Event Generation and Statistical Sampling with Deep Generative Models

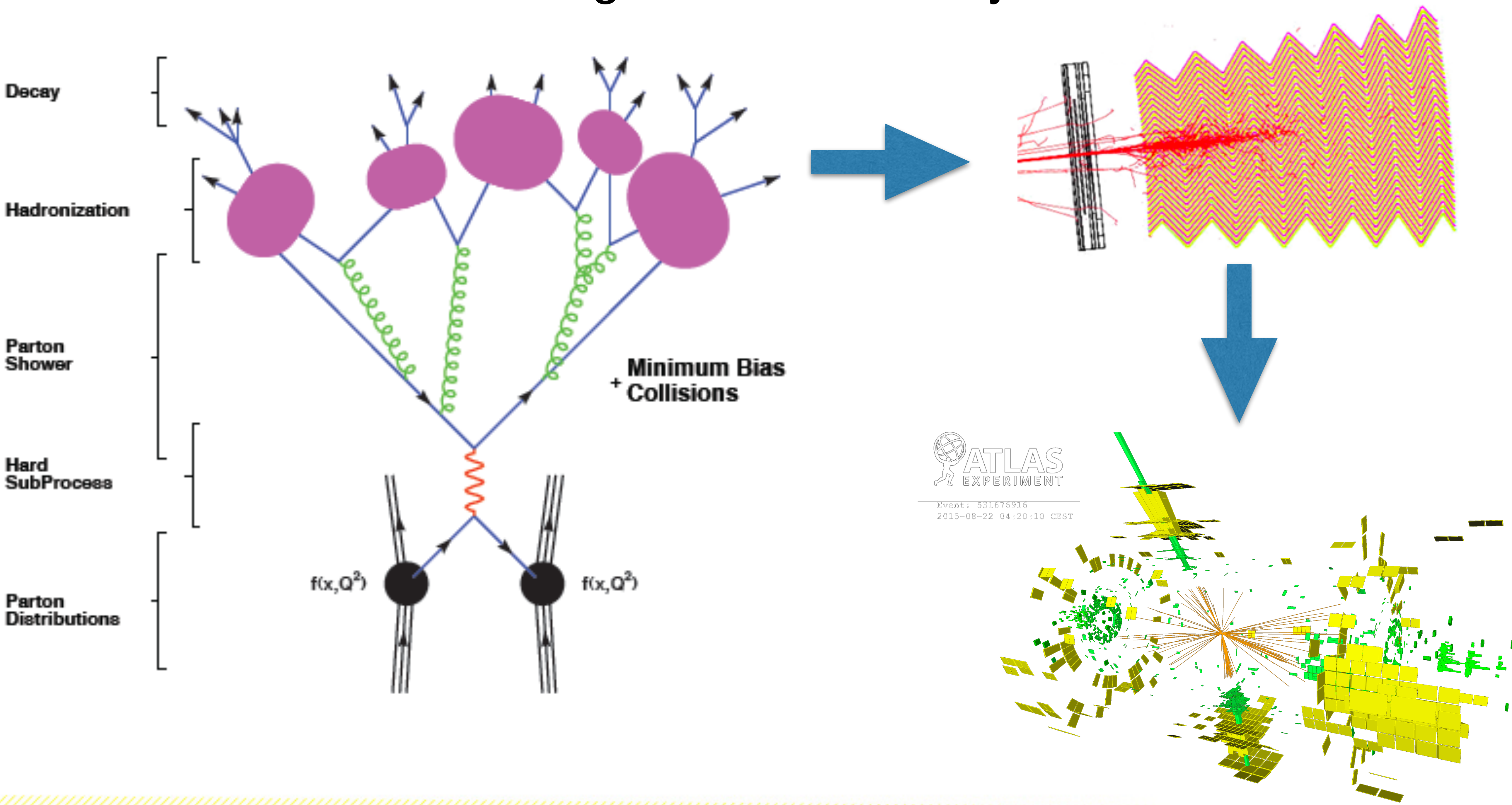
Rob Verheyen

Sydney Otten,[1,2,*] Sascha Caron,[1,3,†] Wieske de Swart,[1] Melissa van Beekveld,[1] Luc Hendriks,[1] Caspar van Leeuwen,[4] Damian Podareanu,[4] Roberto Ruiz de Austri,[5] and Rob Verheyen[1]

[1] *Institute for Mathematics, Astro- and Particle Physics IMAPP Radboud Universiteit, Nijmegen, The Netherlands*
[2] *GRAPPA, University of Amsterdam, The Netherlands*
[3] *Nikhef, Amsterdam, The Netherlands*
[4] *SURFsara, Amsterdam, The Netherlands*
[5] *Instituto de Fisica Corpuscular, IFIC-UV/CSIC University of Valencia, Spain*

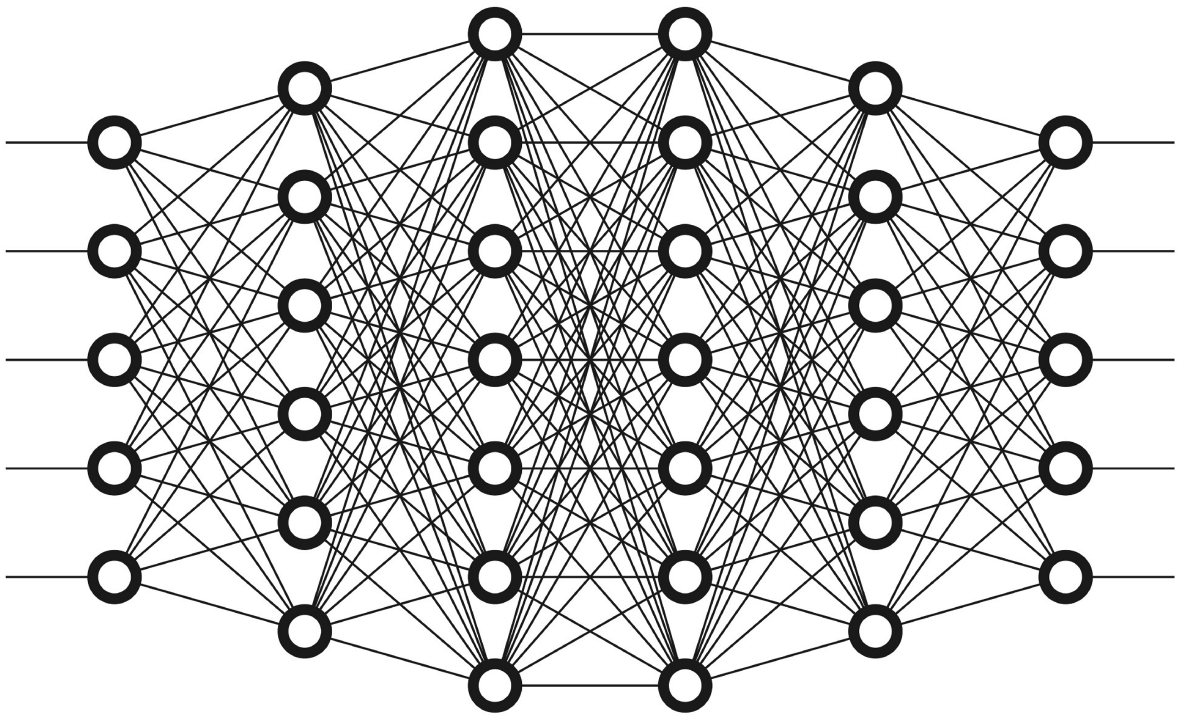
Introduction

Event generation is really hard!



