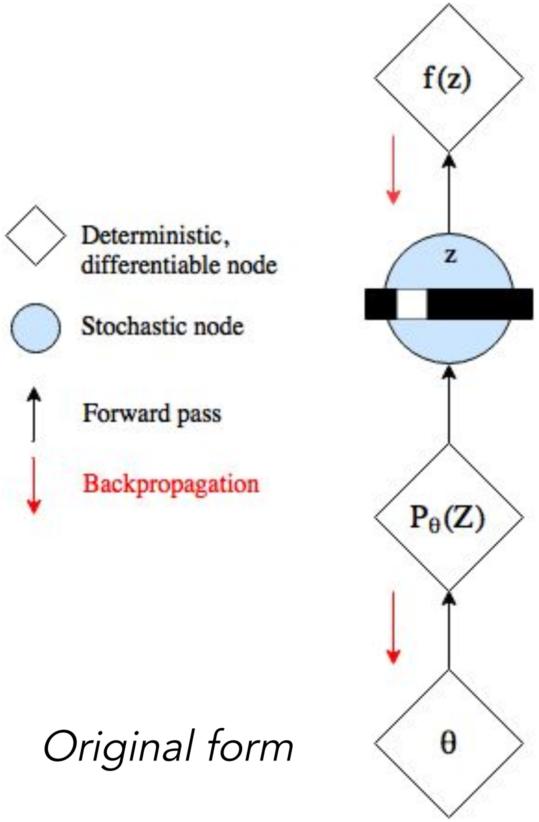
- The latent space encodes **discrete features**: finite and enumerable quantities, like counts.
- The Categorical is relaxed by a **temperature** parameter, τ: $\tau \to 0$ (categorical), $\tau \to \infty$ (uniform).

Gumbel-Softmax Trick

Concrete Reparameterization with Gumbel-Softmax

The Concrete Distribution: continuous relaxation of discrete random variables

Issue: the categorical distribution is not differentiable



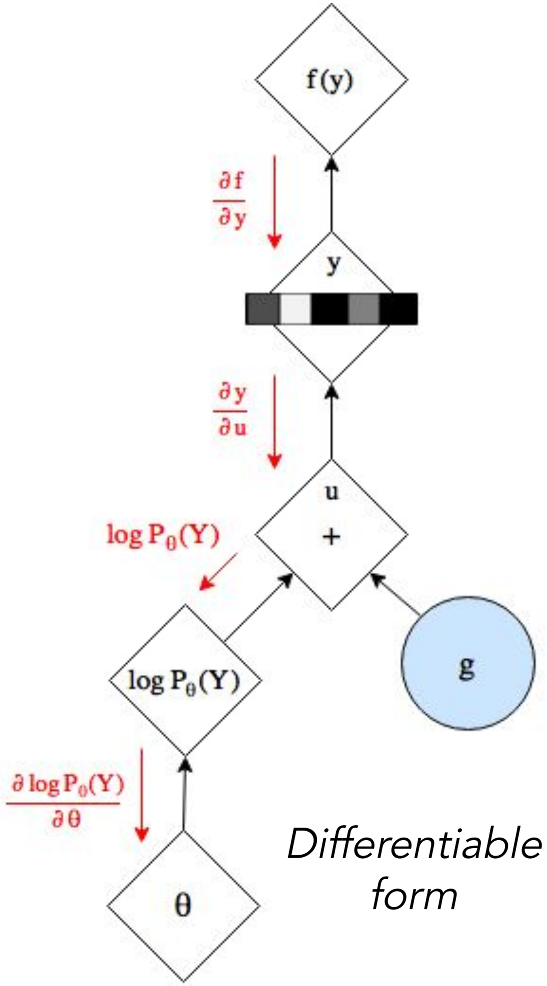
Original form

$\Rightarrow$

To reparameterize:

1. Forward encoder to get logits
$$\alpha = q_\phi(x)$$
2. Sample from Uniform distribution
$$u \sim U(0,1)$$
3. Compute Gumbel noise g
$$g = -\log(-\log u)$$
4. Get the Gumbel-Softmax samples
$$q = softmax(\frac{\alpha + g}{\tau})$$

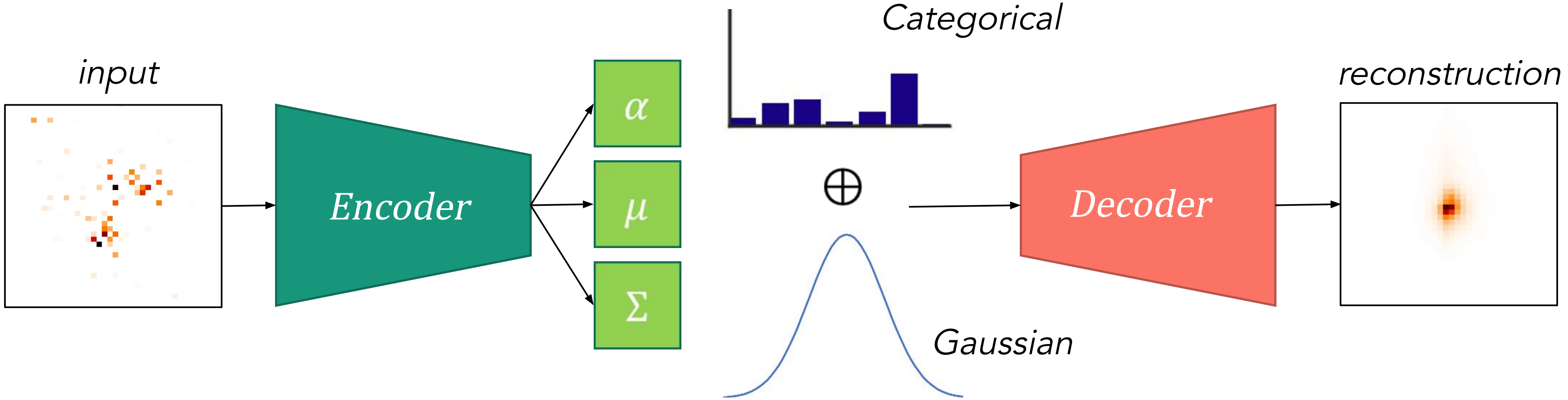
$\Rightarrow$



Differentiable form

Joint-VAE

Can encode both ***continuous*** and ***discrete*** features:



How the model is trained:

Priors $p(z)$

$$L_{\phi,\theta}(x) = L_{reco}(x,\hat{x}) + \beta \underbrace{D_{KL}\left(\boldsymbol{N}(\boldsymbol{\mu_\phi},\boldsymbol{\Sigma_\phi}) \mid N(0,I)\right)}_{\text{Continuous KL}} + \beta \underbrace{D_{KL}\left(\boldsymbol{Cat}(\boldsymbol{\alpha_\phi},\boldsymbol{\tau}) \mid Cat(1/C)\right)}_{\text{Discrete KL}}$$

