Variational Autoencoders

Generative and Deep Learning (GDL)

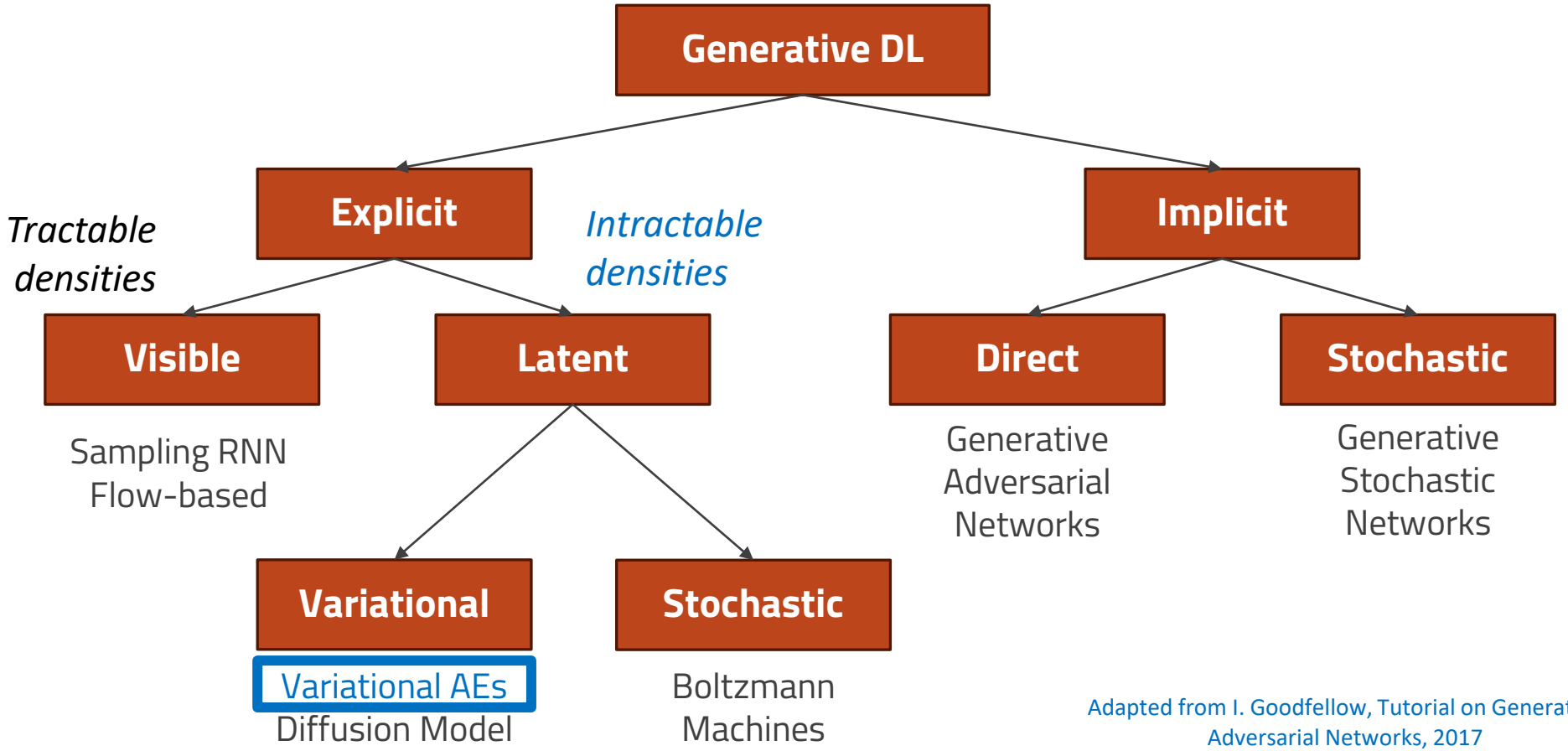
Davide Bacciu (davide.bacciu@unipi.it)

UNIVERSITÀ DI PISA

Lecture Outline

- Autoencoders as explicit density learning models
- Variational autoencoders
  - A neural latent variable model
  - Variational inference
  - Reparameterization trick
- Variational autoencoders for representation learning

Today in the taxonomy

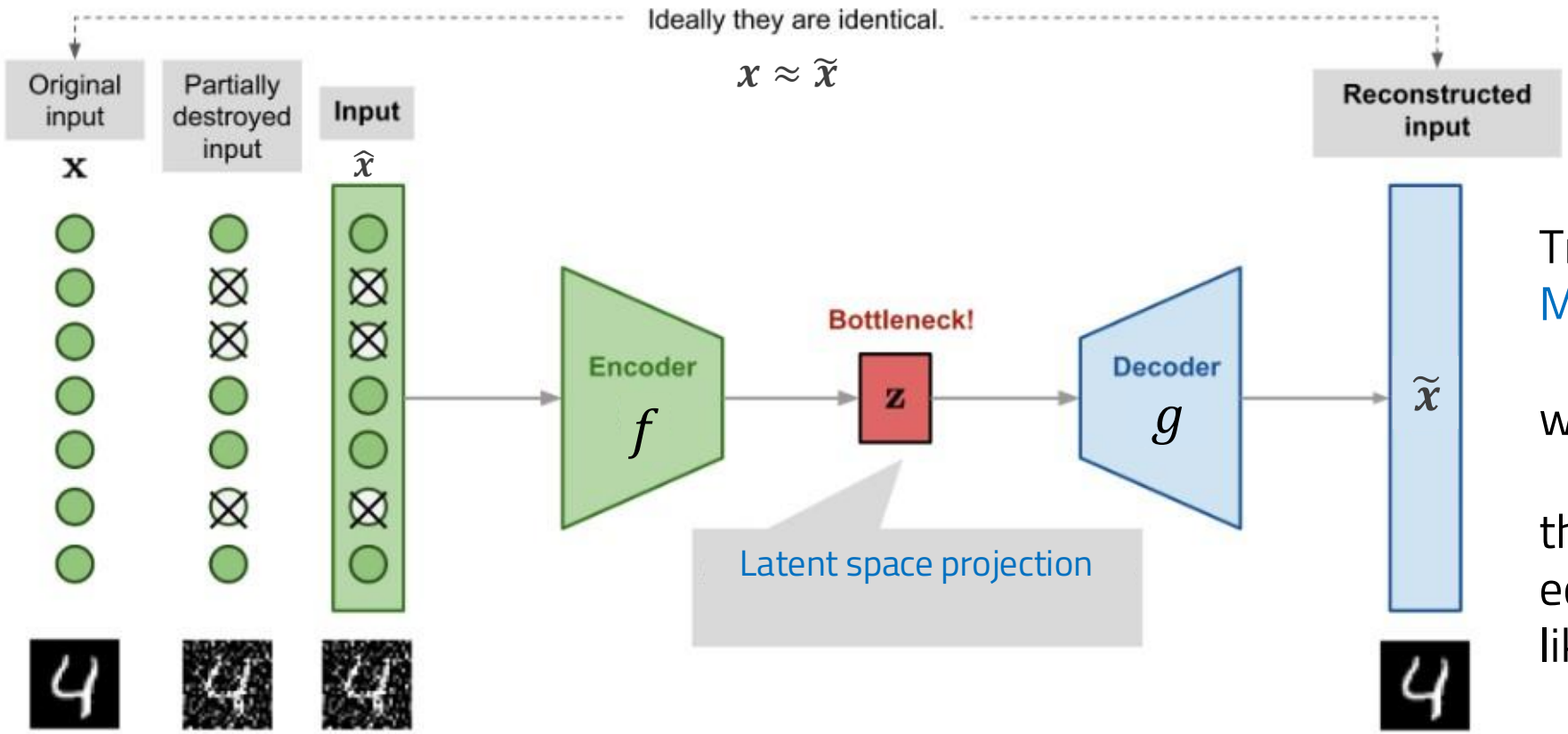


Adapted from I. Goodfellow, Tutorial on Generative Adversarial Networks, 2017

Autoencoders and density learning

Remember me?

The denoising autoencoder (DAE)

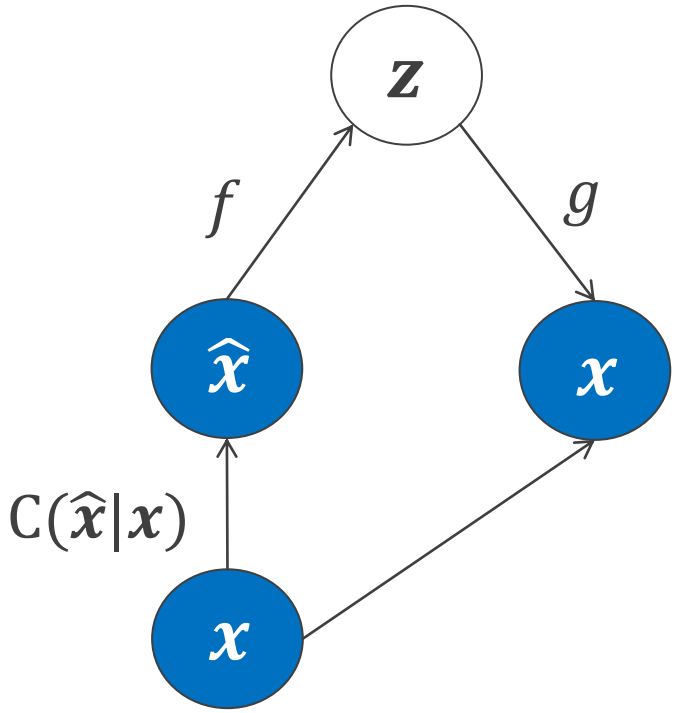


Trains by minimizing MSE

when
$P(x|z) \sim \mathcal{N}(g(z), \sigma^2)$
then minimizing MSE is equal to maximizing likelihood