Introduction

Can we use deep neural networks to do event generation?



Possible applications:

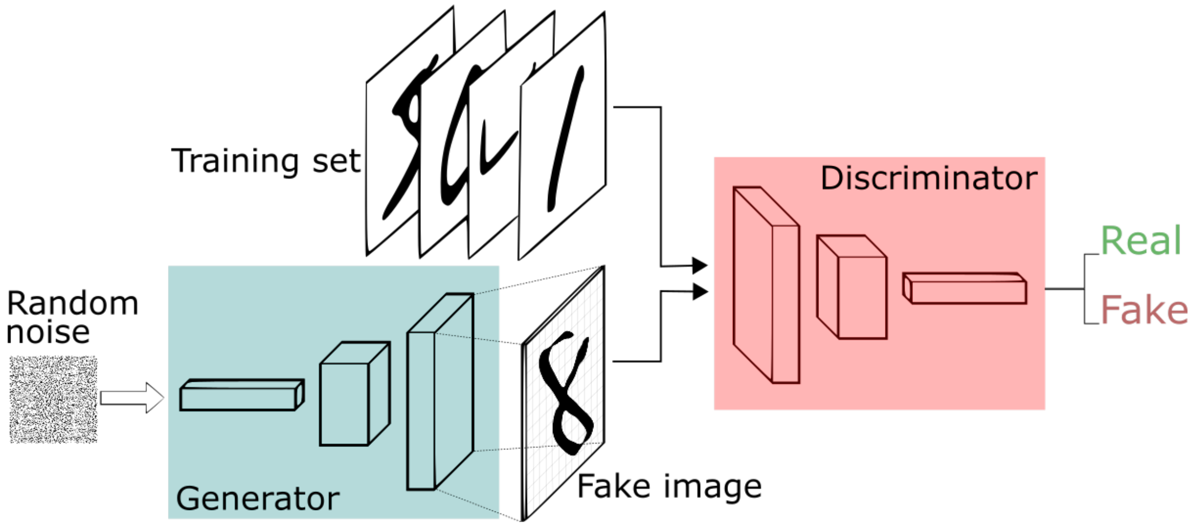
- Faster
- Data driven generators
- Targeted event generation

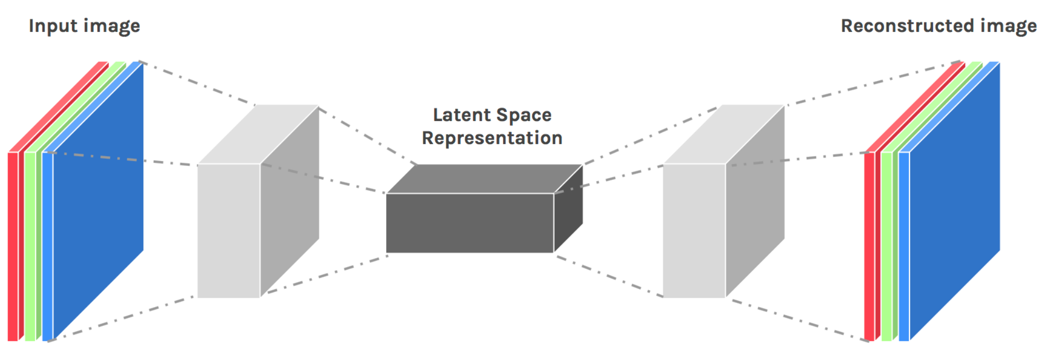
Introduction

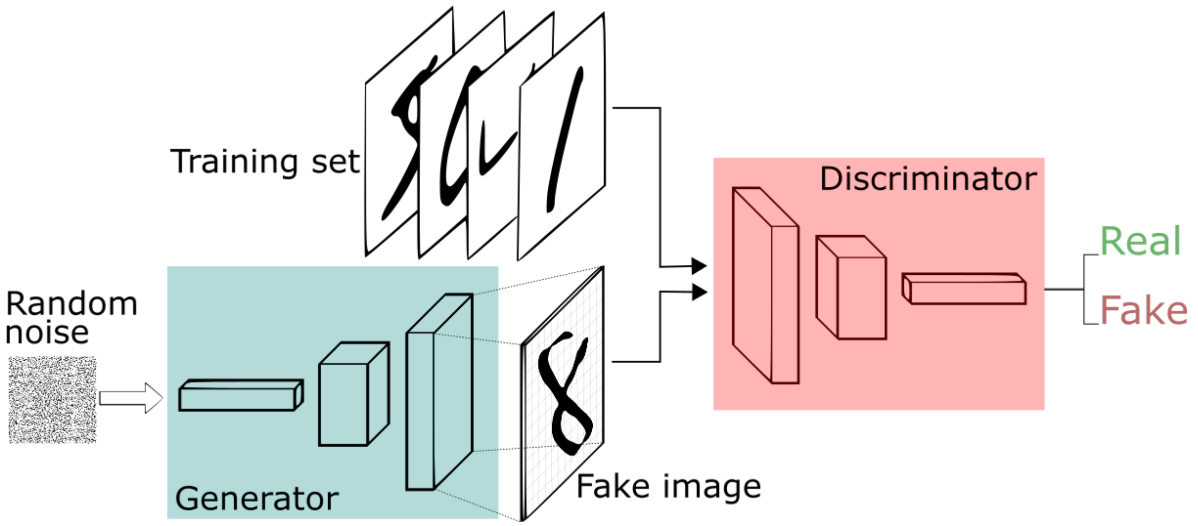
Study of different types of unsupervised generative models

- Generative Adversarial Networks
- Variational Autoencoders
- Buffer Variational Autoencoder

Can these networks be used to sample probability distributions?