Anomaly Detection

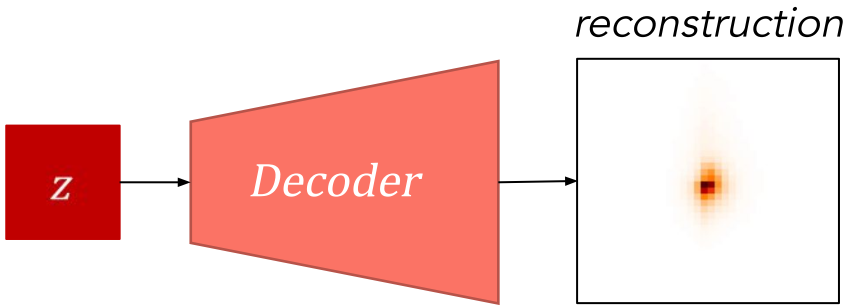
Reconstruction-based

Latent-based

Pros & Cons

Reconstruction-based Anomaly Scores

AD scores can be defined from **reconstructed images** x':



- **MSE**: sum of squared differences of pixel values P

$$S_{MSE}(x, x') = \sum_{p \in P} \left(x_p - x'_p\right)^2$$

- **BCE**: sum of binary-cross entropies on pixels (note: only if normalized in [0, 1])

$$S_{BCE}(x, x') = -\sum_{p \in P} x_p \log x'_p + \left(1 - x_p\right) \log(1 - x'_p)$$

- **Dice** (see ref.): measures the overlap between the predicted and original image

$$S_{Dice}(x, x') = \frac{\sum_{p \in P} x_p^2 + \sum_{p \in P} {x'_p}^2}{2 \sum_{p \in P} x_p \cdot x'_p}$$

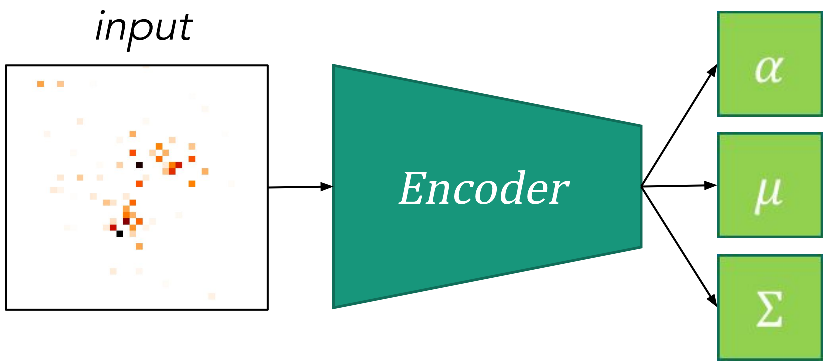
- **PixelDiff**: difference of pixel sums (*)

$$S_{diff}(x, x') = 1 - \sum_{p \in P} x'_p$$

*True image sum to one.

Latent-based Anomaly Scores

AD scores defined from the ***joint* latent space** $z = (\alpha, \mu, \Sigma)$:



- **Idea**: KL divergence between learned distribution and prior!
- **KL Continuous**: divergence between learned Normal $N(\mu, \Sigma)$ and standard Normal prior $N(0, I)$

$$S_{KL,cont}(\mu, \Sigma) = -\frac{1}{2}\sum_i (1 + \log \Sigma_{ii} - \mu_i^2 - \exp \Sigma_{ii})$$

- **KL Discrete**: divergence between relaxed Categorical $Cat(\alpha, \tau)$ and the *uniform* Gumbel-Softmax prior $Cat(1/C)$ - where C is the number of classes.

$$S_{KL,disc}(\alpha) = \sum_i (\pi_i \log \pi_i) - (\pi_i \log 1/C) \qquad \text{where } \pi = softmax(\alpha)$$

*Sums are over latent dimensions.

Discussion: Pros & Cons

Reconstruction-based AD:

- Easier to define anomaly scores, e.g. from common loss functions and metrics.
- Scores values can be interpreted by image quality metrics or visual inspection.
- Requires forward pass of whole model (encoder + decoder): slower.

Latent-based AD:

- Possibly difficult to interpret: high-dim latent space cannot be visualized.
- Scores can be difficult to design, e.g. analytical KLD – but equally performant.
- Faster: requires only encoder predictions.
- Suitable for model optimization and FPGA deployment.

Model Acceleration

Quantization

FPGA Study

Compression: Quantization with QKeras

Quantization transforms floating-point arithmetic to fixed-point precision:

- Less #bits to reduce memory footprint, and FPGA resources.
- Quantization is applied on both *weights*, and *activations*.
- Quantization-Aware-Training (QAT) maintains high accuracy at low-precision <16, 6>: total with of 16bits, 10bits for floats and 6bits for integers.
- Yields lower latency and energy consumption [J] (by QTools).

Model	**Total Energy** $[\mu J]$
Reduced Encoder	5.4957
Quantized Encoder	3.3434

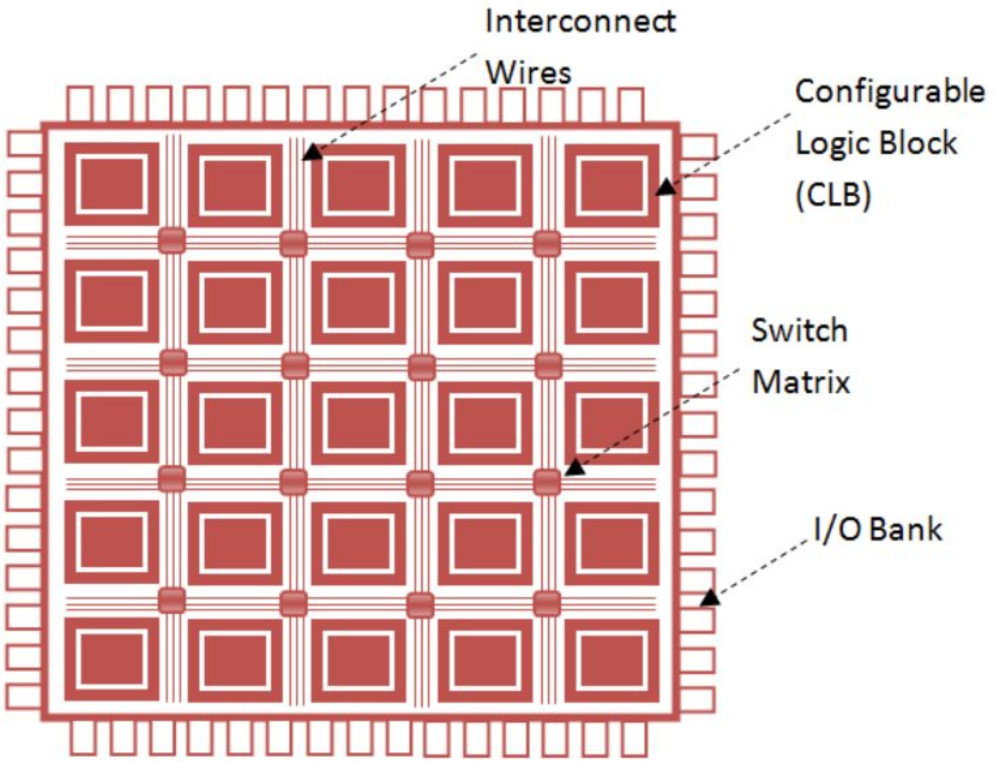
Energy consumption reduced by 39%

Not quantized:

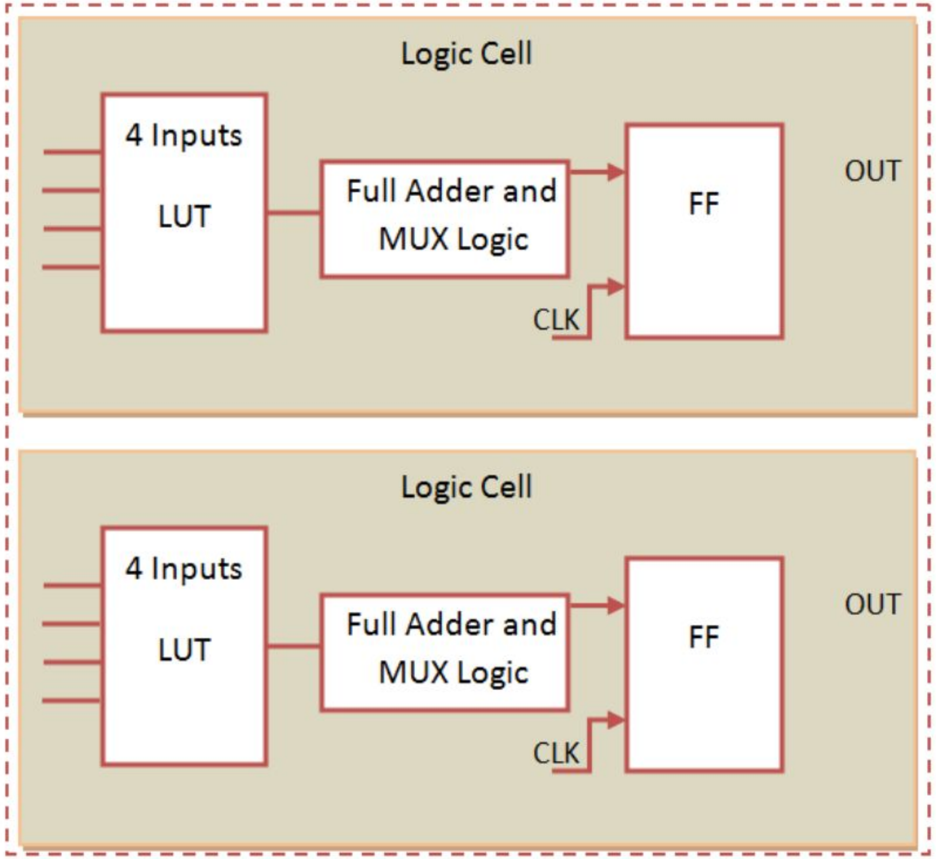
Layer (Type)	**Energy [nJ]**
dconv_b0 (Conv2D)	87.9
conv1_b0_0 (Conv2D)	1382.4
conv2_b0_0 (Conv2D)	1382.4
dconv_b1 (Conv2D)	432.0
conv1_b1_0 (Conv2D)	540.0
conv2_b1_0 (Conv2D)	540.0
conv1_b1_1 (Conv2D)	540.0
conv2_b1_1 (Conv2D)	540.0
dconv_b2 (Conv2D)	27.0
conv1_b2_0 (Conv2D)	5.4
conv2_b2_0 (Conv2D)	5.4
conv_fin (Conv2D)	2.7
z_categorical (Dense)	1.5
z_mean (Dense)	2.4
z_var (Dense)	2.4

Field-Programmable Gate Arrays

FPGAs are hardware-programmable devices:



An FPGA is made of many replicated units.

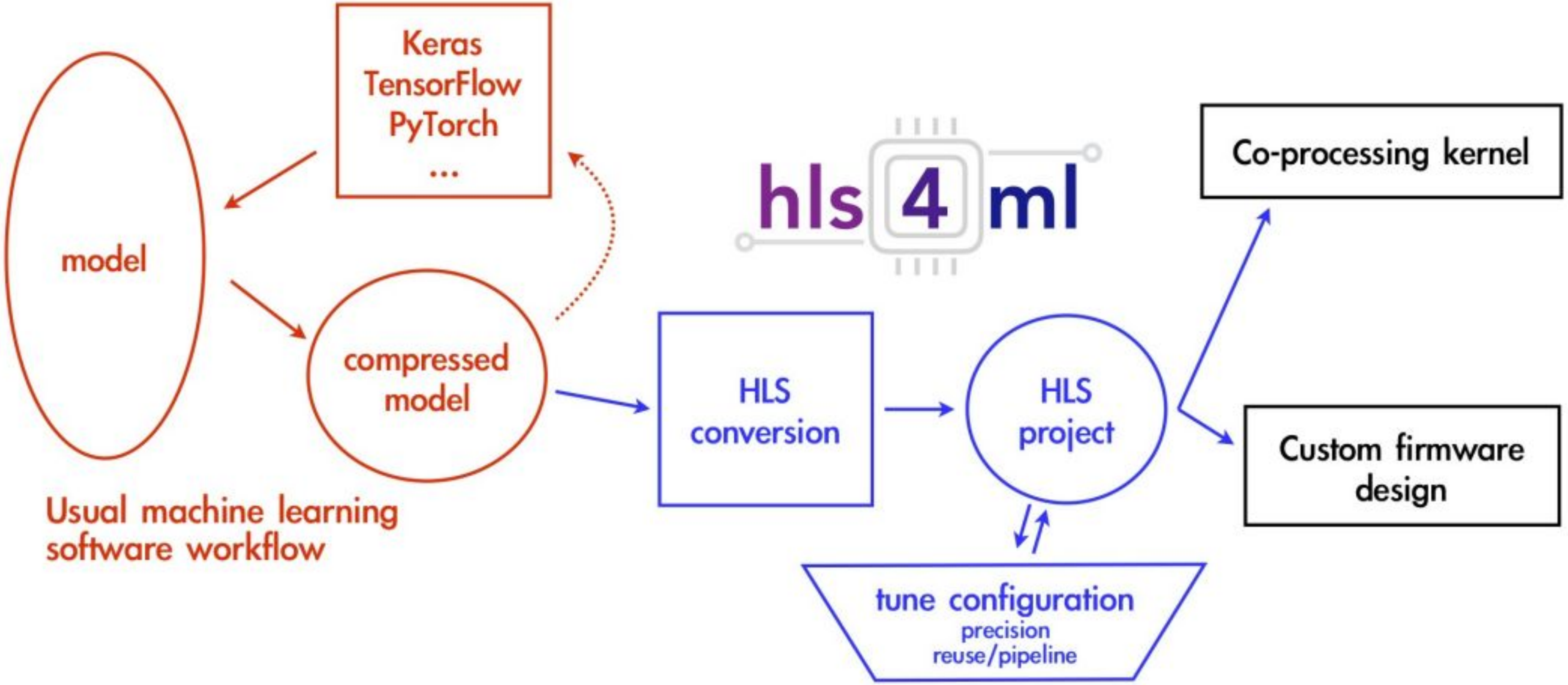


A configurable logic block.

The **HLS4ML** Python Package

ML models have to be translated to Hardware Description Language (HDL) to deploy on FPGA:

- HLS4ML does this.
- Converts layers to High Level Synthesis code, then C++.
- It optimizes also.
- Finally, proprietary SW compilation.
- Synthesized code can be simulated before deployment.