Another Interpretation…

…yes, exactly the one you are thinking of



DAE learns a (conditional) denoising distribution of input data

$$P(\boldsymbol{x}|\hat{\boldsymbol{x}})$$

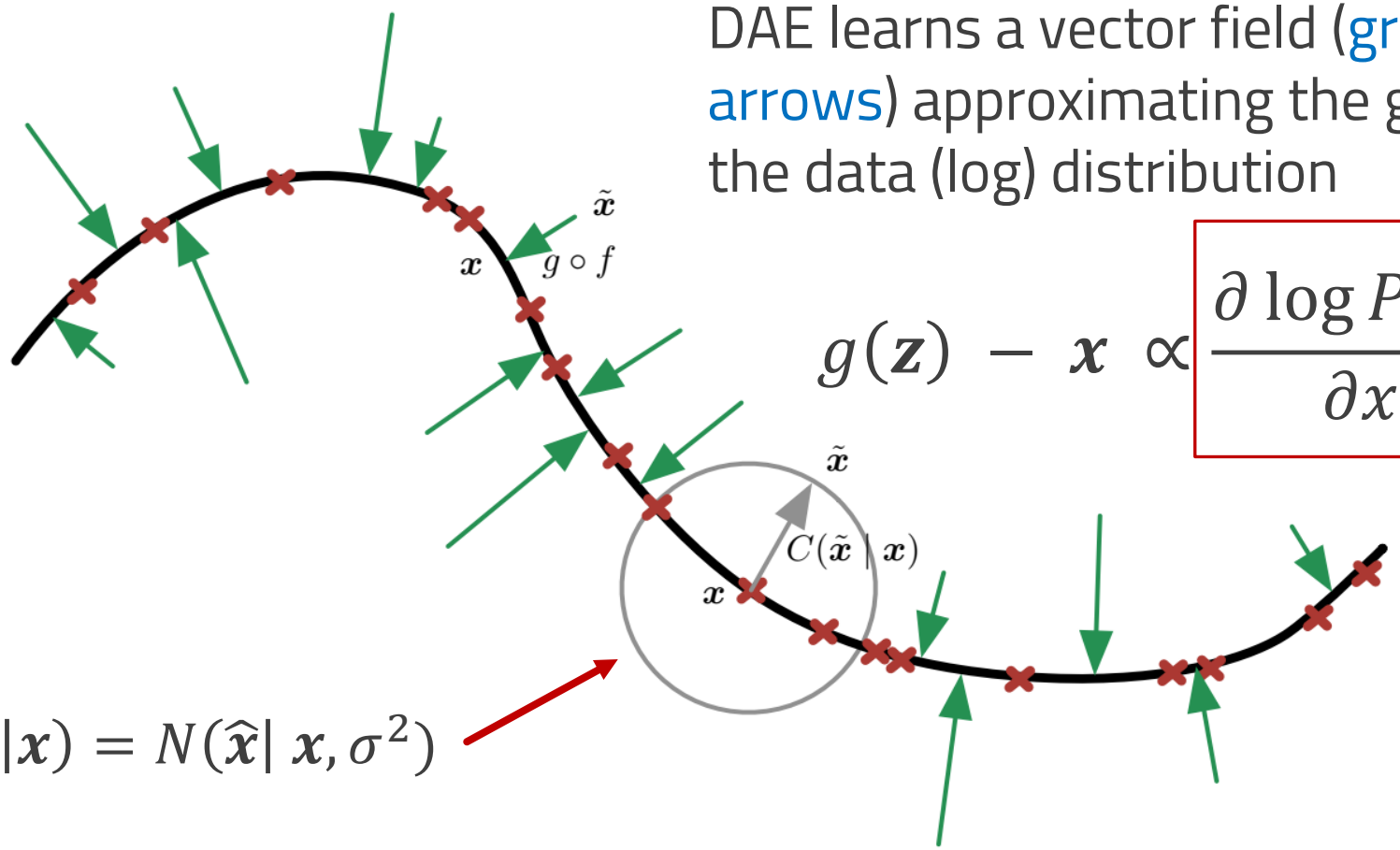
By minimizing the MSE

$$\mathbb{E}_{\boldsymbol{x},\hat{\boldsymbol{x}}}\left[\left|\left|x - g(f(\hat{\boldsymbol{x}}))\right|\right|_2^2\right]$$

or equivalently (under mild assumptions of Normality) by maximixing the conditional log-likelihood

$$\mathbb{E}_{\boldsymbol{x},\hat{\boldsymbol{x}}}[\log P(\boldsymbol{x}|\boldsymbol{z} = f(\hat{\boldsymbol{x}}))]$$

In the manifold learning setting



DAE learns a vector field (green arrows) approximating the gradient of the data (log) distribution

$$g(\boldsymbol{z}) - \boldsymbol{x} \propto \frac{\partial \log P(\boldsymbol{x})}{\partial x}$$

The score

$$\mathrm{C}(\hat{\boldsymbol{x}}|\boldsymbol{x}) = N(\hat{\boldsymbol{x}}|\, \boldsymbol{x}, \sigma^2)$$

Score function

- Given a probability density p($\boldsymbol{x}$), the score function is the gradient with respect to the input of the log-density

$$s(x) = \frac{\partial \log p(x)}{\partial x} = \nabla_x \log p\,(x)$$

$p(x)$ is unknown and only samples from $p(x)$ are available!

- $\log p\,(x)$ $\Rightarrow$ how likely a point is
- $\nabla_x \log p\,(x) \Rightarrow$ direction of steepest increase in probability

The score tells you in which direction you should move x to reach higher-likelihood (density) regions

DAE provides a way to estimate the score without p(x)

DAE and the score

- The DAE optimizes $\min \mathbb{E}_{\boldsymbol{x},\hat{\boldsymbol{x}}}\left[\left|\left|x - g(f(\hat{\boldsymbol{x}}))\right|\right|_2^2\right]$, which has an optimal minimizer when

$$g\big(f(\hat{\boldsymbol{x}})\big) = \mathbb{E}[\boldsymbol{x}|\hat{\boldsymbol{x}}] \text{ (conditional mean)}$$

- For small Gaussian noise (in $\hat{\boldsymbol{x}} = \boldsymbol{x} + \epsilon$) we can use Tweedie's formula

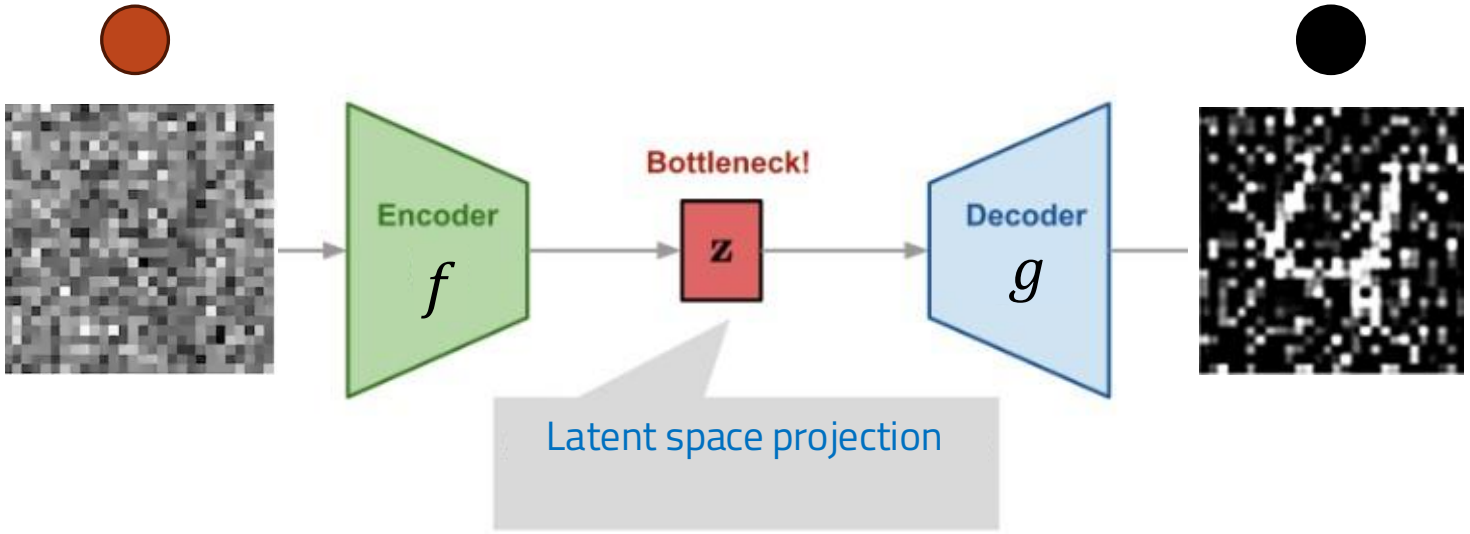
$$\mathbb{E}[\boldsymbol{x}|\hat{\boldsymbol{x}}] = \hat{\boldsymbol{x}} + \sigma^2 \nabla_{\hat{\boldsymbol{x}}} \log p\,(\hat{\boldsymbol{x}})$$

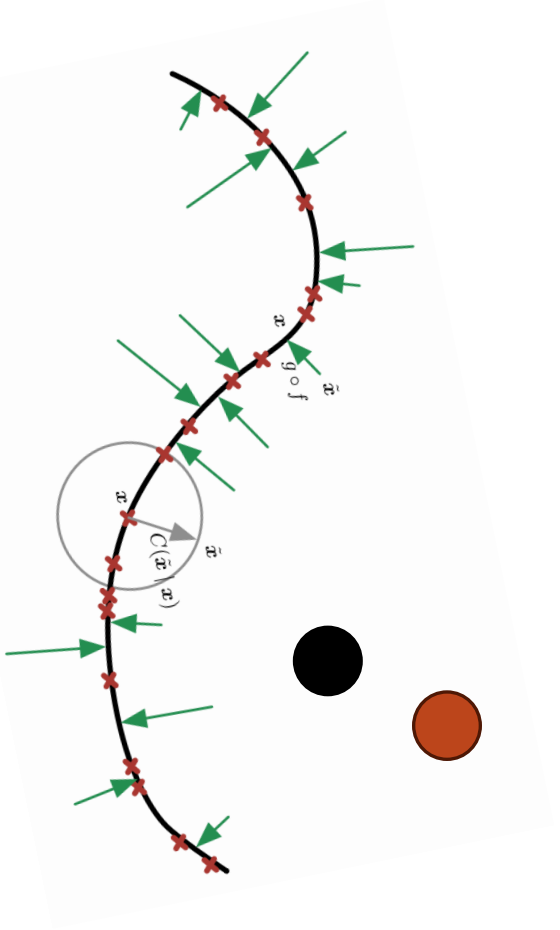
- Plugging the minimizer and solving for the score yields

$$\frac{g\big(f(\hat{\boldsymbol{x}})\big) - \hat{\boldsymbol{x}}}{\sigma^2} = \nabla_{\hat{\boldsymbol{x}}} \log p\,(\hat{\boldsymbol{x}})$$