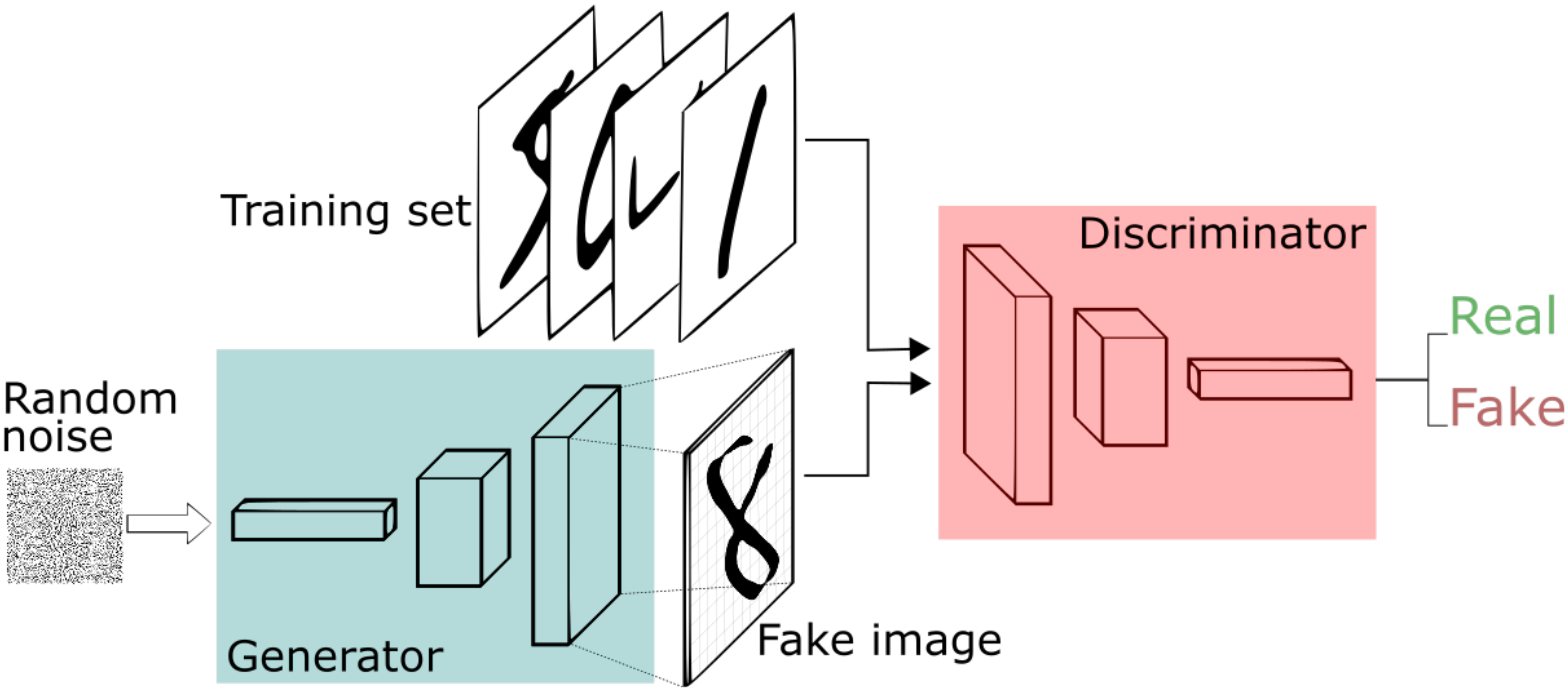




Generative Adversarial Networks (GANs)

Generative Adversarial Networks

Two networks (Generator & Discriminator) that play a game against each other



Generative Adversarial Networks

Loss function:

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim p_{data}(x)} \log[D(x)] + \mathbb{E}_{z \sim p_z(z)} \log[1 - D(G(z))]$$

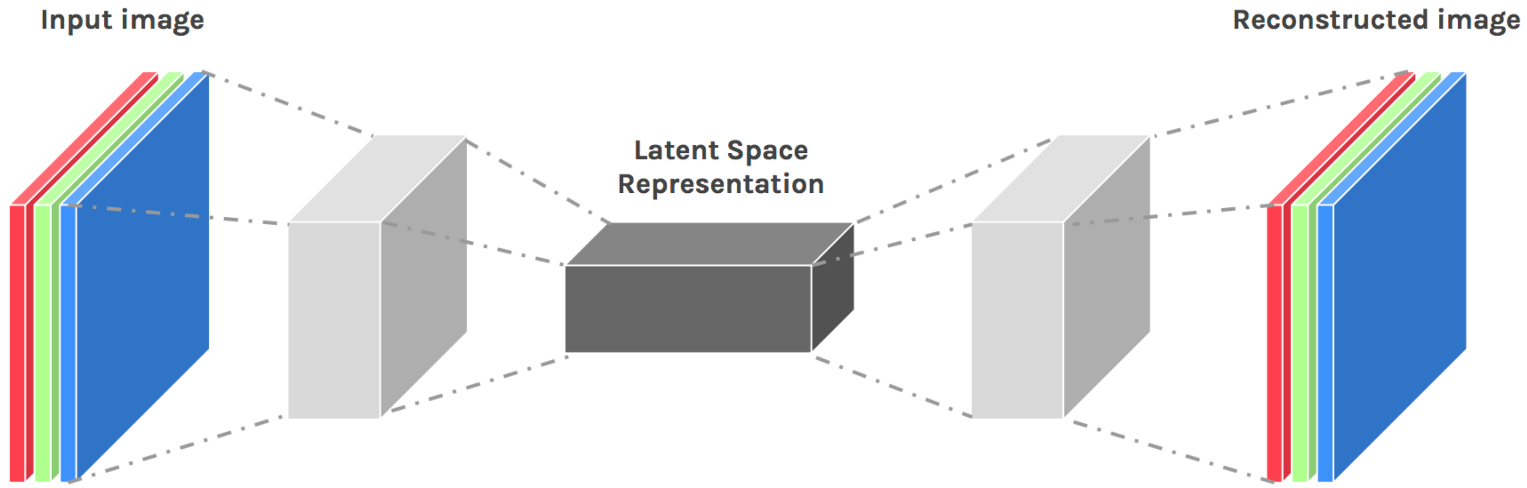
Nash equilibrium:

$$p_{data}(x) = p_{gen}(x)$$

$$D(x) = \frac{1}{2}$$

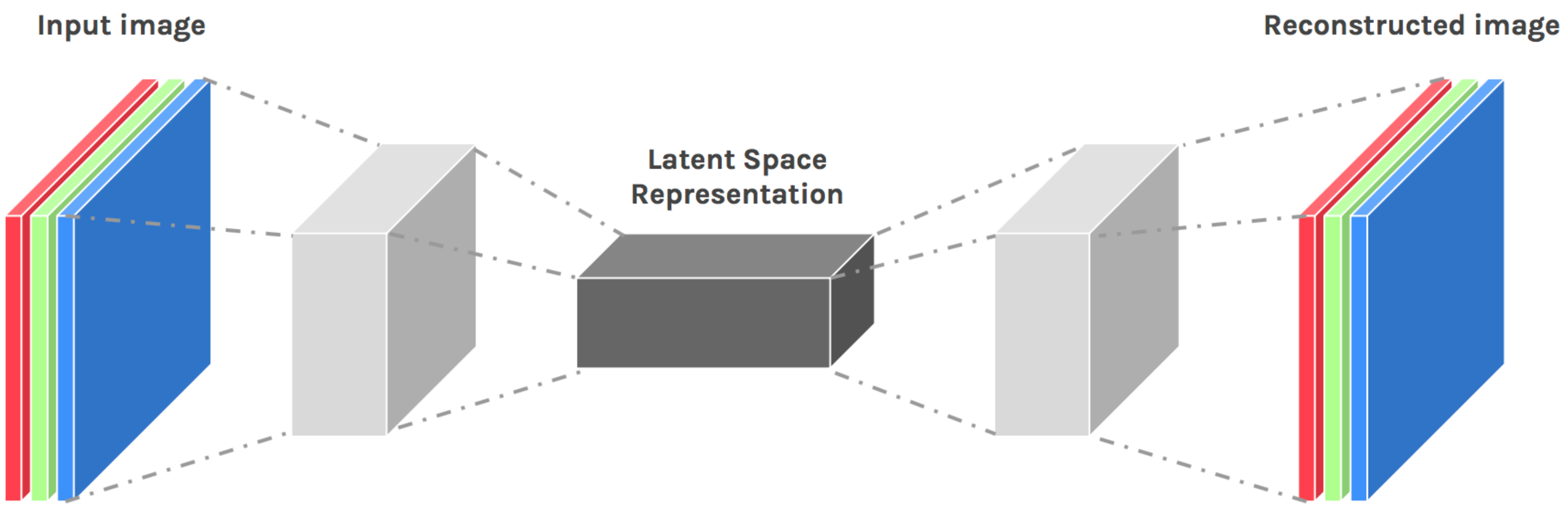
Generative Adversarial Networks

1812.04948



Variational Autoencoders (VAEs)