FPGA Implementation Feasibility Study

FPGA are programmable accelerators that can enable **real-time inference**:

- Network synthesis is done via HLS4ML toolkit.
- From a synthesized layer or block, we can estimate the **resource factor** ρ via:

Formula based on convolution computational complexity from [ref]: $O(k^2(hw)/sd_{in}d_{out})$

$$\rho(a,b) = \frac{(h^a_{in}/s) \times (w^a_{in}/s) \times d^a_{in} \times d^a_{out}}{(h^b_{in}/s) \times (w^b_{in}/s) \times d^b_{in} \times d^b_{out}}$$

AMD/Xilinx Alveo U250

- To **estimate** FPGA resources (e.g., LUTs) multiply ρ by a known layer or block:

conv2_b0_0 won't fit in FPGA, so we estimated its resource consumption.

Layer Name	$h_{in}, w_{in}, d_{in}, d_{out}$	**DSP**(%)	**LUT**(%)	**FF** (%)	**BRAM**(%)
conv2_b0_0	20,20,16,10	*752(6)*	*6189696(**358**)*	*229536(∼7)*	*288(5)*
conv2_b1_1	10,10,20,5	47(∼0)	386856(22)	14346(∼0)	18(∼0)
dconv_b2	5, 5, 4,4	15(∼0)	309544(17)	9352(∼0)	4(∼0)
final block	5, 5, 4,2	851(6)	91763(5)	28074(∼0)	377(7)

Results

Reconstructions

Anomaly Performance

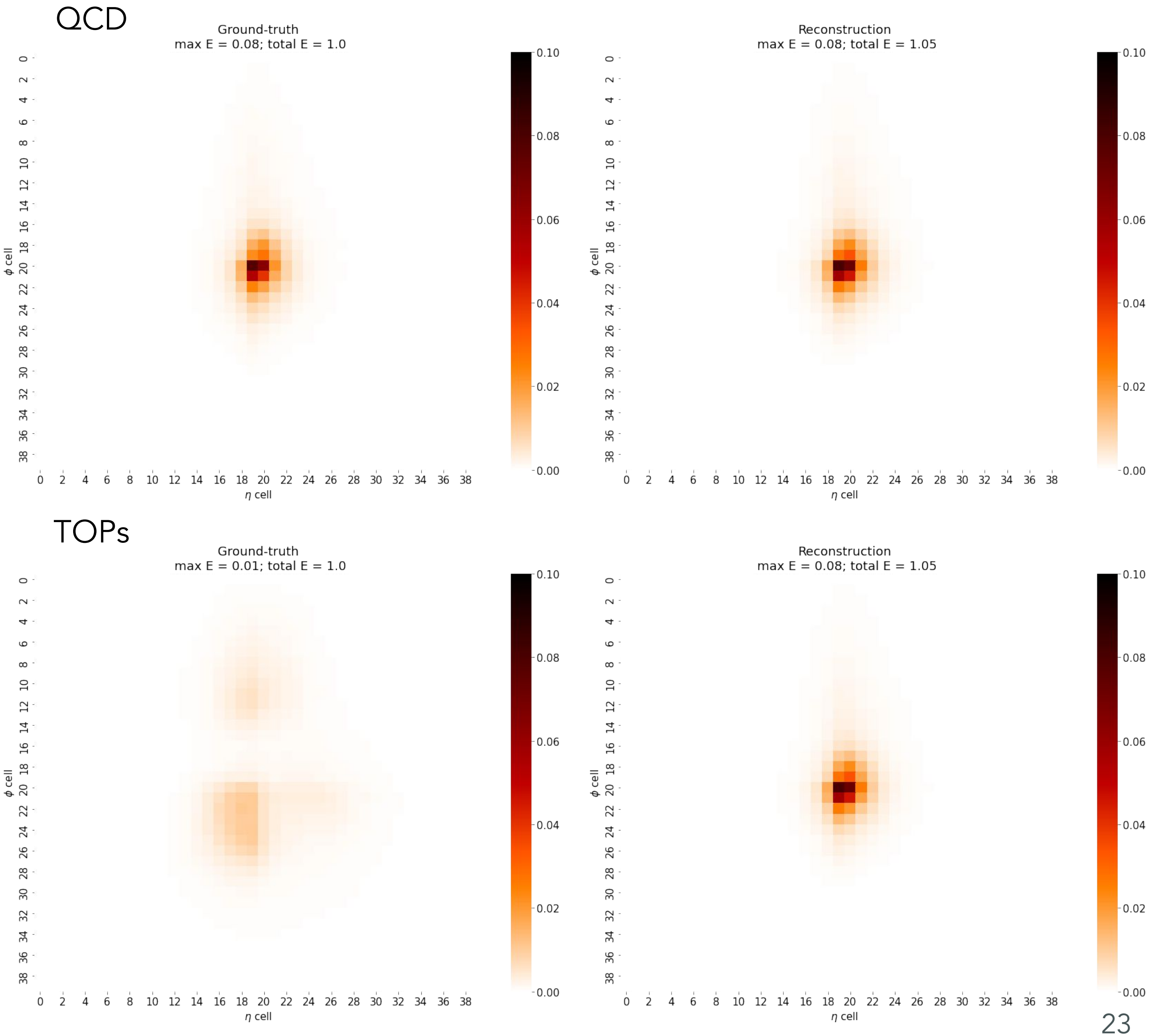
Comparison: large vs quantized

Reconstructed Samples

Reconstructed images *averaged* over test-set.

The ground-truth is on the left.