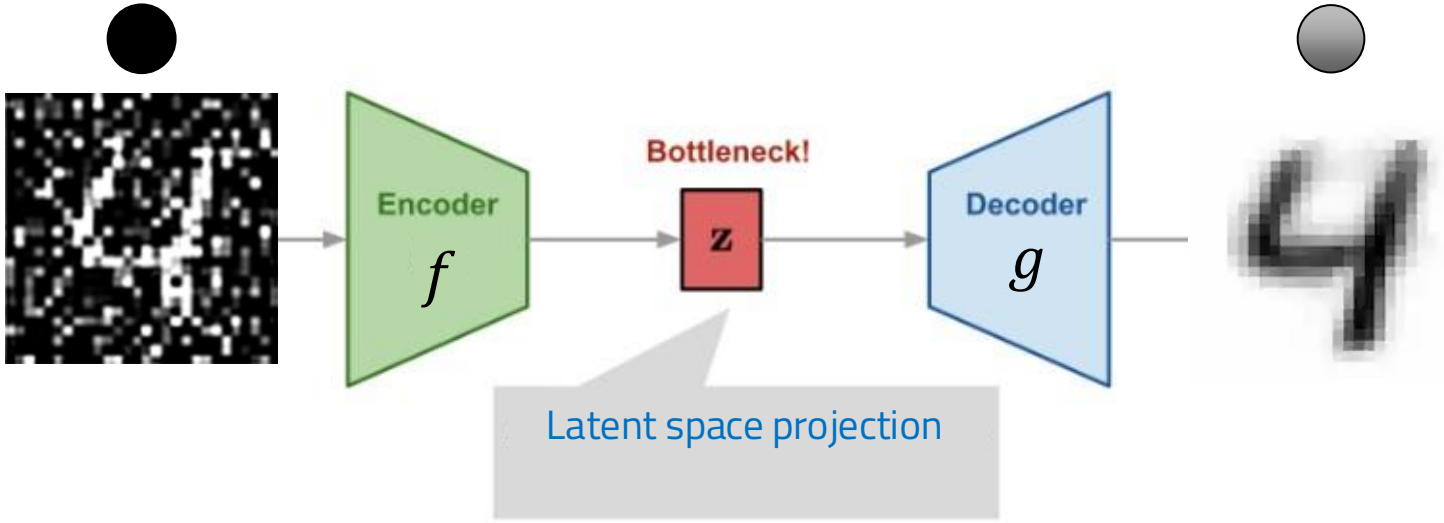
- That is the vector field learned by the DAE

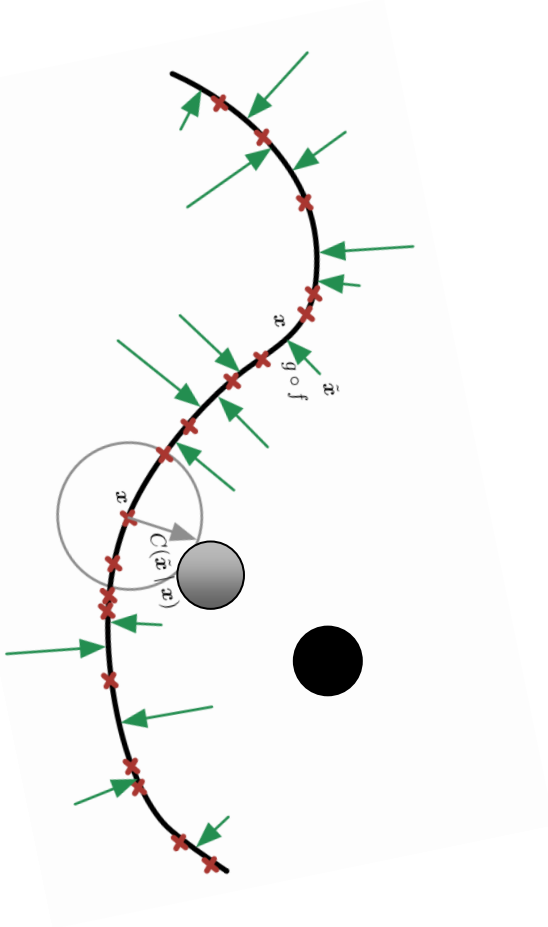
So… can sample an autoencoder?



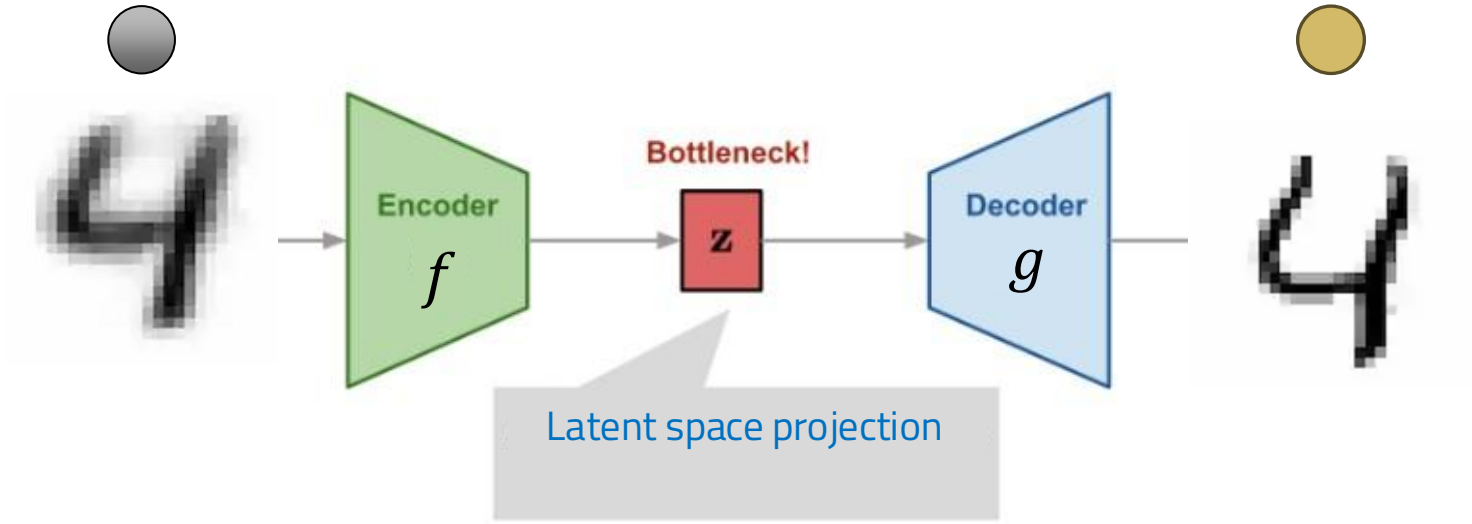


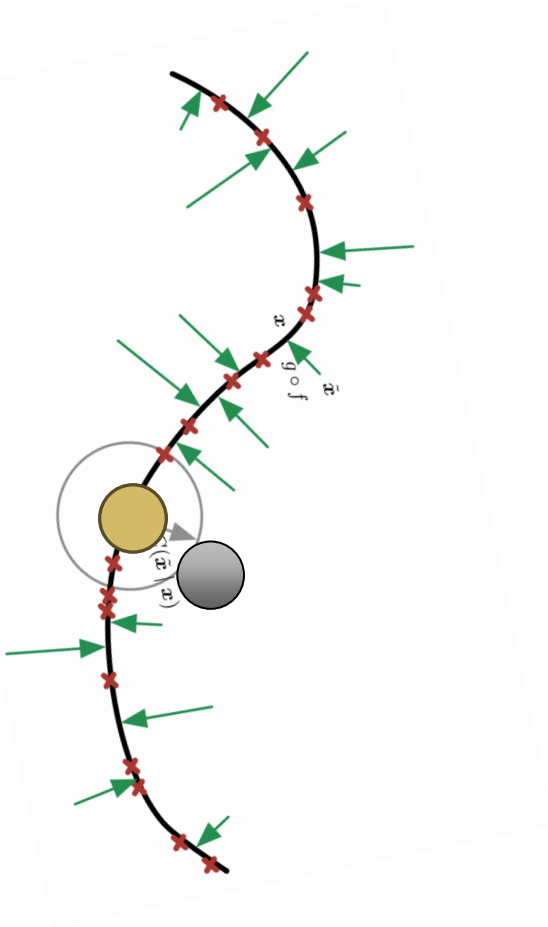
So… can sample an autoencoder?





So… can sample an autoencoder?