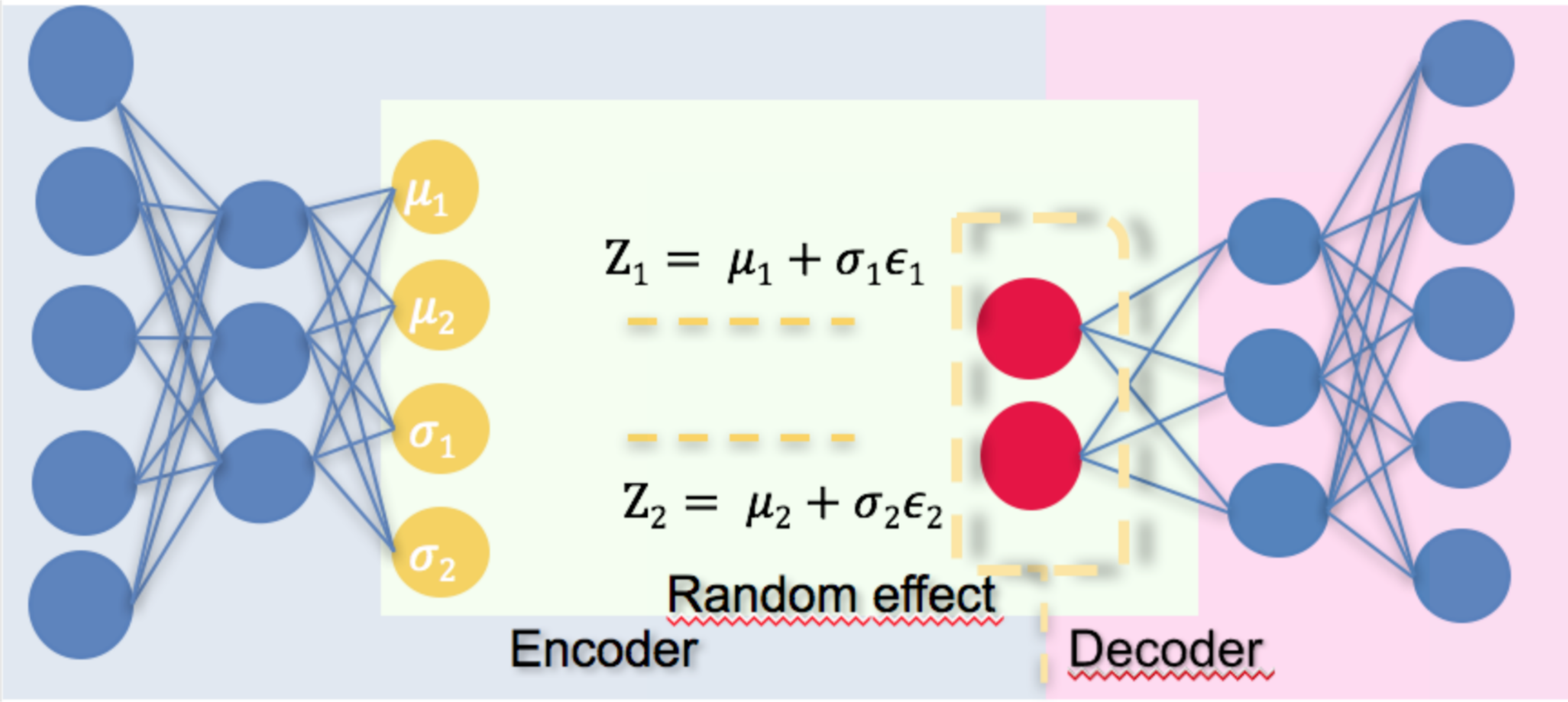
Autoencoders



- Data is encoded into latent space
- Dim of latent space is often lower than dim of data

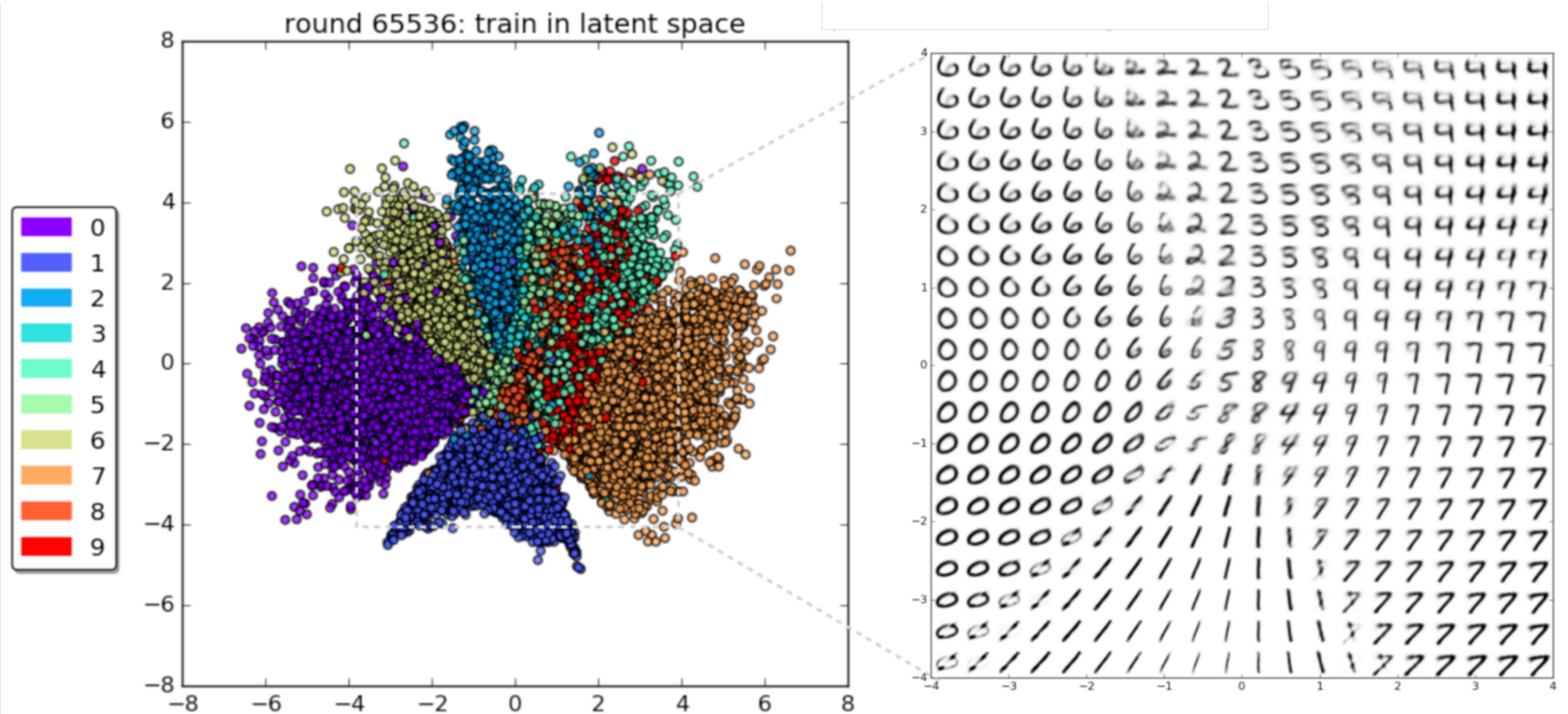
Variational Autoencoders

Add degree of randomness to training procedure



Variational Autoencoders

Points in latent space are ordered



Radboud University

Variational Autoencoders

Loss function

$$\mathcal{L}_{\mathrm{VAE}} = (1-\beta)\frac{1}{N}(\vec{x}_i - \vec{y}_i)^2 + \beta D_{\mathrm{KL}}(\mathcal{N}(\mu_i, \sigma_i), \mathcal{N}(0,1))$$