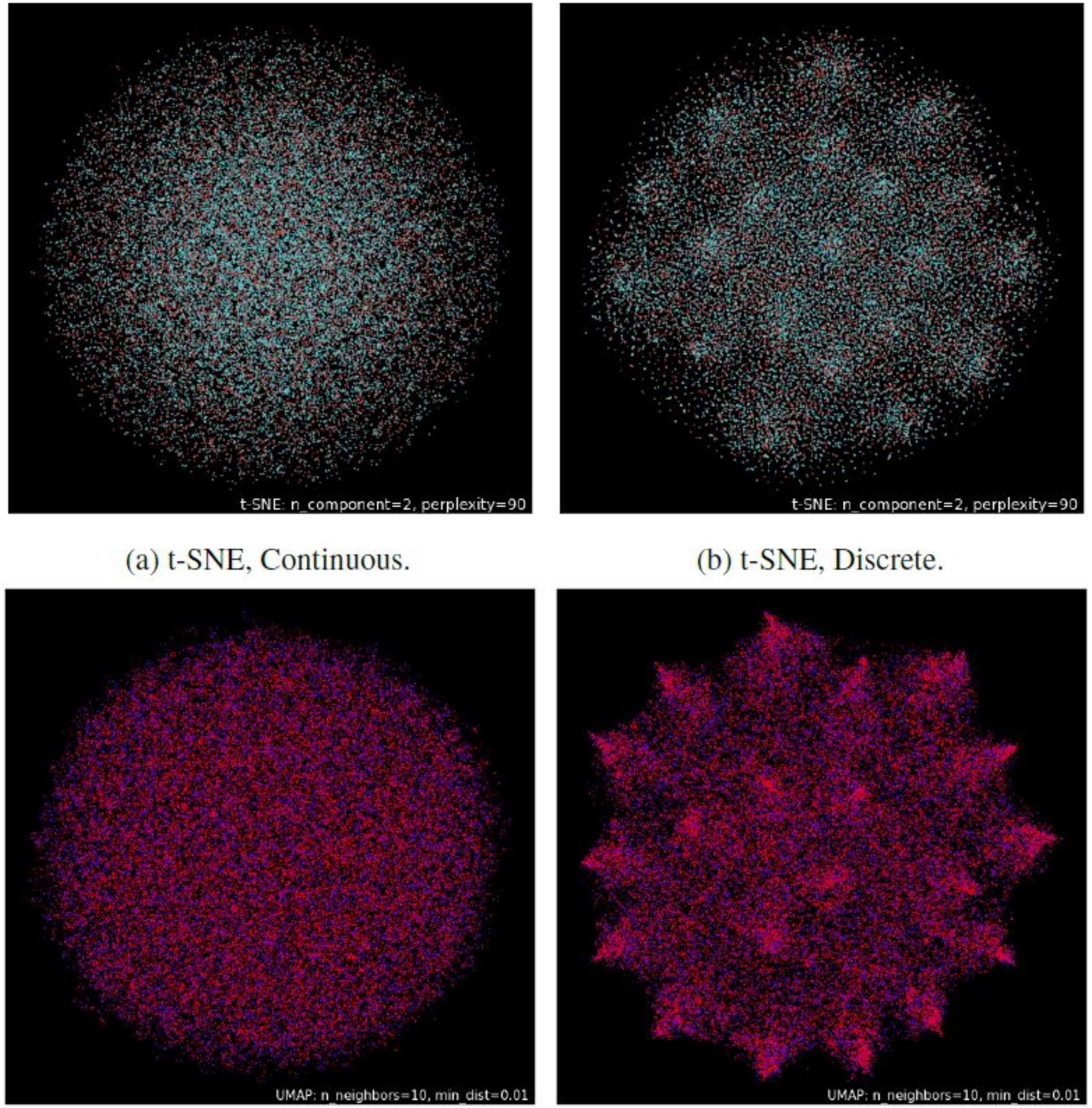
- QCD (top-row) are closely reconstructed: *low error*.
- Tops (bottom) are predicted to be QCD-like: *high error*.



Joint Latent Space

Learned latent spaces by our Joint-VAE; projected to 2d.

- Spaces: a 32-d Gaussian, and 20-d Categorical.
- We can see the 20 class-clusters for the categorical space.



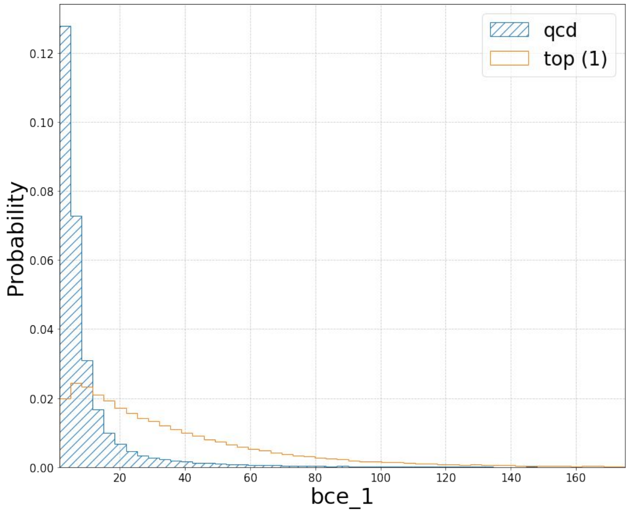
(a) t-SNE, Continuous.

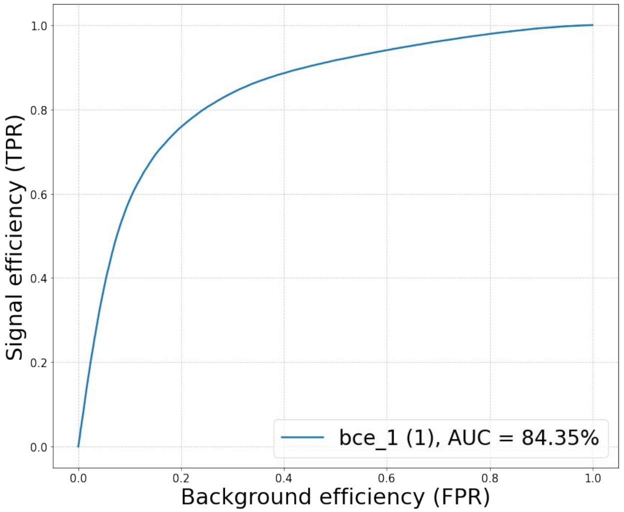
(b) t-SNE, Discrete.

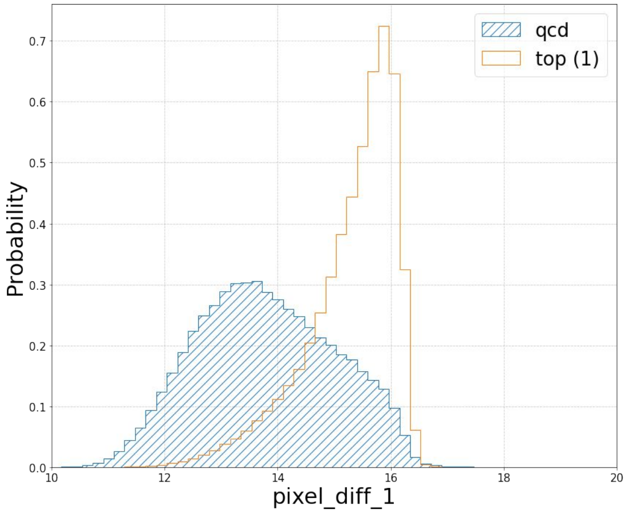
(c) UMAP, Continuous.

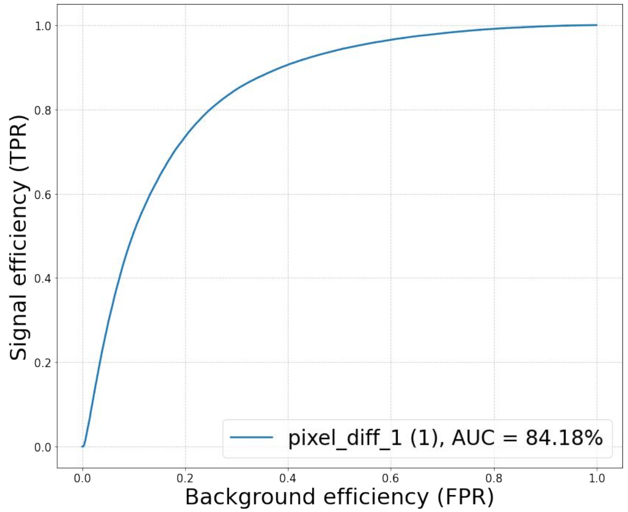
(d) UMAP, Discrete.