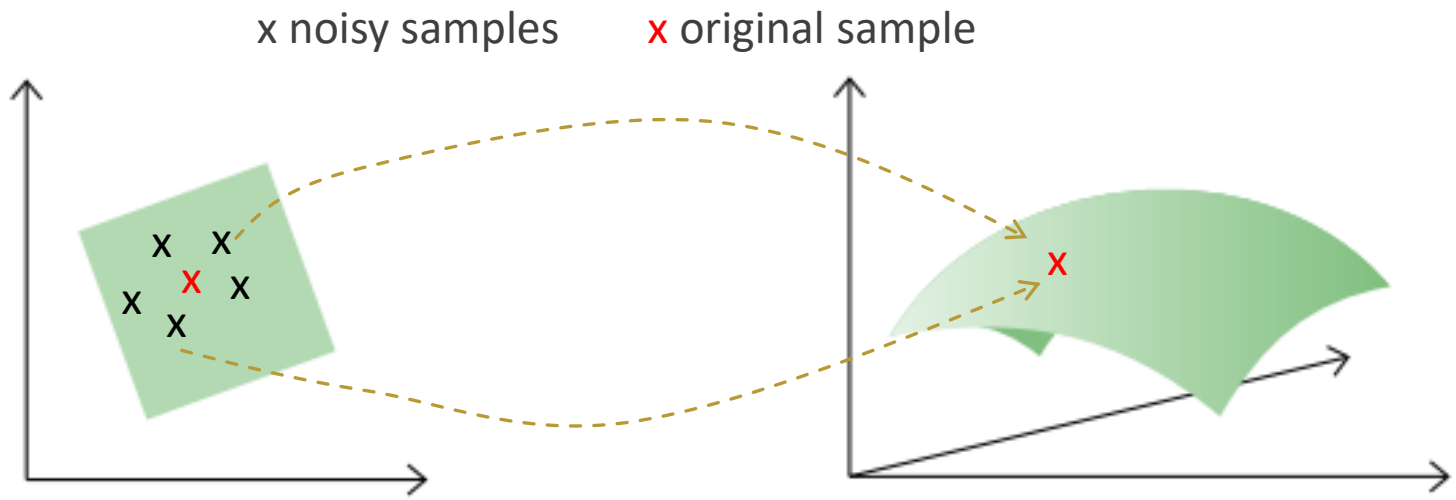


Variational Autoencoders

On the limits of denoising/contractive AEs

We are not learning an explicit data distribution

- We approximate the gradient (of its log) around the training data point
- We don't have guarantees for points outside the training set



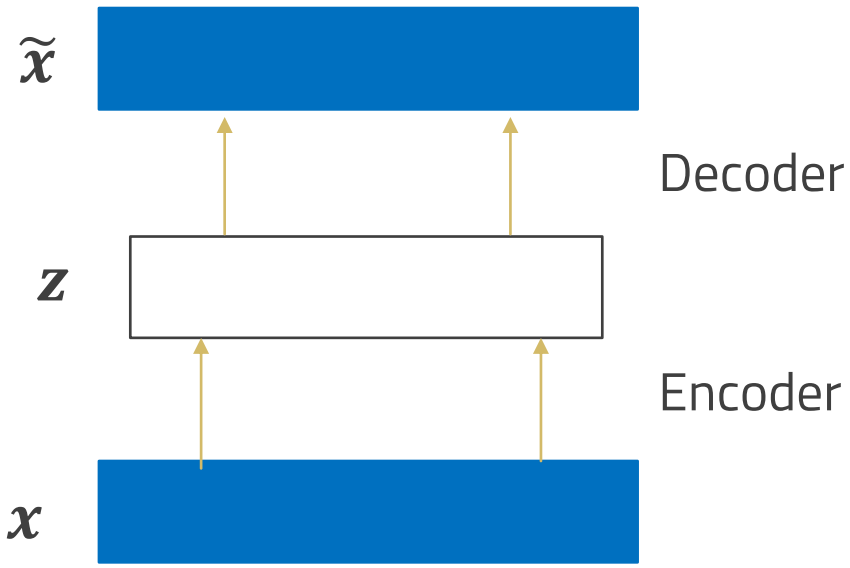
A new goal

- Set the autoencoder to learn explicitly a model distribution $P_\theta(\boldsymbol{x})$ on data $\boldsymbol{x}$ parameterized by weights θ
- An autoencoder represents observable data $\boldsymbol{x}$ in a latent space $\boldsymbol{z}$ given by the activations of the hidden neurons in the bottleneck
- We have a latent variable model regulated by continuous variables $\boldsymbol{z}$

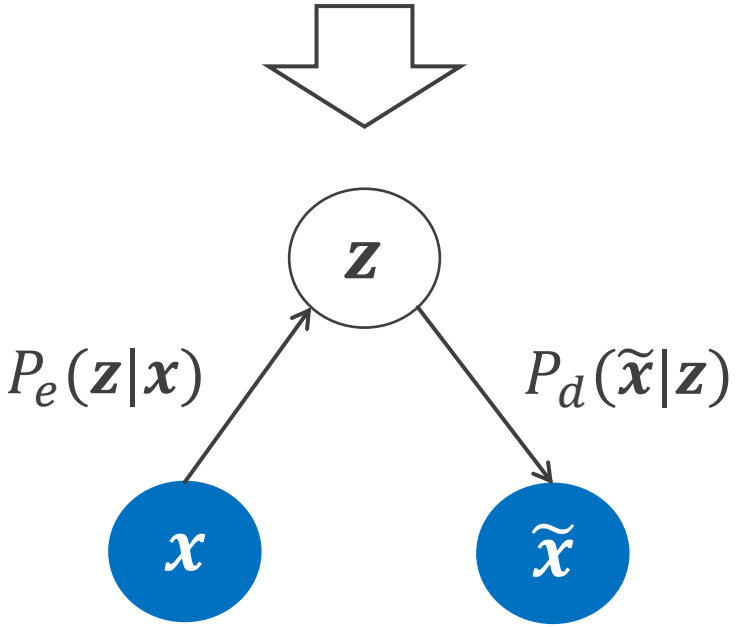
$$P_\theta(\boldsymbol{x}) = \int P_\theta(\boldsymbol{x}|\boldsymbol{z}) P_\theta(\boldsymbol{z}) d\boldsymbol{z}$$

Typically, intractable for nontrivial models
(cannot be computed for all $\boldsymbol{z}$ assignments)
⇒ will need an approximation!

A Neural Network with Latent Variables?



It is not difficult to cast a probabilistic twist on AE (by making encoder-decoder maps probabilistic)



A Deeper Probabilistic Push

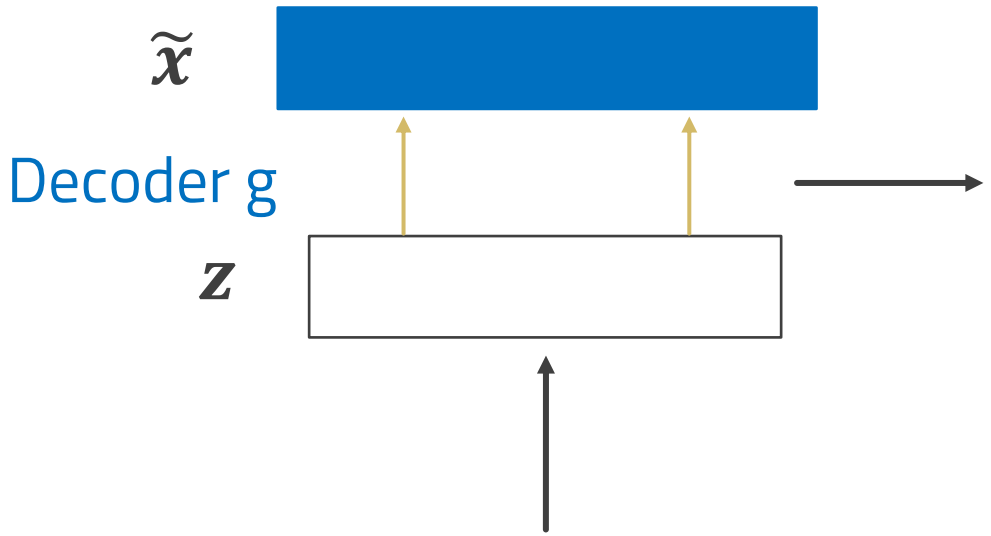
As an additional push in the probabilistic interpretation, we assume to be able to generate the reconstruction from a sampled latent representation

Sample from the true conditional $P(\tilde{x}|z)$

Sample latent variables from the true prior $P(z)$

Of course we don't have access to the true distributions, so how do we approximate them?

Variational Autoencoders (VAE) – The Catch



Represent the $P(\tilde{x}|z)$ distribution through a neural network g

At training time sample **z** conditioned on data **x** and train the decoder g to reconstruct **x** itself from **z**

Sample z from a simple distribution informed by the input (e.g. a Gaussian)

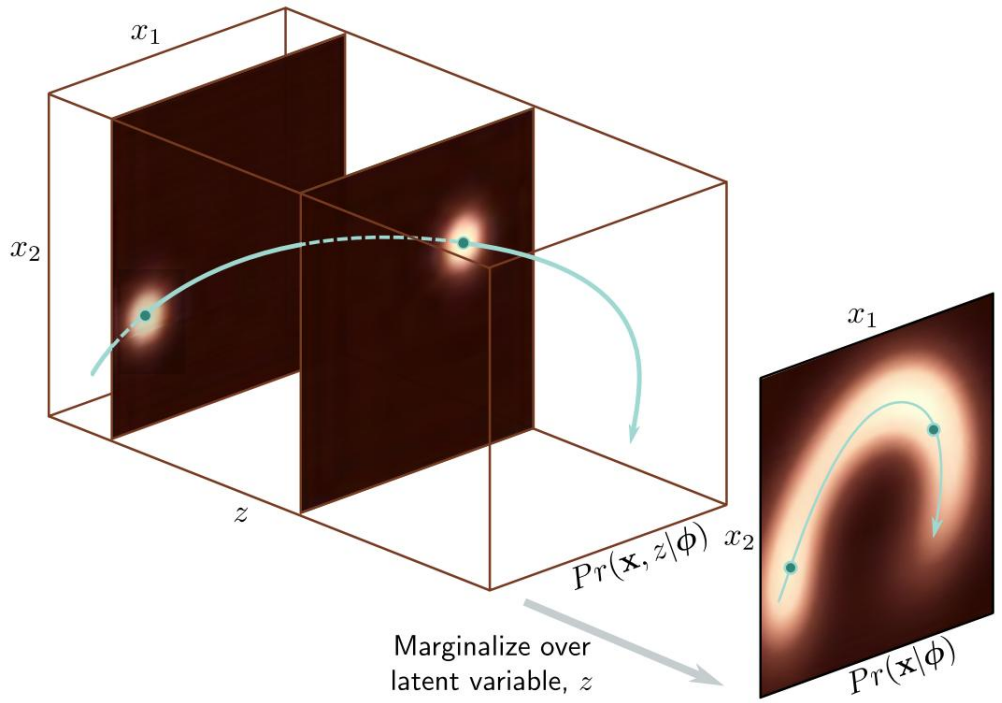
$$z \sim \mathcal{N}(\mu(x), \sigma(x))$$

VAE Training

Ideally, one would like to train maximizing

$$L(D) = \prod_{i=1}^{N} P(\boldsymbol{x_i})$$

$$= \prod_{i=1}^{N} \int P(\boldsymbol{x_i}|\boldsymbol{z}) P(\boldsymbol{z}) d\boldsymbol{z}$$



VAE Training – Is it all this easy?