Mean squared error

Kullback–Leibler divergence

MSE : Gaussians prefer being very narrow
KL Div: Gaussians prefer being close to $\mathcal{N}(0,1)$

β is a hyperparameter: tune by hand

Information Buffer

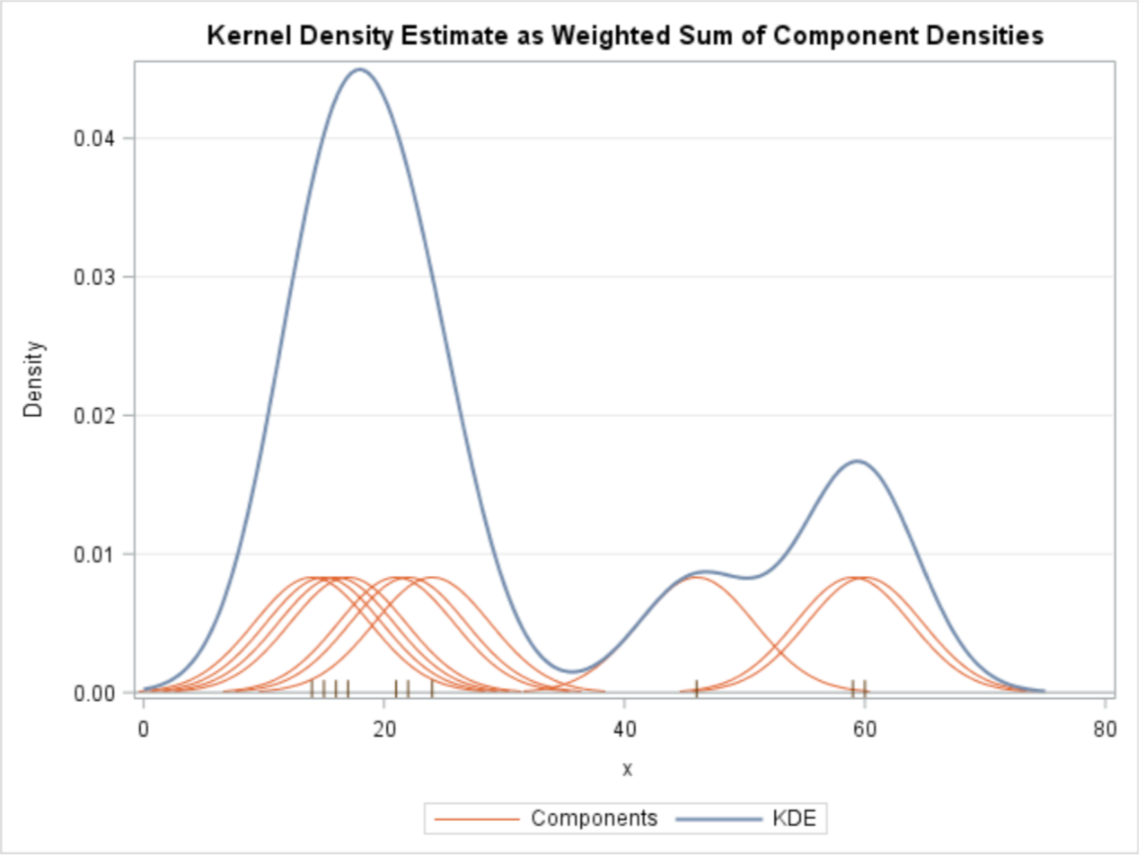
The latent space representation of our datapoints are now ordered

Normally, one would now sample from $\mathcal{N}(0,1)$ in latent space

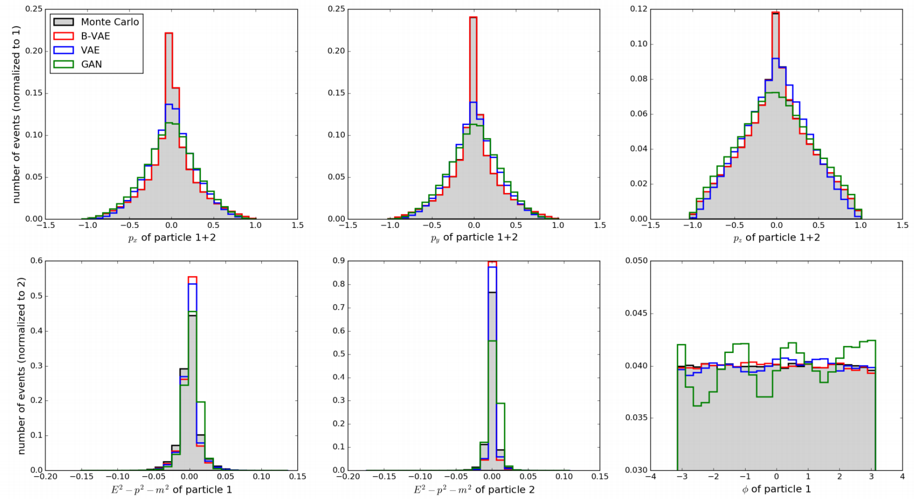
But we can do better: Create information buffer