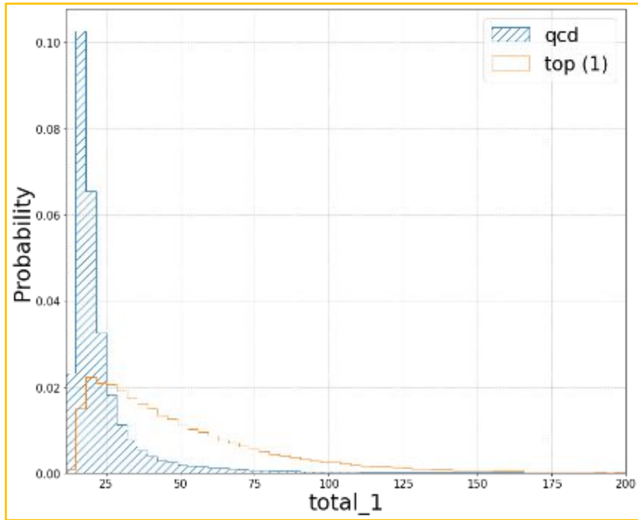
Reconstruction-based Scores: Large Model

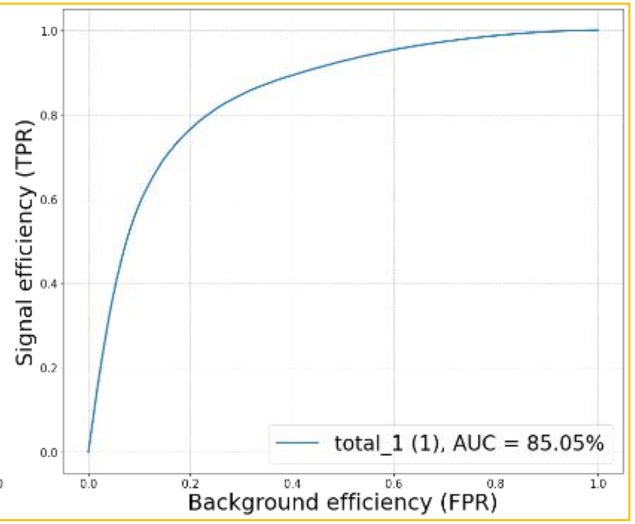


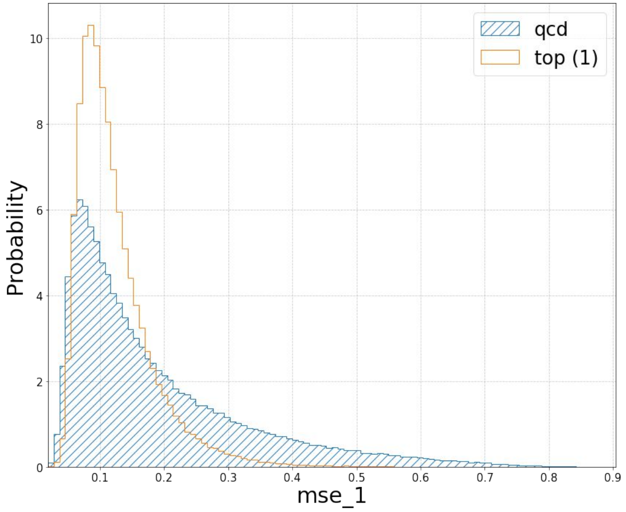


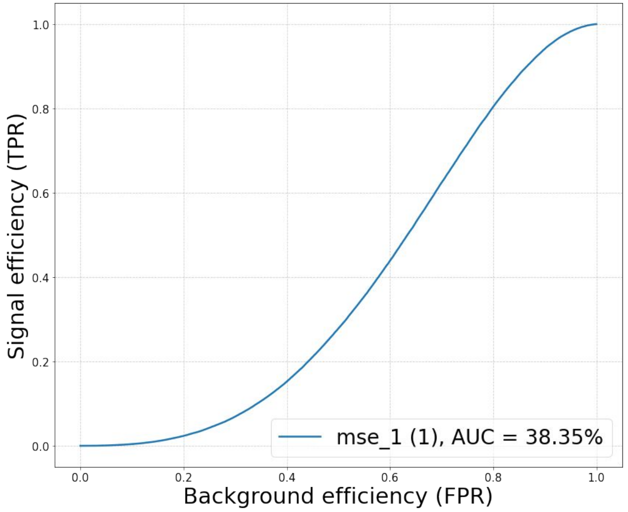






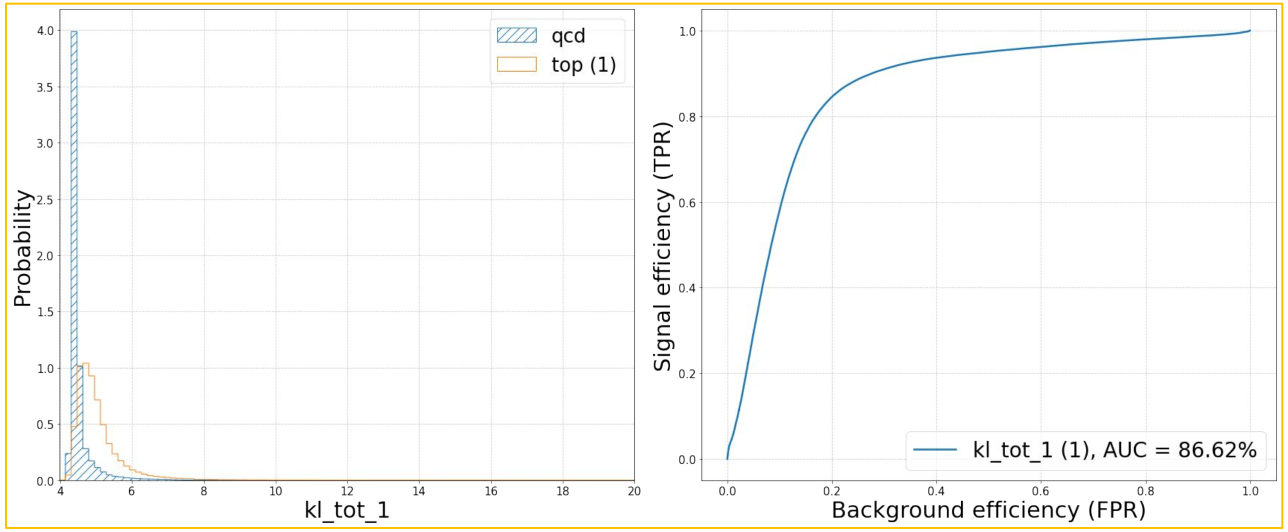


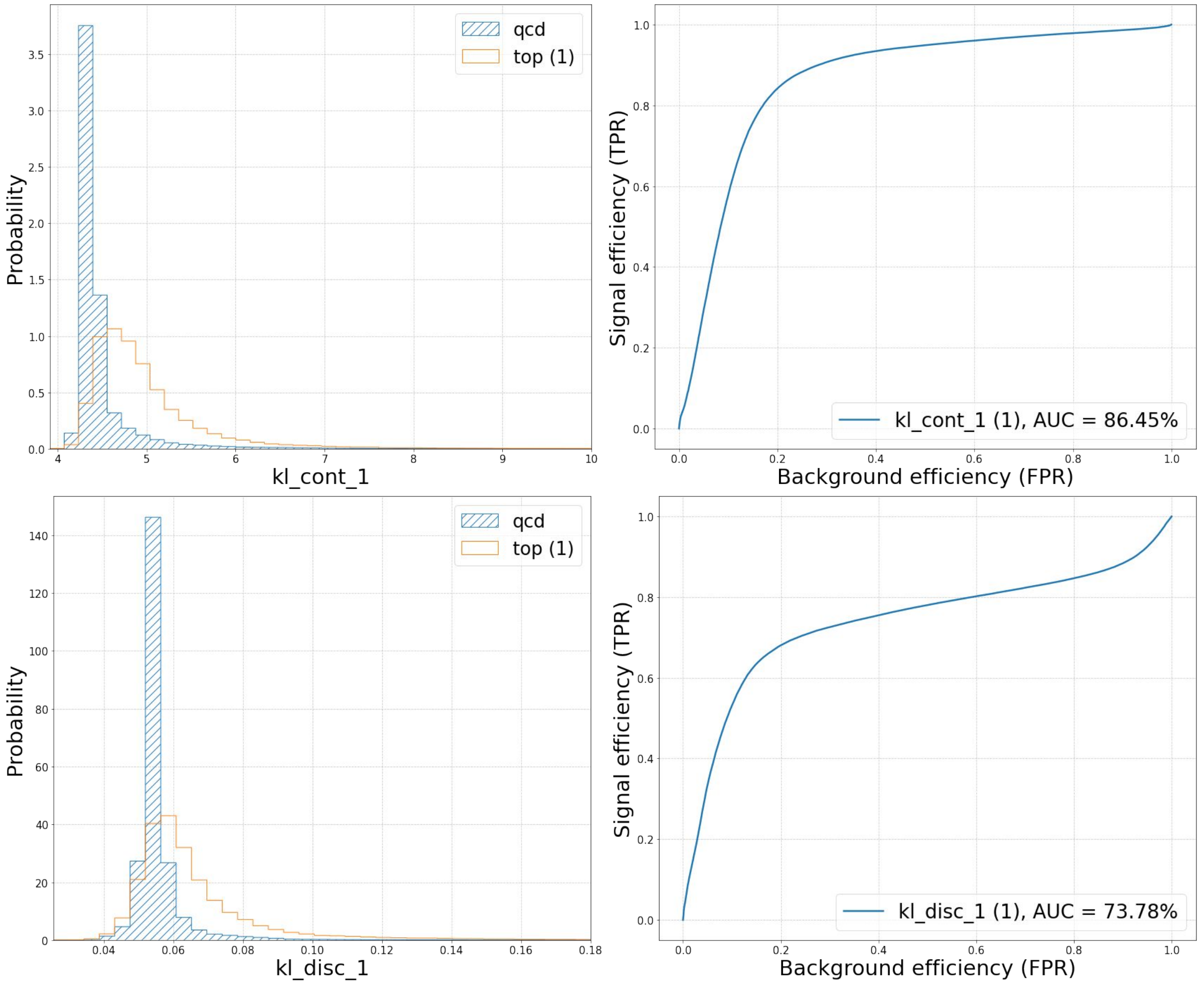