Ideally, one would like to train maximizing

$$L(D) = \prod_{i=1}^{N} P(\boldsymbol{x_i})$$

$$= \prod_{i=1}^{N} \int P(\boldsymbol{x_i}|\boldsymbol{z}) P(\boldsymbol{z}) d\boldsymbol{z}$$

Unfortunately for you: no!

Intractable

Variational approximation

Variational Approximation

The revenge of the ELBO (Evidence Lower BOund)

$$\log P(x|\theta) \geq \mathbb{E}_Q[\log P(x,z)] - \mathbb{E}_Q[\log Q(z)] = \mathcal{L}(x,\theta,\phi)$$

Maximizing the ELBO allows approximating from below the intractable log-likelihood $\log P(x)$

$$\mathcal{L}(x,\theta,\phi) = \mathbb{E}_Q[\log P(x|z)] + \mathbb{E}_Q[\log P(z)] - \mathbb{E}_Q[\log Q(z)]$$

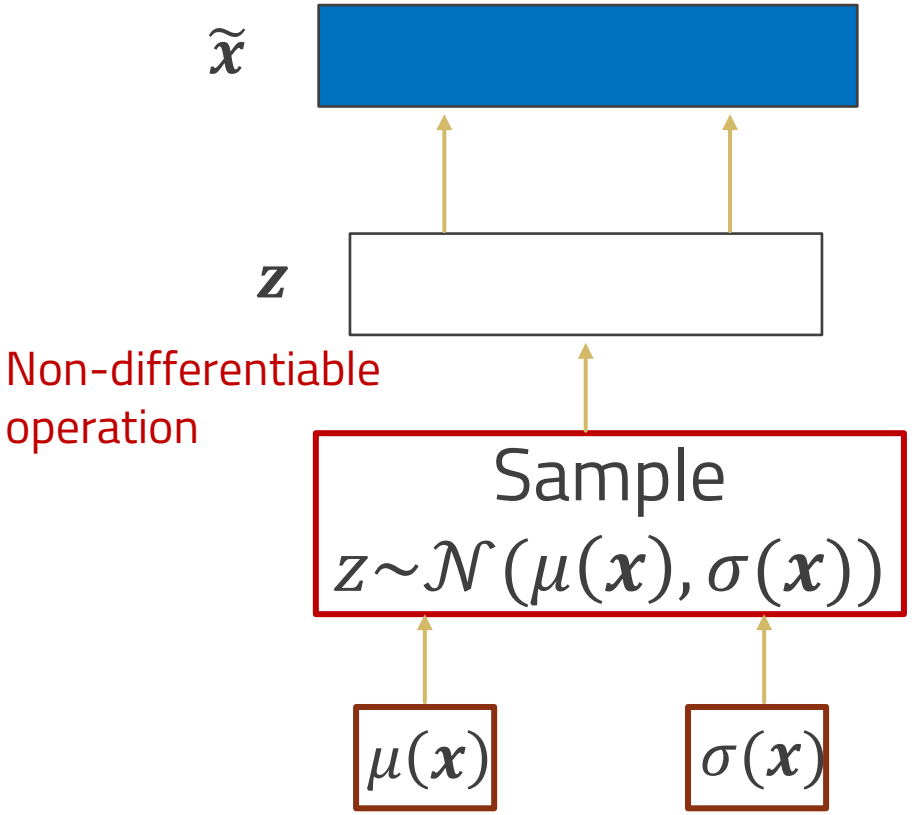
Decoder estimate of the reconstruction (based on a sampled z)

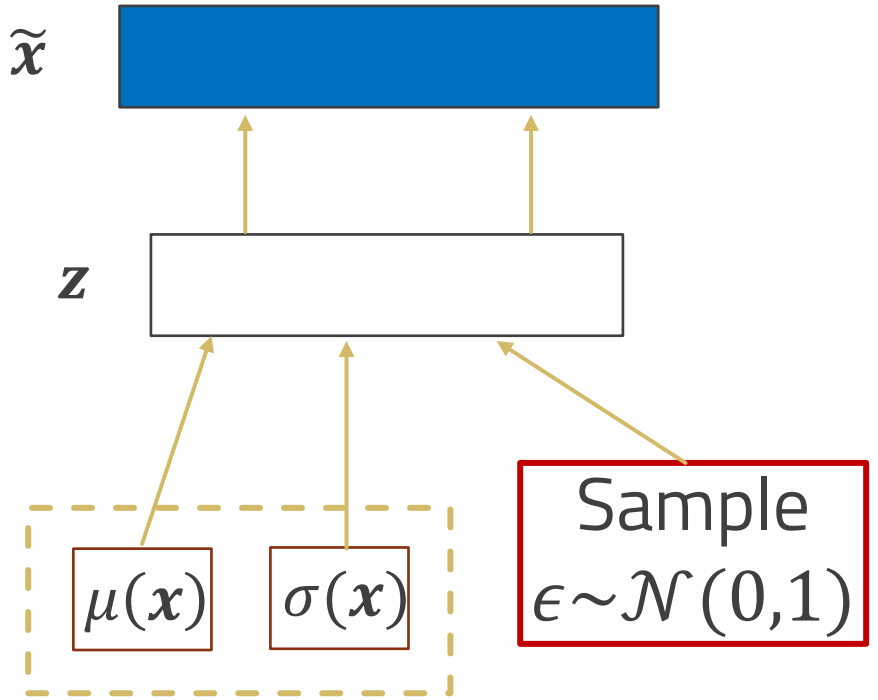
(It is not differentiable!)

$$-KL(Q(z|\phi)||P(z|\theta))$$

Need a Q(z) function to approximate P(z)

Reparameterization Trick

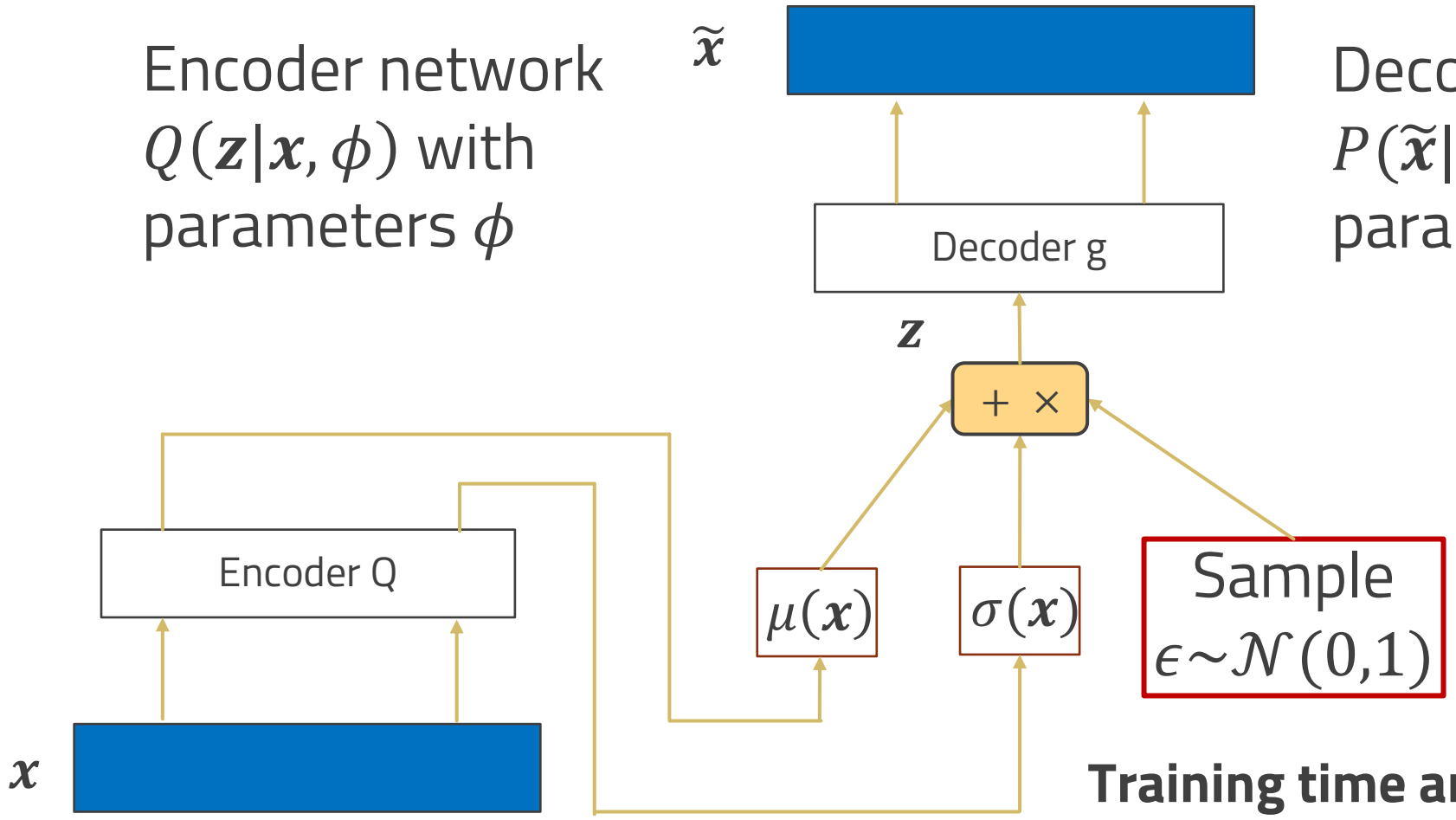




Sampling is limited to non differentiated variable ϵ $\Longrightarrow$ Can backpropagate