$$p(z) = \frac{1}{n}\sum_{i}^{n}\mathcal{N}(\mu_i, \sigma_i)$$

Representation of distribution of training data in latent space



Radboud University

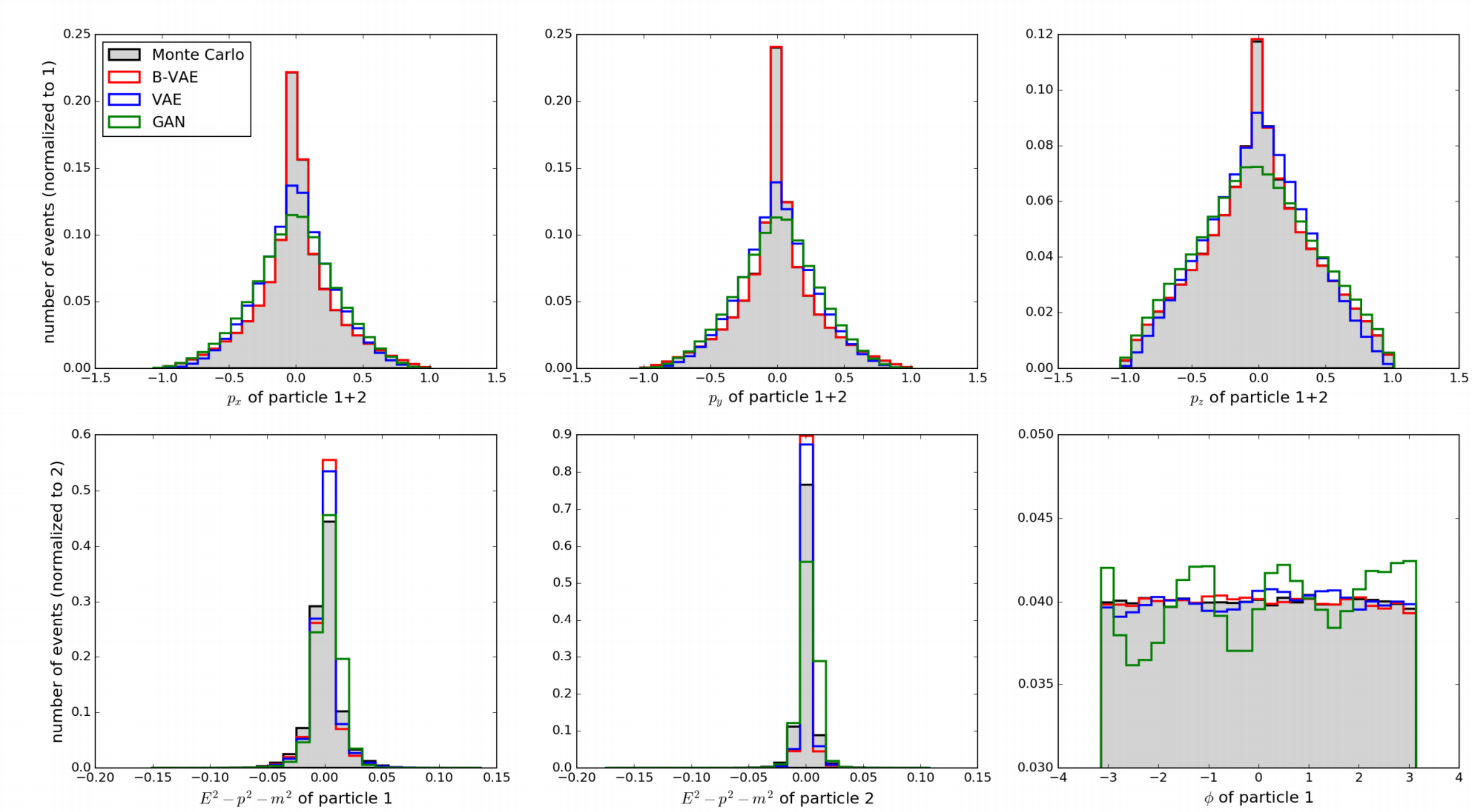


Results

Radboud University

Toy Model

$1 \to 2$ decay with uniform angles and no exact momentum conservation



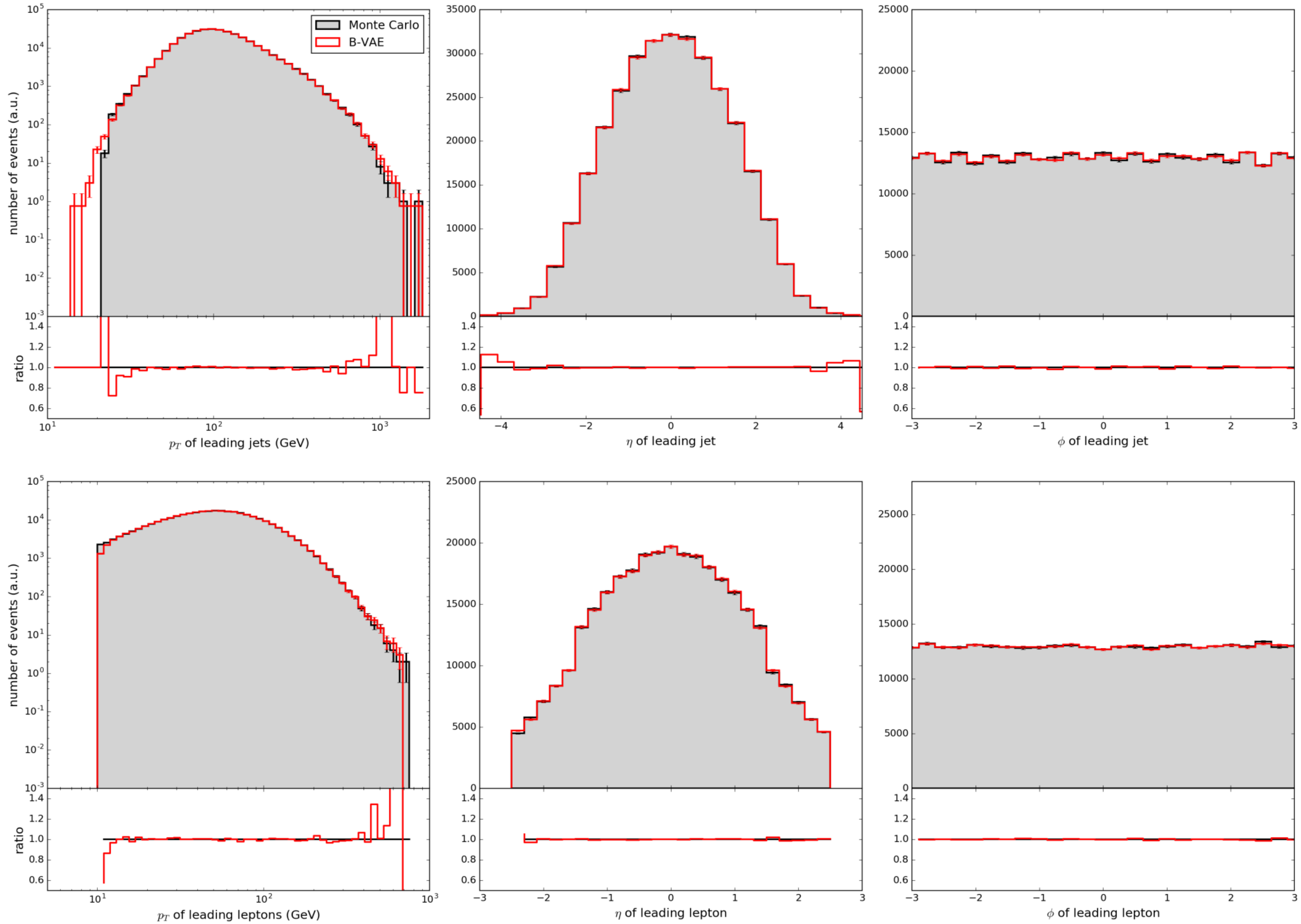
Trained on four-vectors

Top pair production

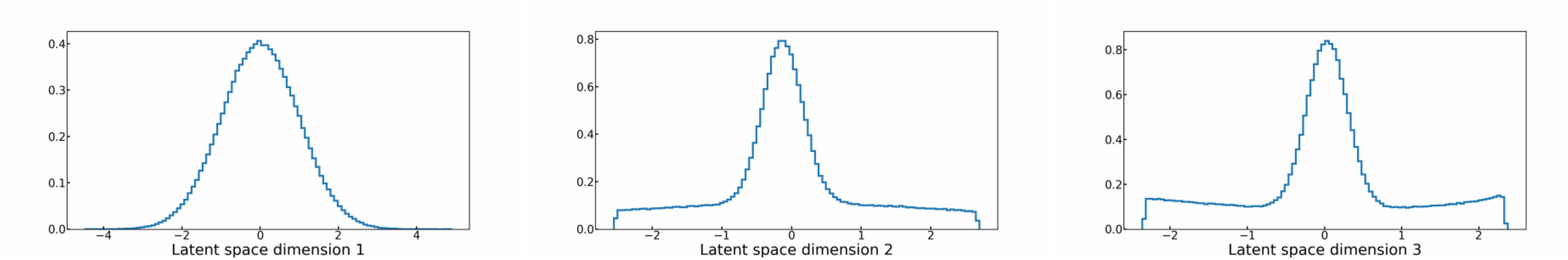
- One top required to decay leptonically
- Number of training points 5×10^5
- MG5 aMC@NLO 6.3.2 + Pythia 8.2 + Delphes 3
- Jets with $p_T > 20 \text{ GeV}$

Event generation with the B-VAE is $\mathcal{O}(10^8)$ faster!

Top pair production



Latent space distributions

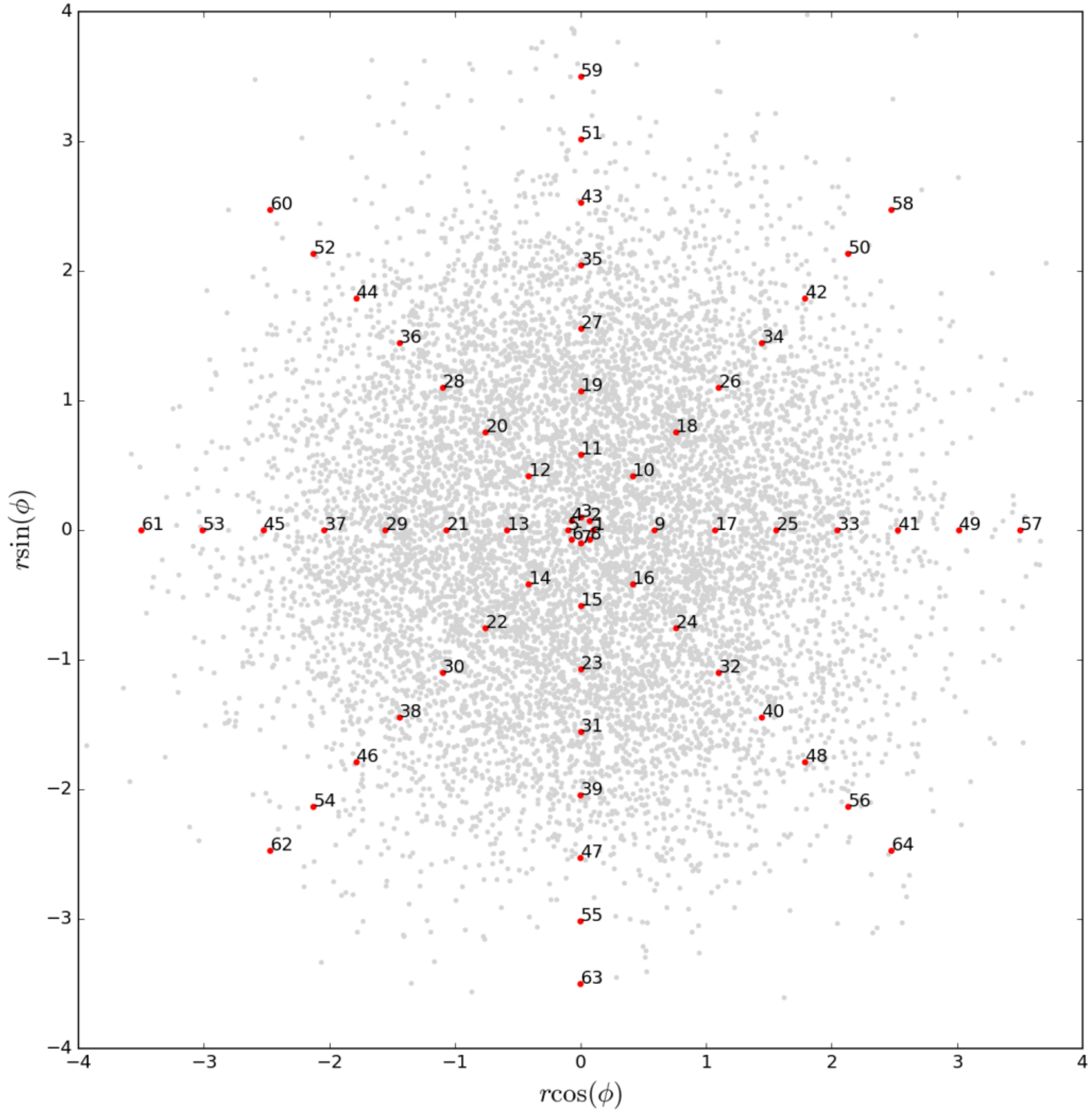


Distributions are still Gaussian-like

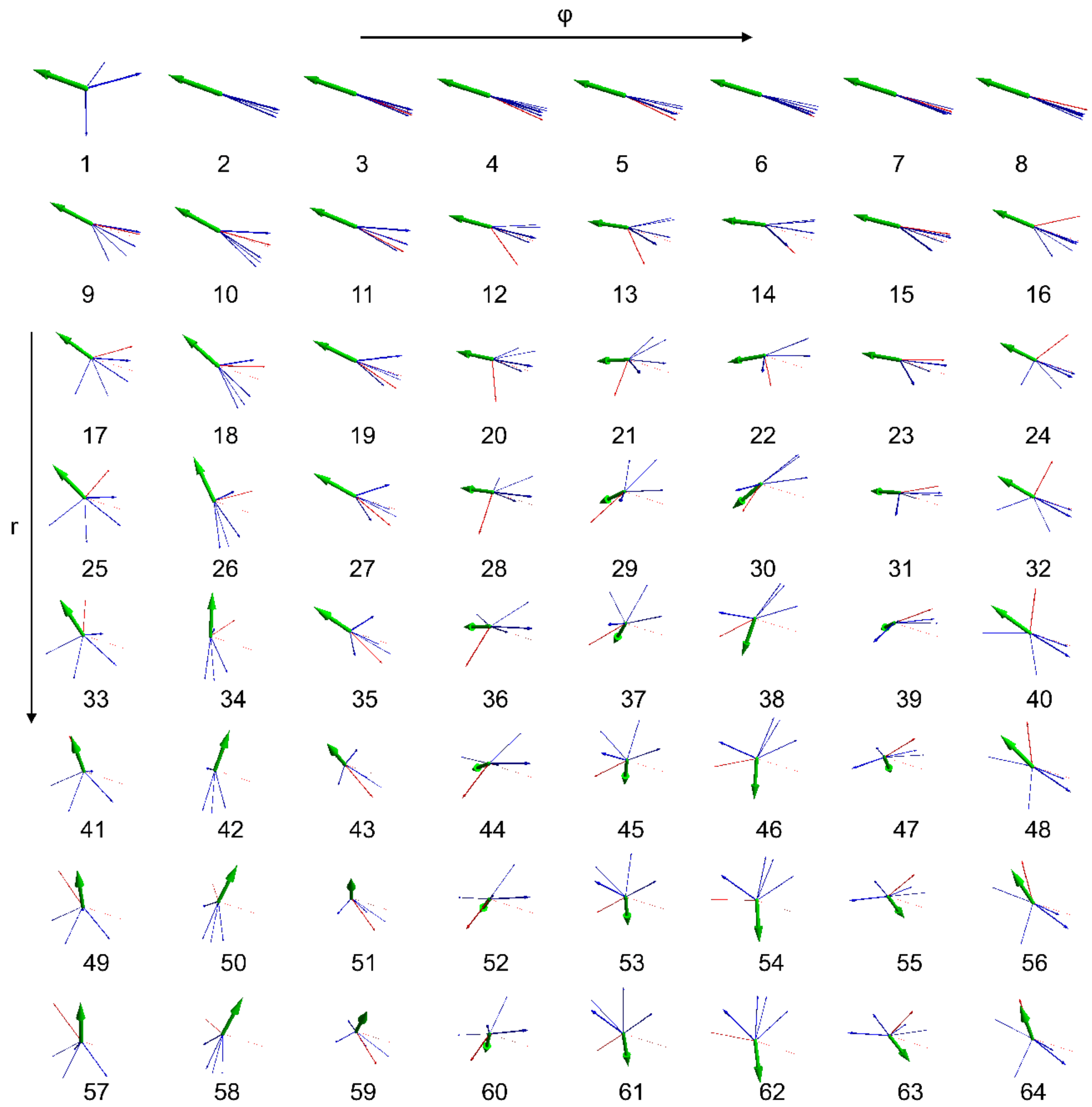
Some have sharp cutoffs: Unphysical events outside

Information buffer very important!

Latent Space Principal Component Analysis



Latent Space Principal Component Analysis

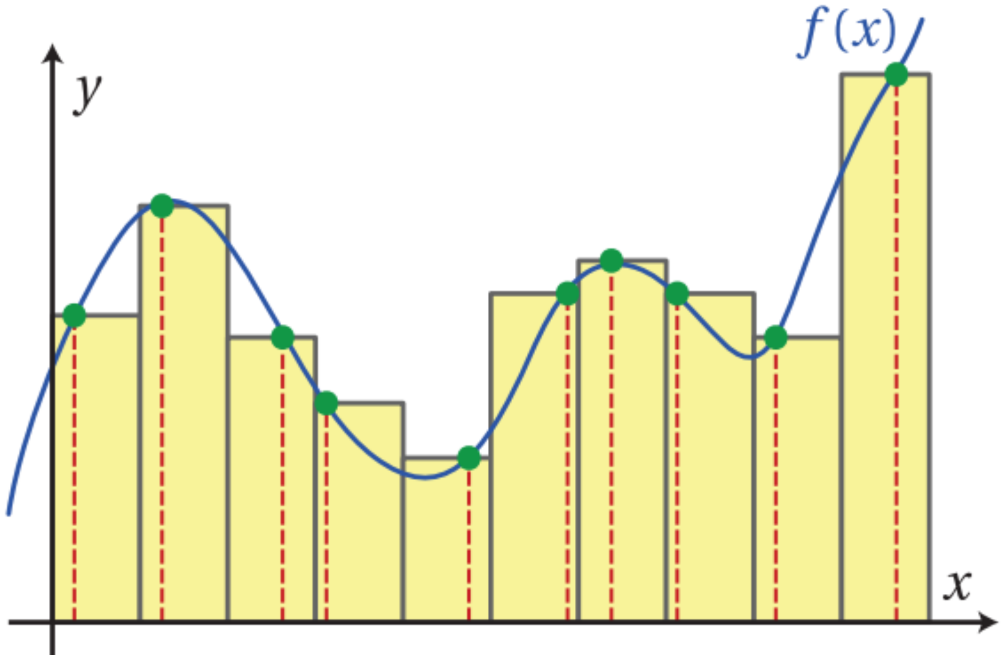


Possible Applications

Most direct application: Importance sampling for ME generation

$$\sigma \propto \int d\Phi |M(\Phi)|^2 = \int d\Phi\, p(\Phi) \frac{|M(\Phi)|^2}{p(\Phi)}$$

Current methods: VEGAS



Recent ML techniques:
Latent variable models
1810.11509

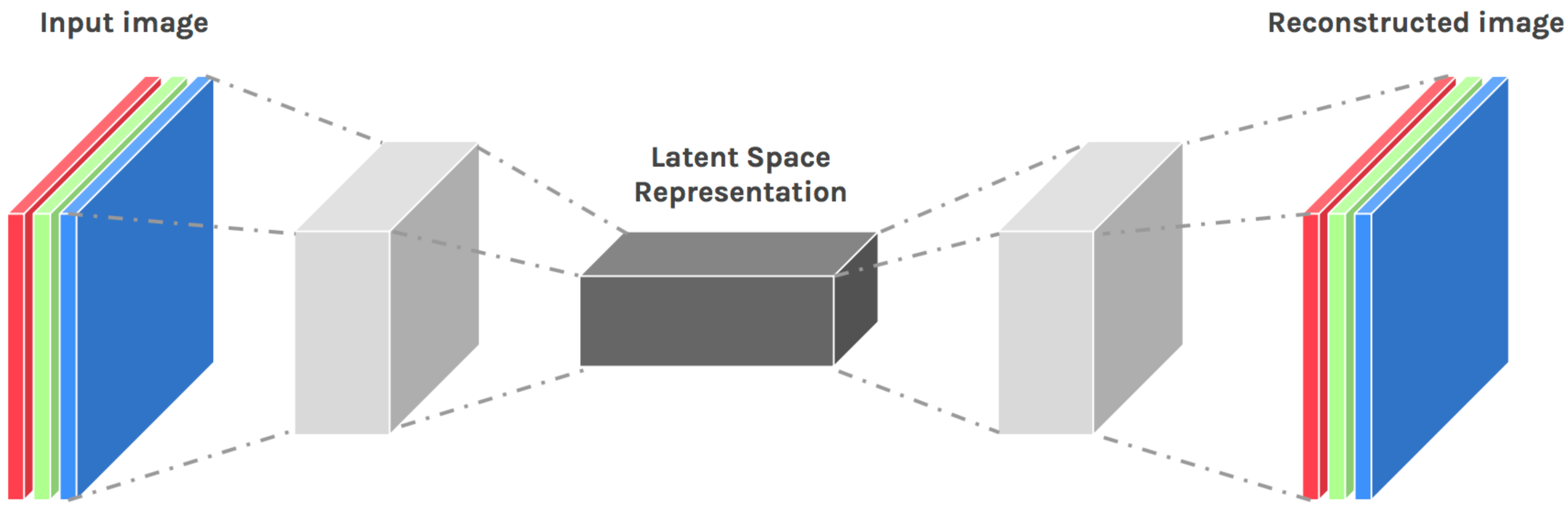
$e^+e^- \to qg\bar{q}$ efficiency:

- VEGAS: ~4%
- LVM: ~ 65%
- B-VAE: ???

Applications & Conclusion

- Data-driven event generators
- Targeted event generation
- Applications outside High Energy Physics?
- ???

Deep neural networks can be used as event generators



Radboud University