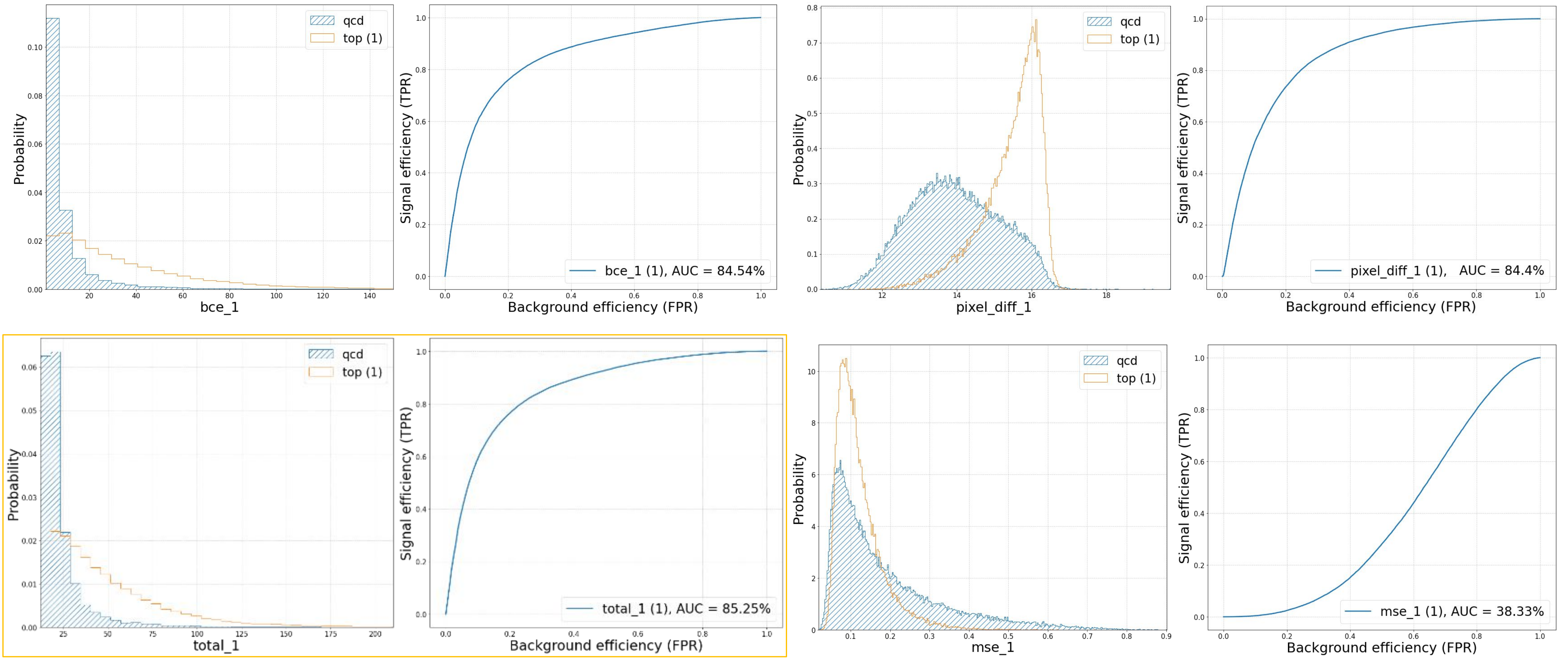
Latent-based Scores: Large Model

By combining both continuous and discrete KL divergences, is possible to further improve the anomaly score.





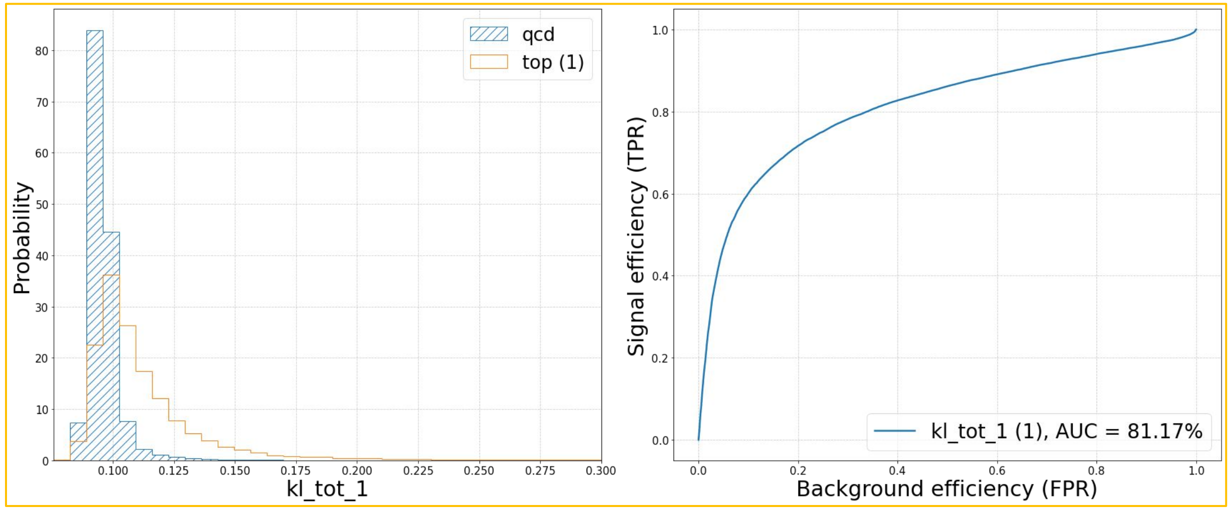
Reconstruction-based Scores: *Quantized* Model