Putting things back together

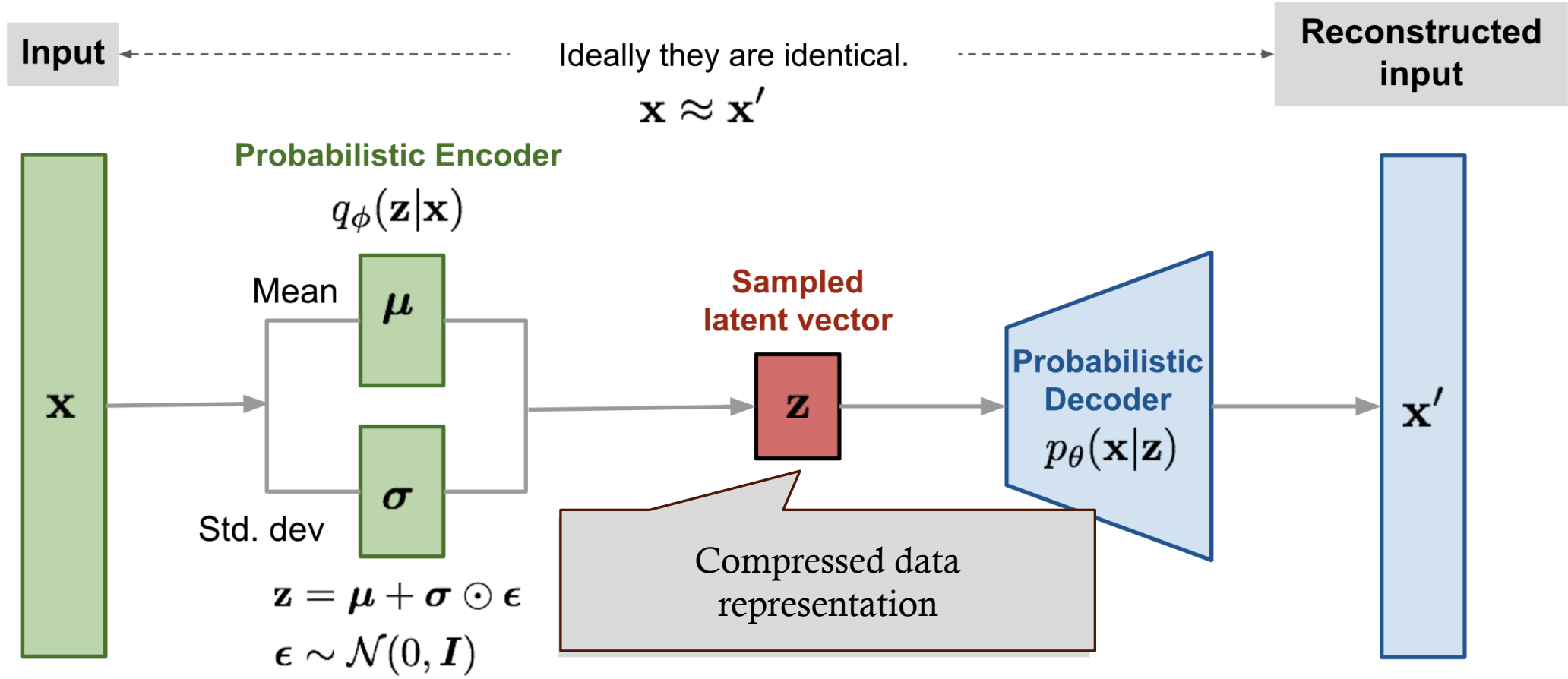
Encoder network $Q(\boldsymbol{z}|\boldsymbol{x}, \phi)$ with parameters ϕ



Decoder network $P(\tilde{\boldsymbol{x}}|\boldsymbol{z}, \theta)$ with parameters θ

Training time architecture

The VAE Architecture



VAE Training

Training is performed by backpropagation on θ, ϕ to optimize the ELBO

reconstruction

$$\mathcal{L}(x, \theta, \phi) = \mathbb{E}_Q\left[\log P(x|z = \mu(x) + \sigma^{1/2}(x) * \epsilon, \theta)\right]$$
$$-KL(Q(z|x, \phi)||P(z|\theta))$$

regularization

Can be computed in closed form when both Q(z) and P(z) are Gaussians

$$KL(\mathcal{N}(\mu(x), \sigma(x)) \,||\mathcal{N}(0,1)\,)$$

Train the encoder to behave like a Gaussian prior with zero-mean and unit-variance

VAE Loss – Another view on differentiability

In principle we would like to optimize the following loss by SGD

$$\mathbb{E}_{X\sim D}[\mathbb{E}_{z\sim Q}[\log P(x|z)] - KL(Q(z|x,\phi)||P(z))]$$

which can be rearranged following the reparametrization trick

$$\mathbb{E}_{X\sim D}[\mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}[\log P(x|z=\mu(x)+\sigma^{1/2}(x)*\epsilon,\theta)] - KL(Q(z|x,\phi)||P(z))]$$

No expectation is w.r.t distributions that depend on model parameters

⇒ We can move gradients into them

Information Theoretic Interpretation

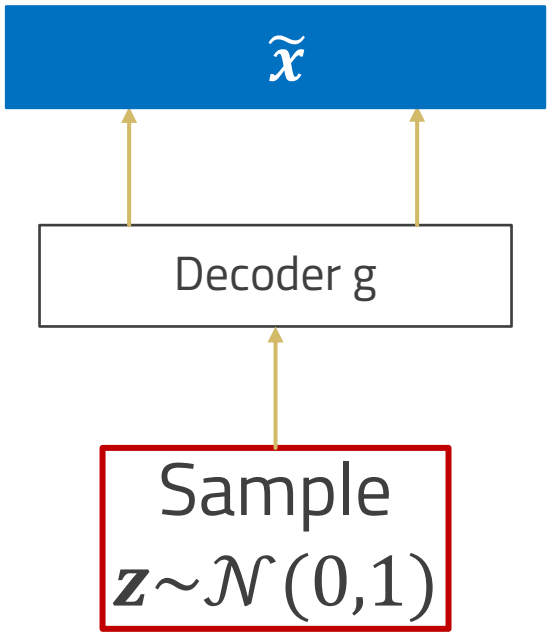
$$\mathbb{E}_{X\sim D}[\mathbb{E}_{z\sim Q}[\log P(x|z)] - KL(Q(z|x,\phi)||P(z))]$$

Number of bits required to reconstruct x from z under the ideal encoding (i.e. $Q(z|x)$ is generally suboptimal)

Number of bits required to convert an uninformative sample from $P(z)$ into a sample from $Q(z|x)$

Information gain - Amount of extra information that we get when z comes from $Q(z|x)$ (informed) instead of from $P(z)$ (uninformed)

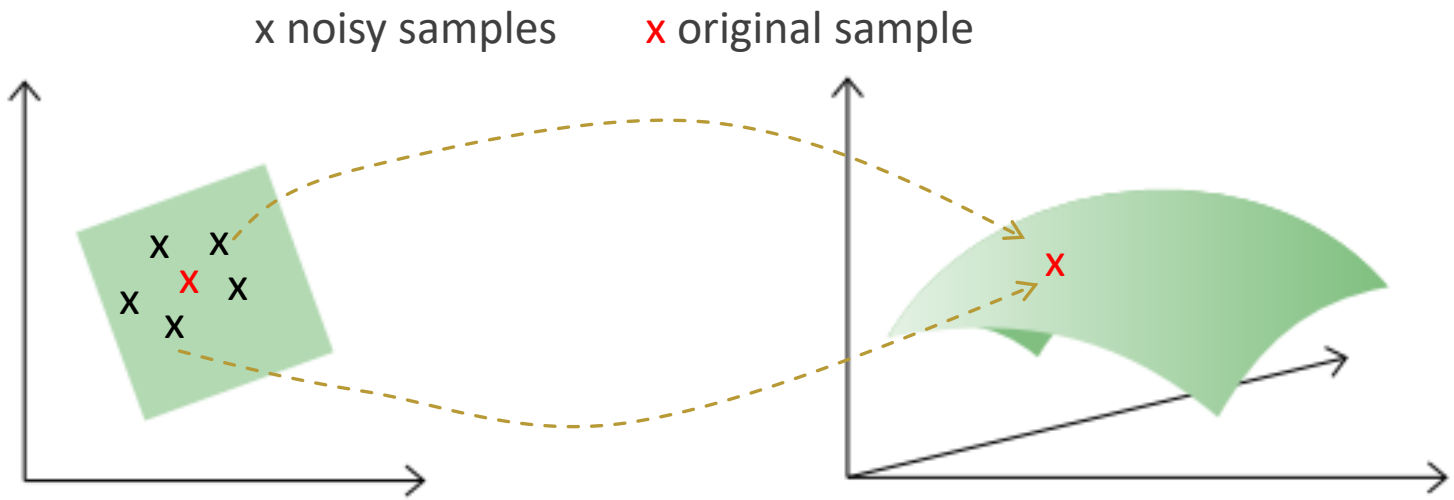
Sampling the VAE (a.k.a. testing)



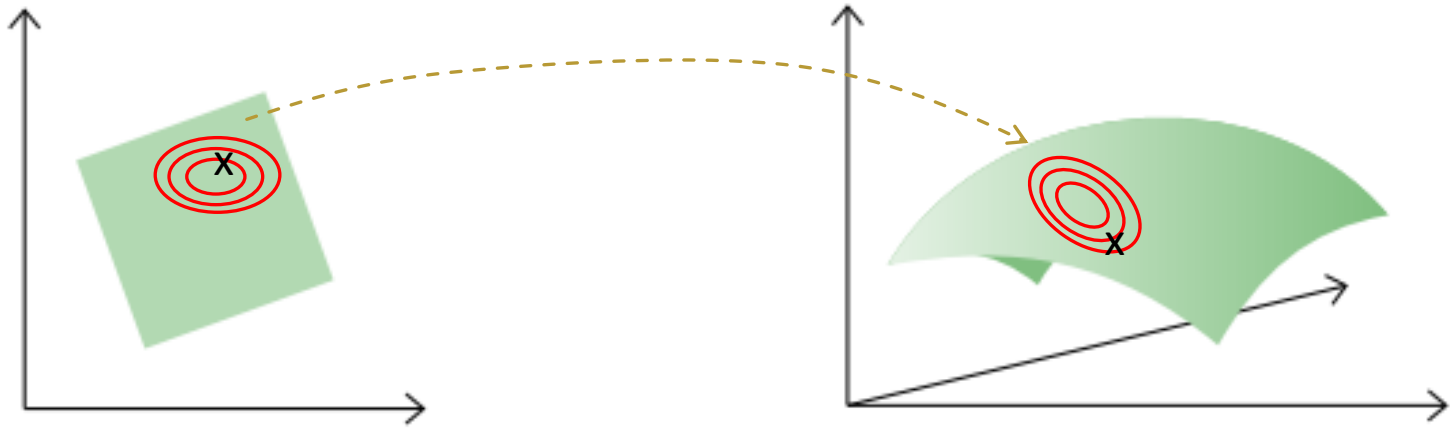
At test time detach the encoder, sample a random encoding and generate the sample as the corresponding reconstruction

VAE vs Denoising/Contractive AE

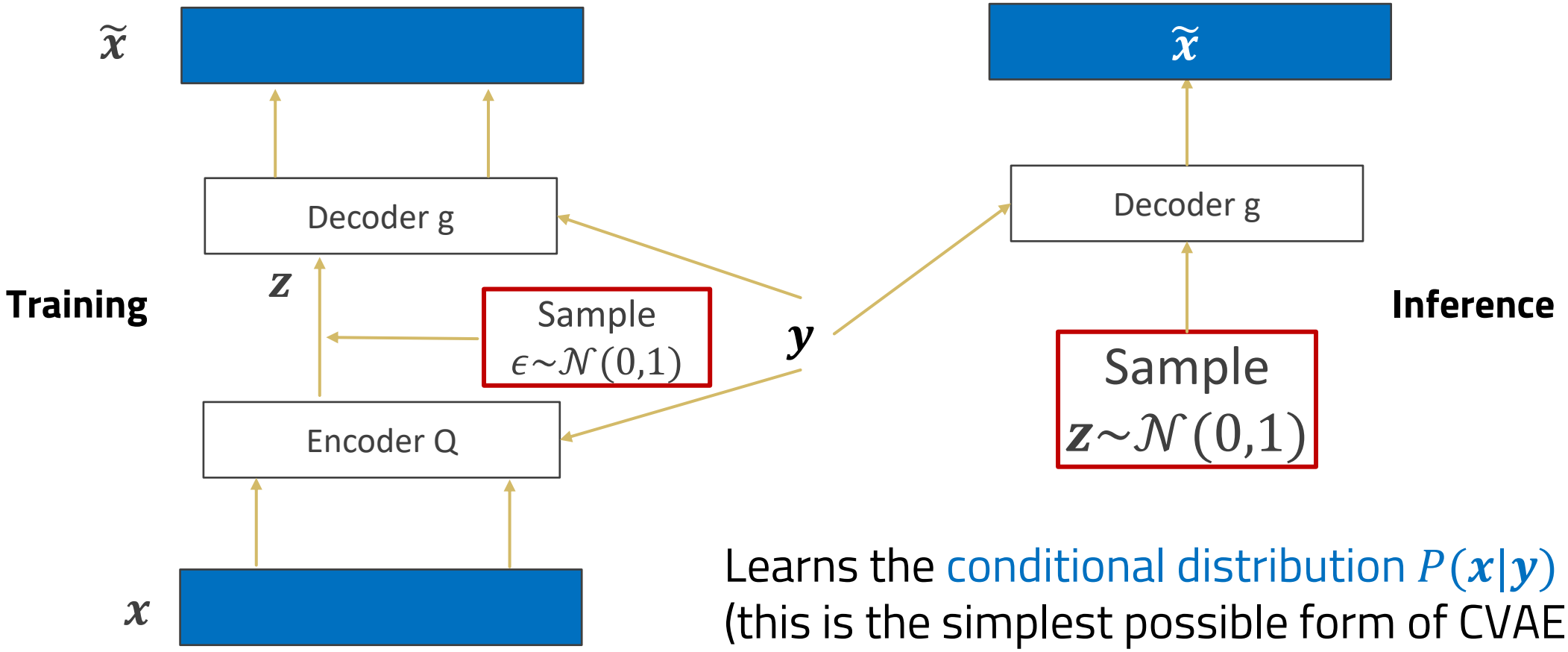
Contractive AE



Variational AE



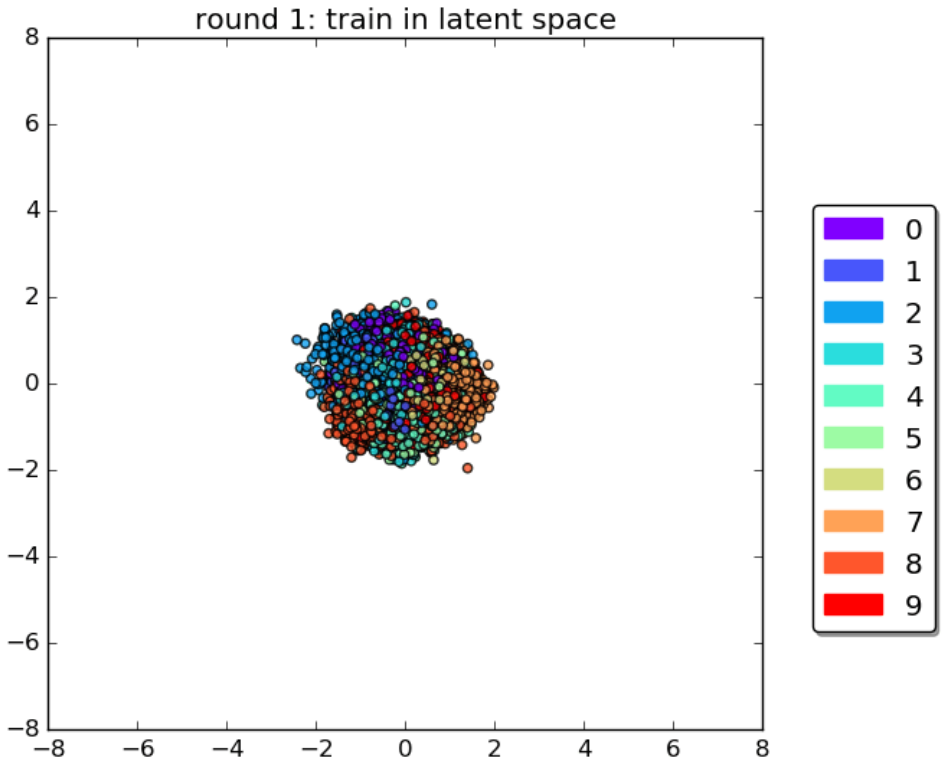
Conditional Generation (CVAE)



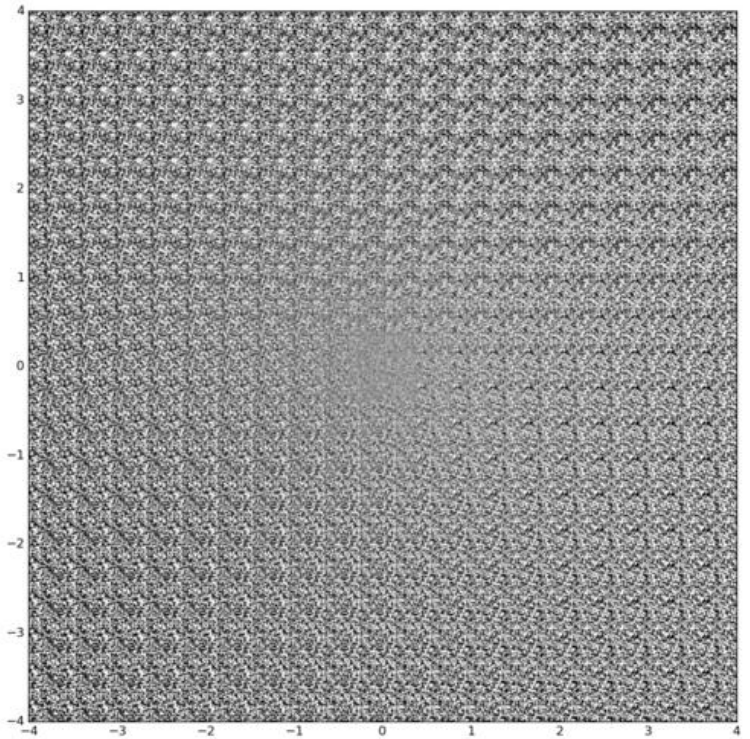
Learns the conditional distribution $P(x|y)$
(this is the simplest possible form of CVAE)

VAE and Representation Learning

VAE Examples - Digits



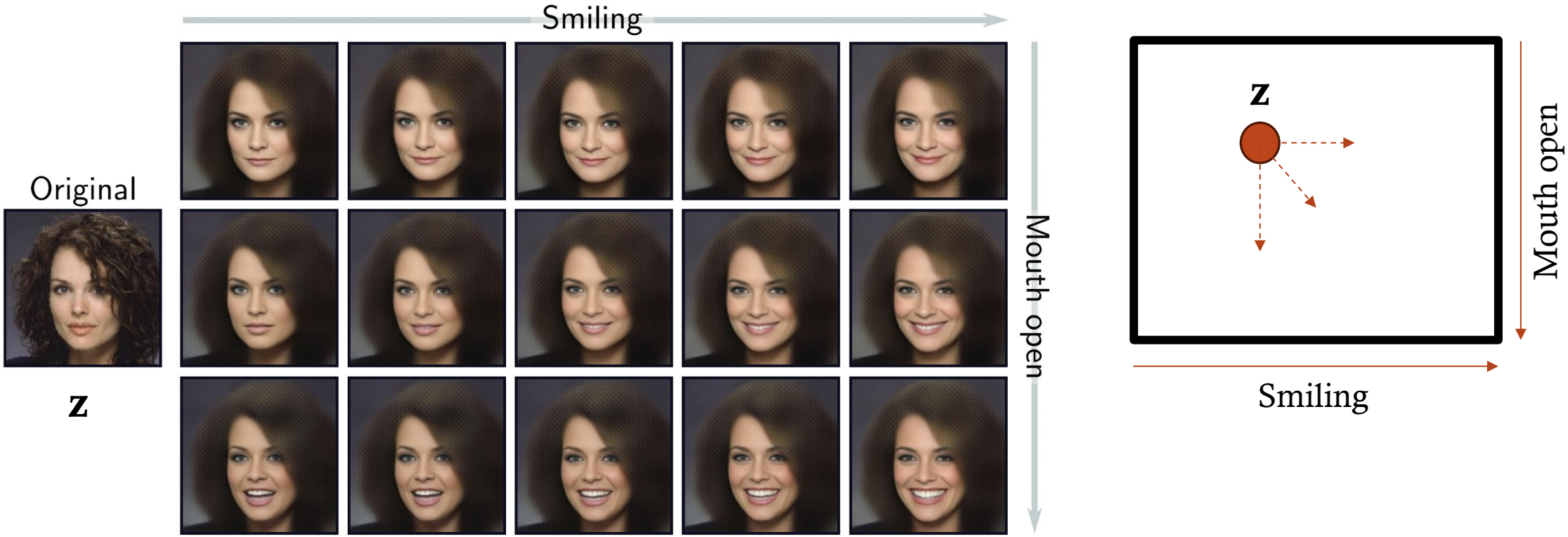
The latent space is organized according to data



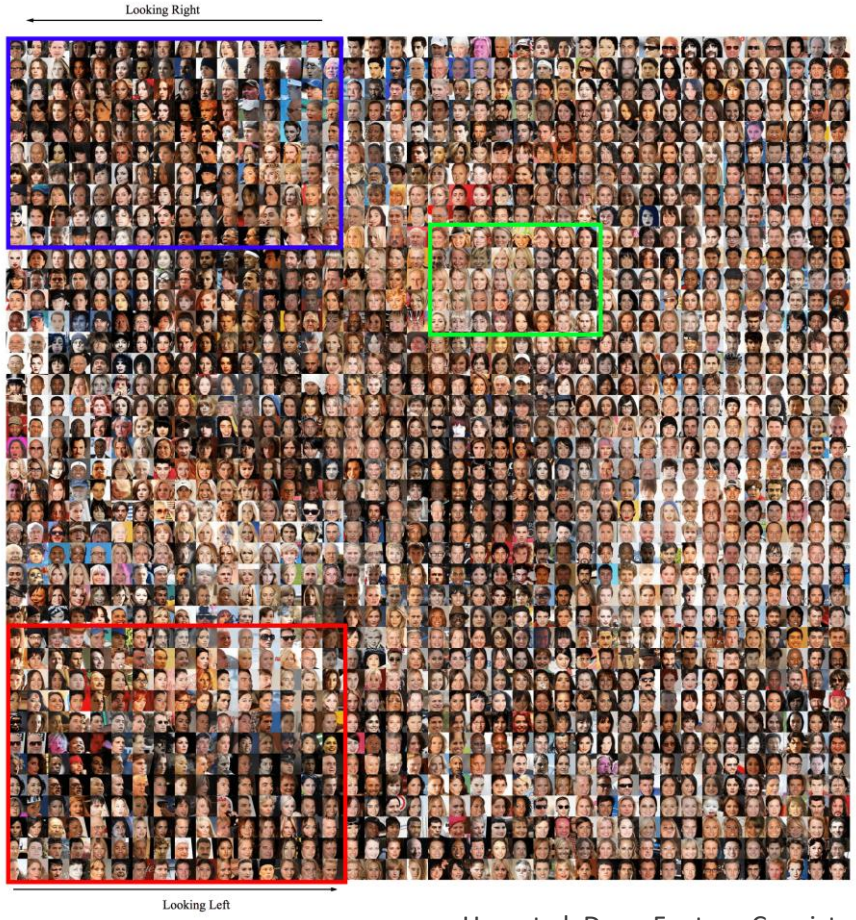
Reconstruction of points sampled from latent space

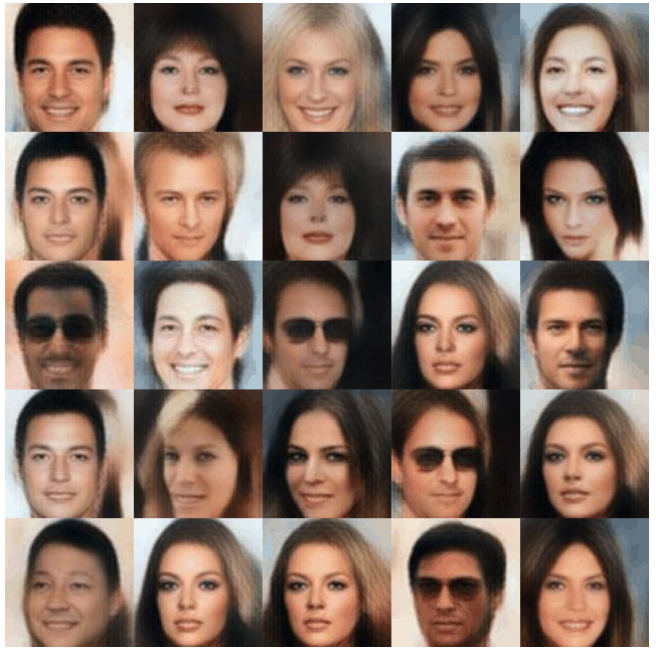
Image credits @ fastfowardlabs.com

Modifying the latent representation along relevant factors of variation



VAE Examples - Faces

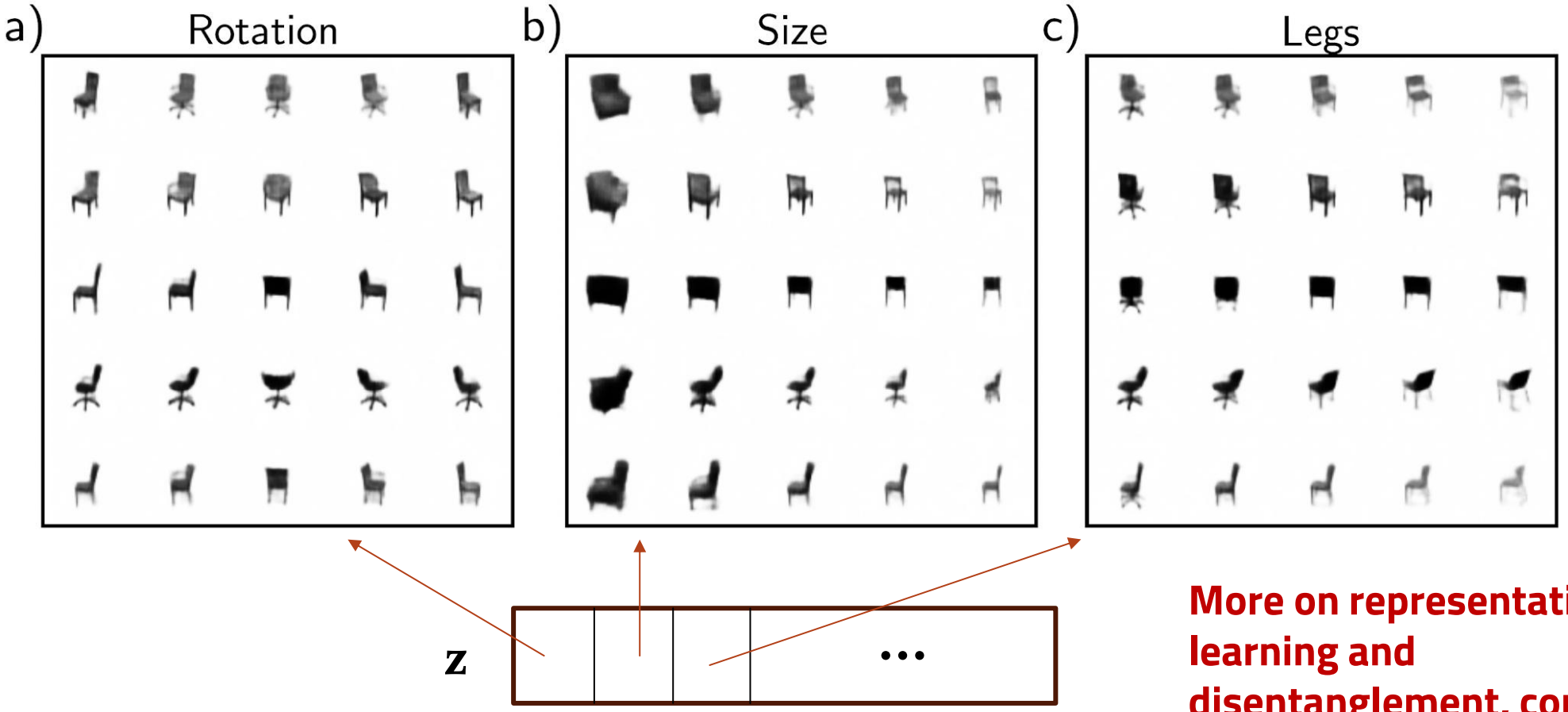




Latent space interpolation

Hou et al, Deep Feature Consistent Variational Autoencoder, 2017

Disentanglement



More on representation learning and disentanglement, coming up soon

Wrap-up

Take Home Messages

- Even the simplest autoencoder hides a probabilistic interpretation
  - Deeper and more modern that we can understand now (more about this to come)
- VAE – Learn complex distributions over latent variables through a variational approximation using neural networks
  - Learns a latent representation useful for inference
  - Not the best model for sample quality..
  - …but the reference model for representation learning

Next Lecture

- Learning a sampling process
- Generative adversarial networks
- Hybrid Variational-Adversarial approaches