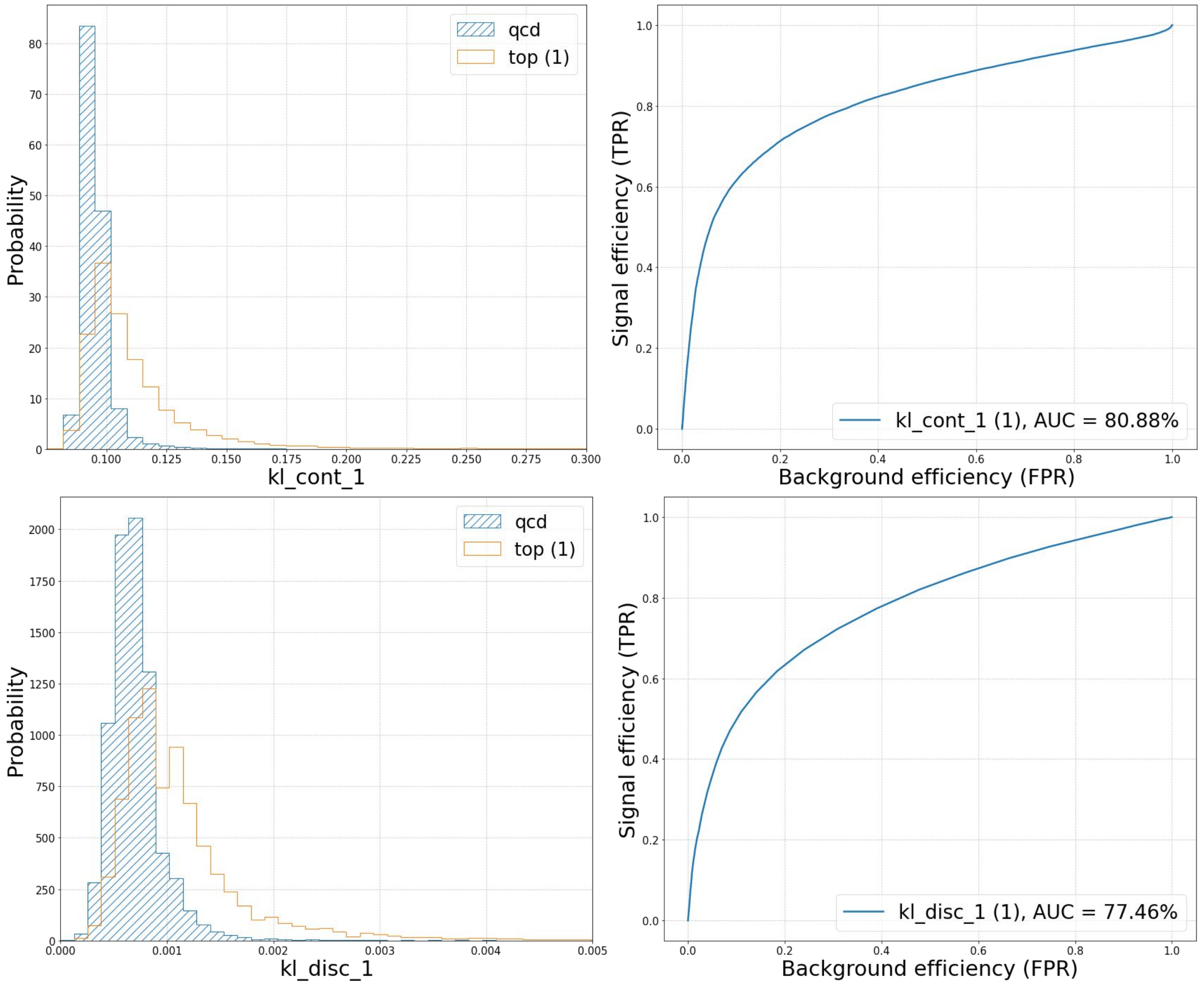


Latent-based Scores: *Quantized* Model

By quantizing we lose performance also on the latent space, so the KL scores.

But the trend is maintained.





Comparison: Large vs Quantized Model

Summary of ROC-AUC performance per metric:

- **Large model** has 262k (encoder) + 545k (decoder) params: 6 residual blocks.
- **Quantized model** has 10k (**constraint***: max. 1024 params per layer) + 545k params: 4 residual blocks.
- Latent dimensions for both models are 32 (continuous) and 20 (discrete).
- Decoder is the same ⇒ similar AD performance.
- #params and quantization impacts on encoder, KL-based metrics.

*Constraint is due to Vivado synthesis.

AD Score	**Large**	**Quantized**
MSE	38,35%	38,33%
Pixel-diff	84,18%	84,4%
BCE	84,35%	84,54%
Total (Dice + BCE)	85,05%	**85,25%**
KL Cont.	86,45%	80,88%
KL Discrete	73,78%	77,46%
KL Total (Cont. + Disc.)	**86,62%**	81,17%

Conclusions

Summary

Limitations

Outlook

Summary

Variational Auto-Encoders are suitable models for anomaly detection:

- We don't assume any specific signal ⇒ *not sensitive to particular BSM scenario*.
- The model is only trained to reconstruct the QCD background.
- Combining both continuous and discrete latent spaces achieves better AD performance.
- Latent-based AD is competitive with reconstruction-based scores,
  allowing to deploy only the encoder model.
- Model compression via weight and activation quantization can be done with Qkeras: saving energy, memory, and accelerator resources.
- Model synthesis for FPGA deployment can be done by HLS4ML.

Limitations and Outlook

General limitations of such kind of approaches:

- Need test samples of different kind of signals to asses generalization to BSM models.
- The VAE method is simple to train, but optimizes a different objective (i.e. reconstruction loss) ⇒ we have little control about maximizing the target AD score (e.g. KL-divergence)
- FPGA deployment can be challenging: accelerator resources are limited while DL layers are costly (like convolutions.), especially on image-like inputs.
- Moreover, vendors can add additional constraints: like maximum #params per layer.
- Limited support of libraries: for example HLS4ML is compatible with few common layers.
- Need better methods that yield very compact models: knowledge distillation?

github.com/LorenzoValente3/JointVAE4AD

Thanks for the Attention!

Questions?

Contacts:

lorenzo.valente3@studio.unibo.it

luca.anzalone2@unibo.it

marco.lorusso11@unibo.it

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA