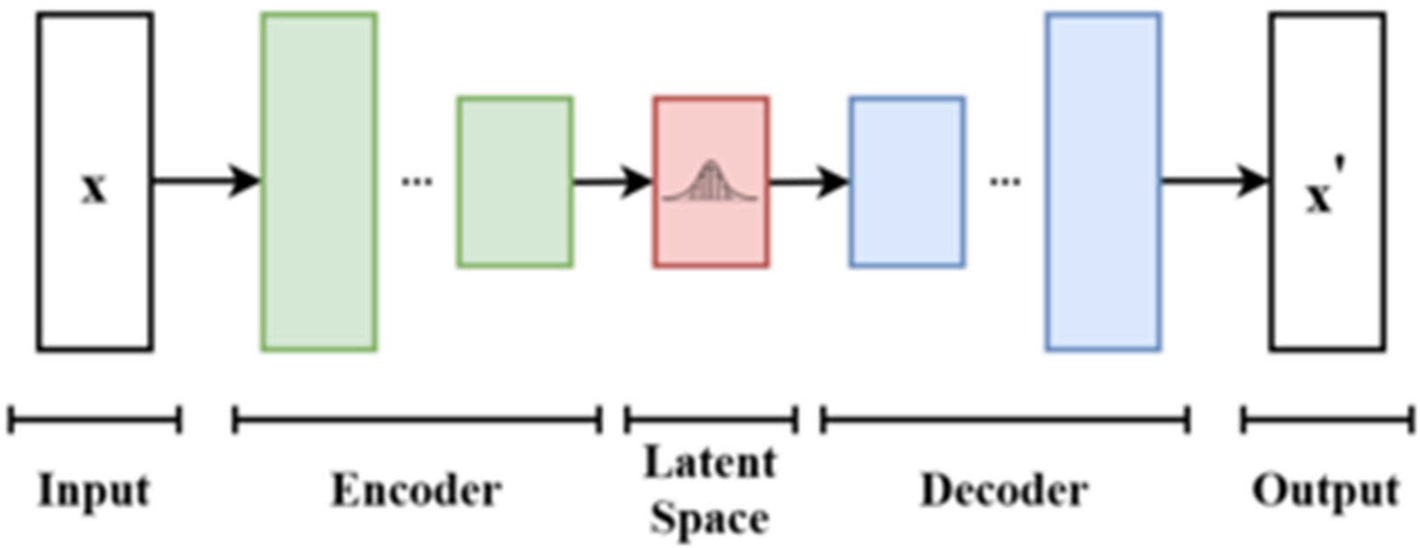
Variational Autoencoders



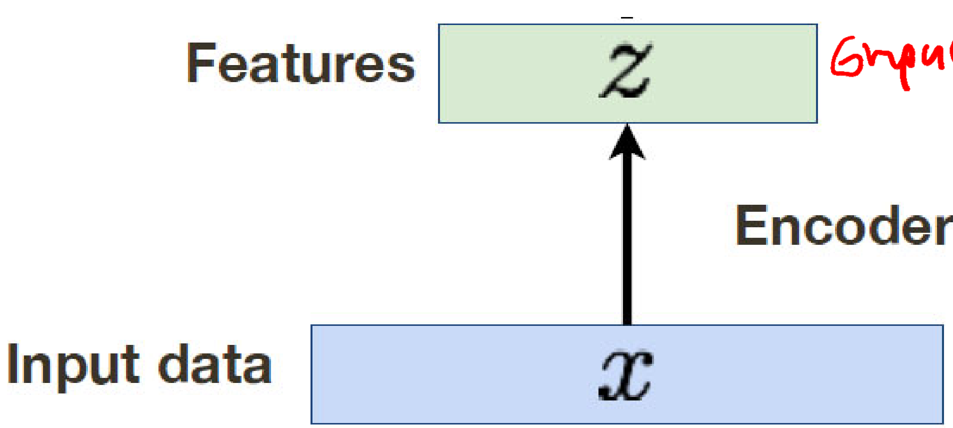
Evangelos Kalogerakis

Generative models
(discussed in this course)

- Autoregressive models [done]
- **Variational Autoencoders**
- Diffusion Models [TODO next week]
- Generative Adversarial Networks [lower priority, after Easter]

~~Variational~~ Autoencoders

Unsupervised approach for learning a lower-dimensional feature representation from **unlabeled** training data

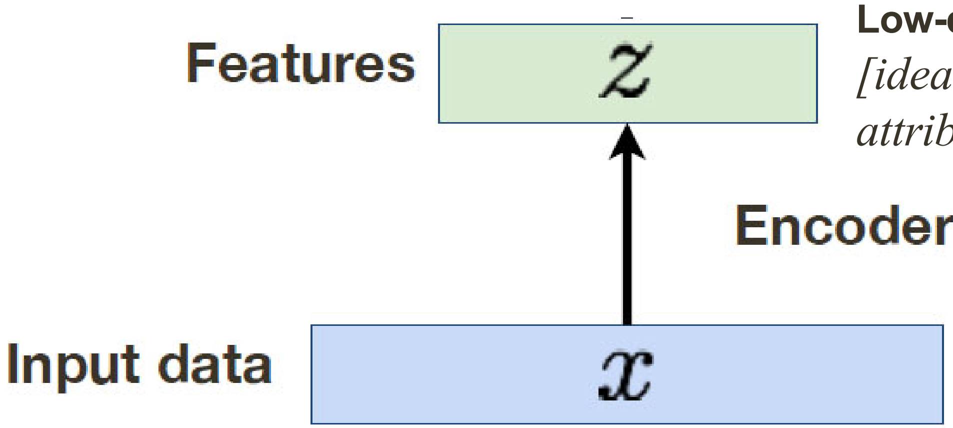


Grpuct

https://www.jeremyjordan.me/autoencoders/

~~Variational~~ Autoencoders

Unsupervised approach for learning a lower-dimensional feature representation from **unlabeled** training data

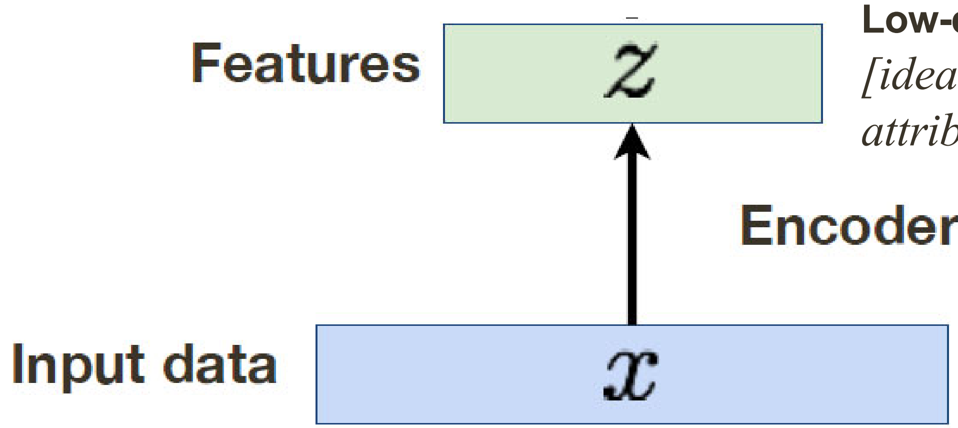


Low-dimensional, fewer dimensions than x
[ideally captures meaningful attributes/modes of variation]

https://www.jeremyjordan.me/autoencoders/

~~Variational~~ Autoencoders

Unsupervised approach for learning a lower-dimensional feature representation from **unlabeled** training data



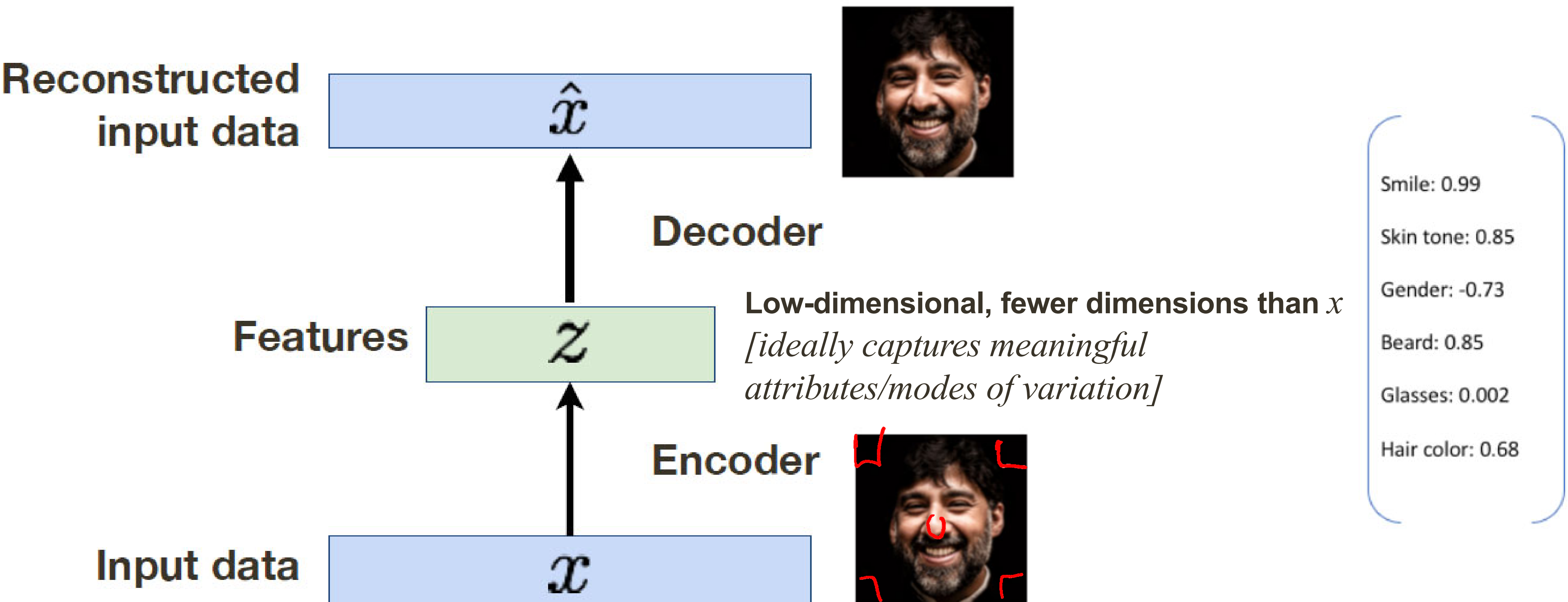
Low-dimensional, fewer dimensions than x
[ideally captures meaningful attributes/modes of variation]

Smile: 0.99
Skin tone: 0.85
Gender: -0.73
Beard: 0.85
Glasses: 0.002
Hair color: 0.68

https://www.jeremyjordan.me/autoencoders/

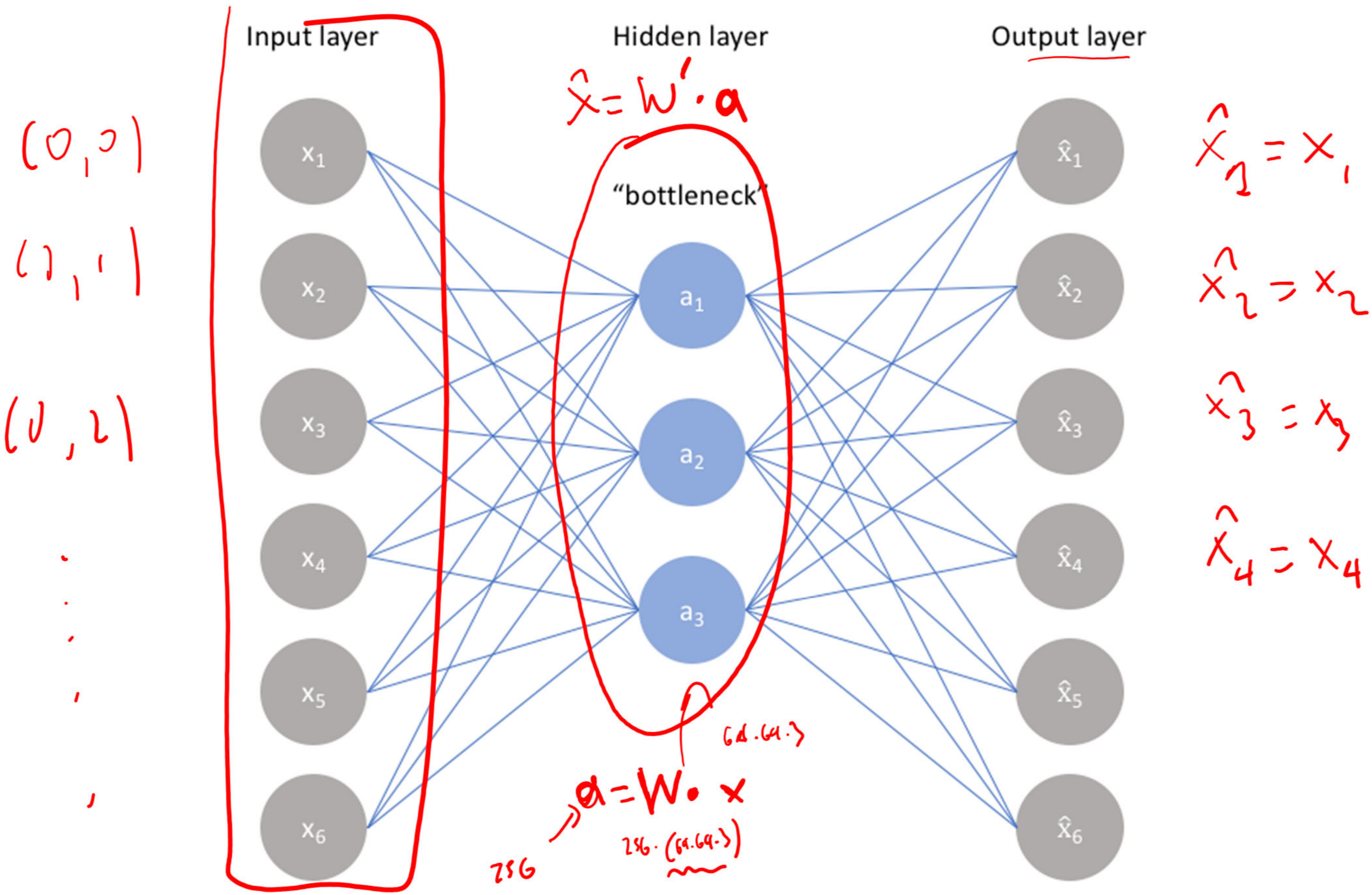
~~Variational~~ Autoencoders

Train such that features can reconstruct original data best they can!



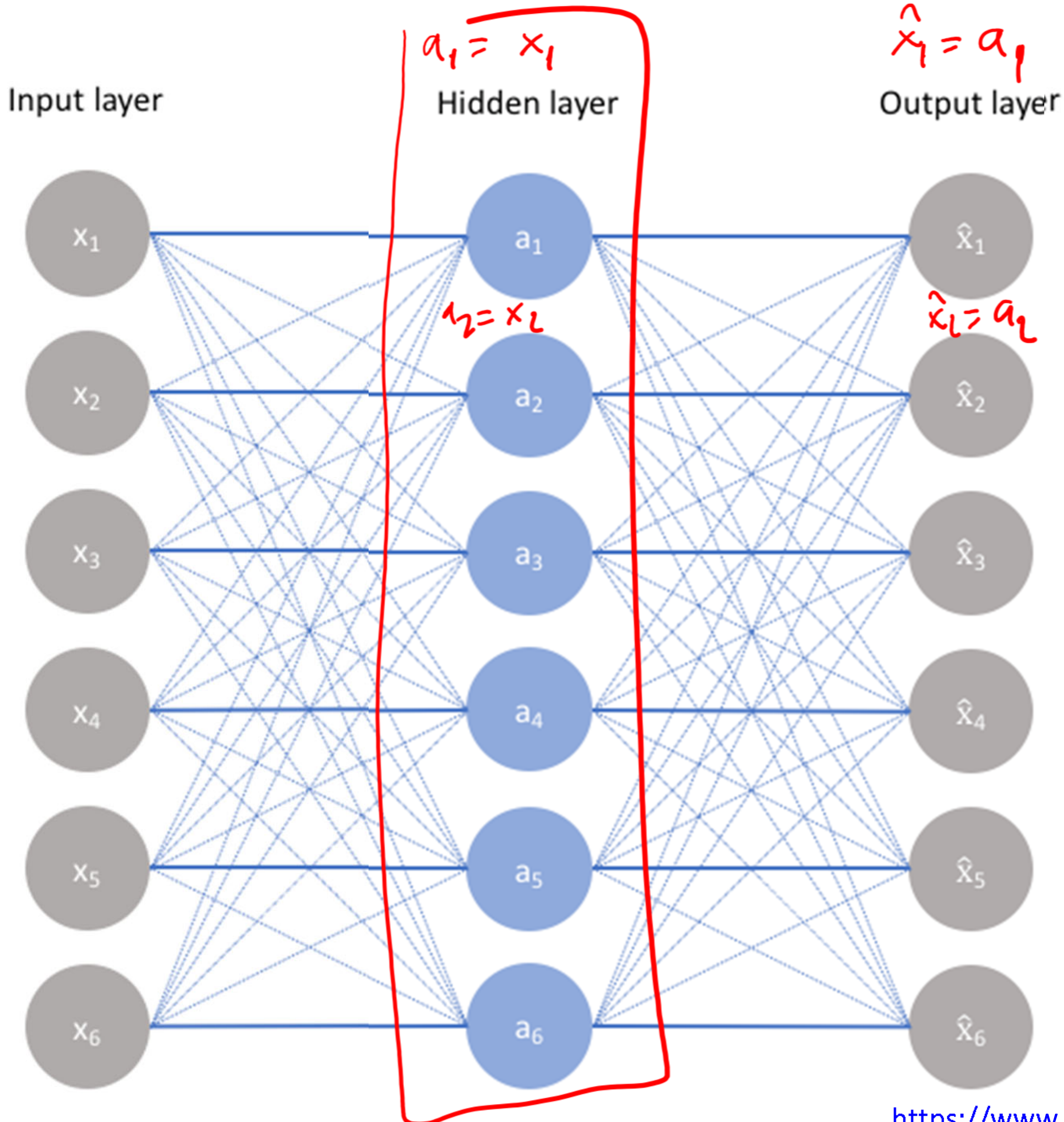
https://www.jeremyjordan.me/autoencoders/

Simplest Autoencoder



https://www.jeremyjordan.me/autoencoders/

Without a compact bottleneck, the network may learn an identity transformation!



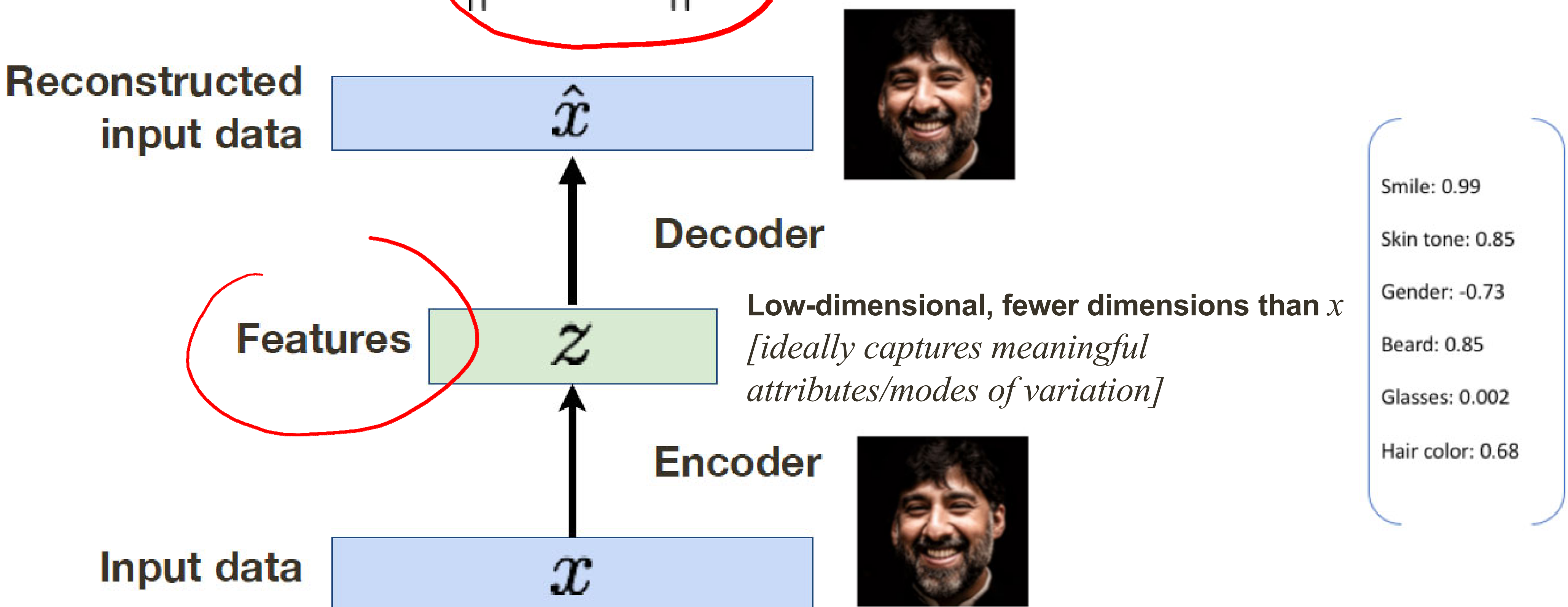
https://www.jeremyjordan.me/autoencoders/

~~Variational~~ Autoencoders

Train such that features can reconstruct original data best they can!

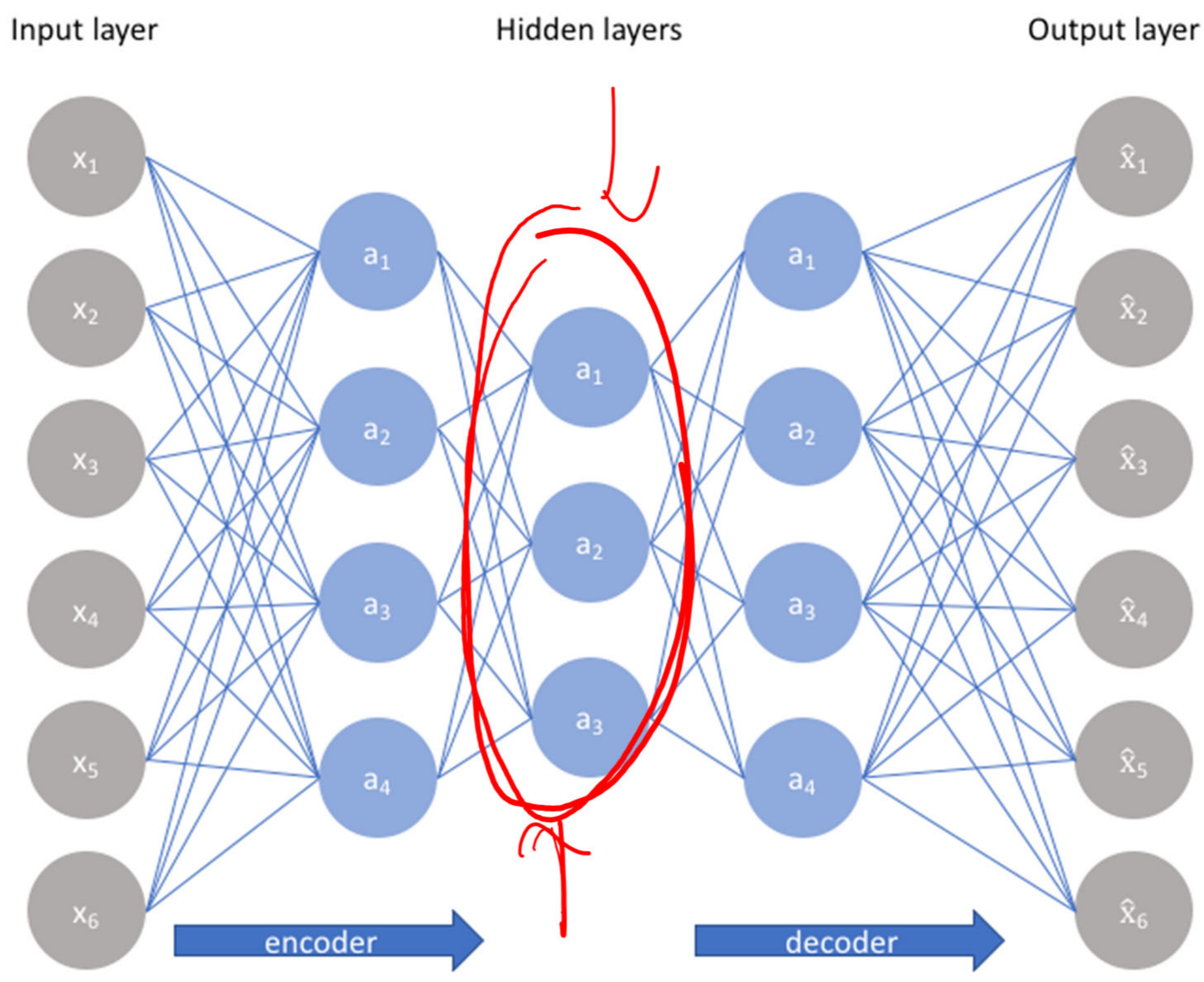
L2 Loss function: self-supervision

$$\|\vec{x} - \hat{\vec{x}}\|^2$$



https://www.jeremyjordan.me/autoencoders/

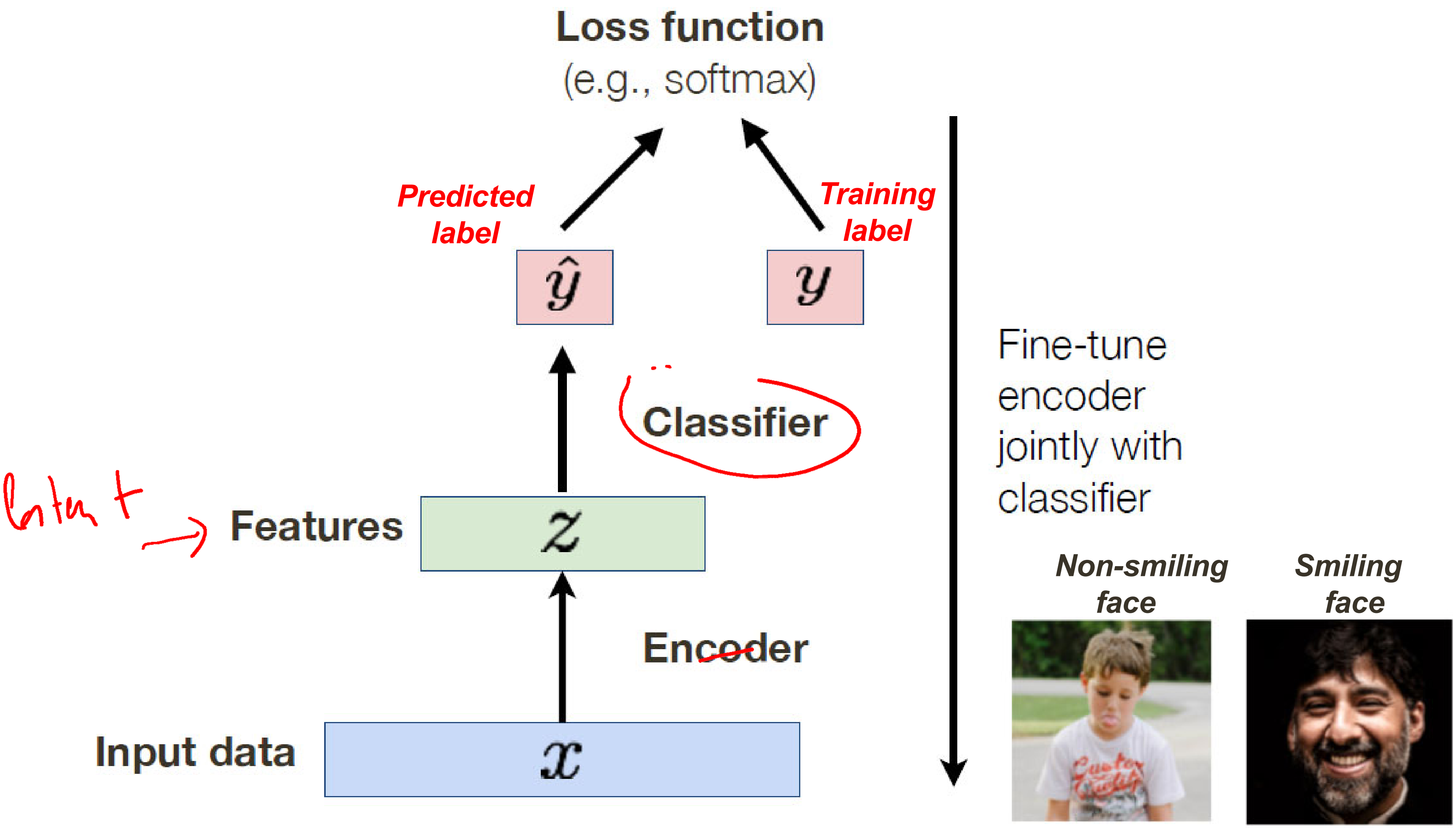
By penalizing the network according to the reconstruction error, our model can learn the most important attributes of the input data and how to best reconstruct the original input from the encoded bottleneck.



What autoencoders are useful for?

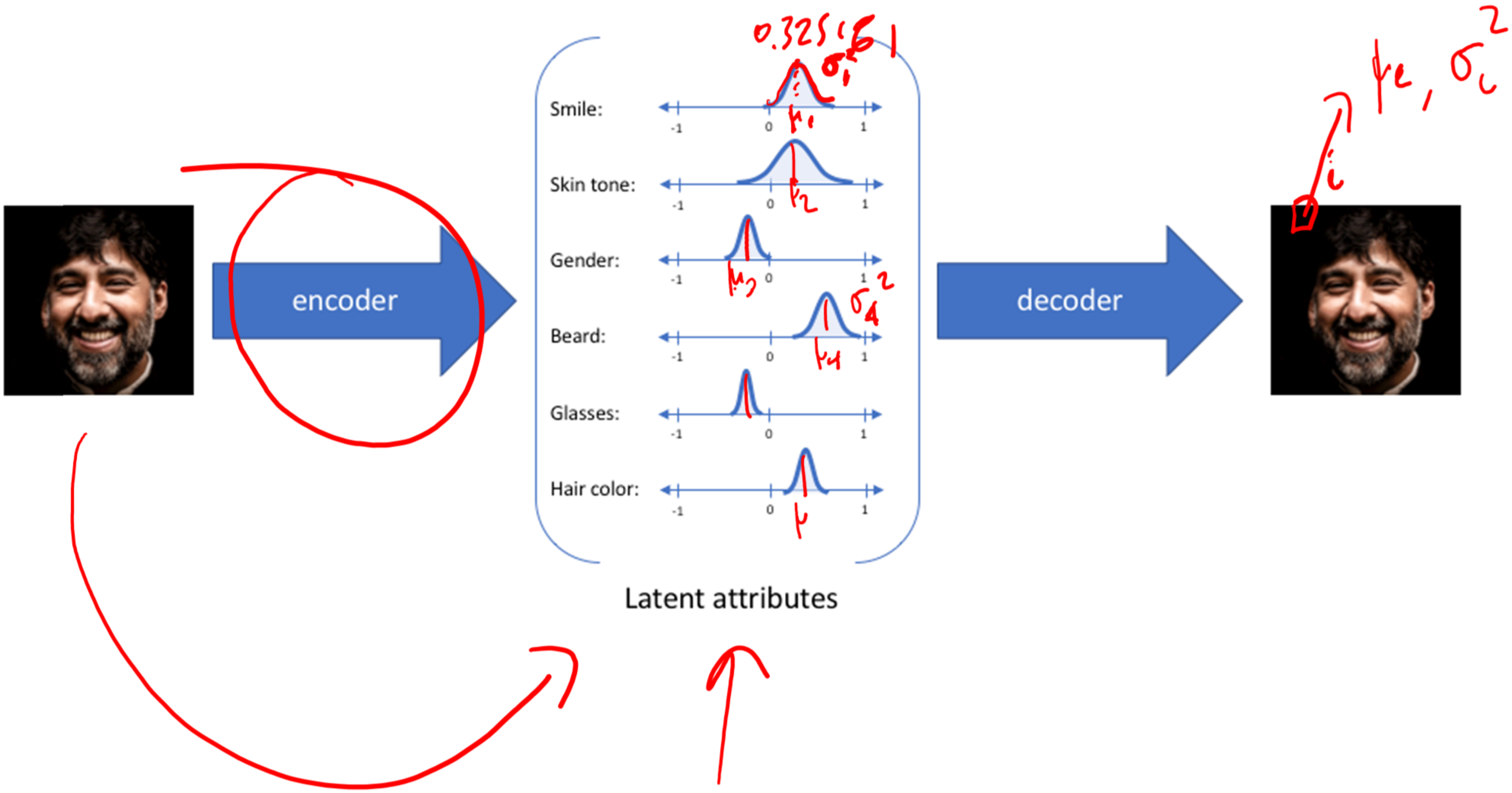
(Non-variational) Autoencoders cannot be used to sample new data!

For recognition: after pre-training with a reconstruction loss, fine-tune encoder for a recognition task with **few amounts of data!**



Variational Autoencoders

The encoder instead outputs a range of possible values (a prob distribution) from which we'll randomly sample to feed into our decoder model.
=> Enforce a continuous, smooth latent space representation.



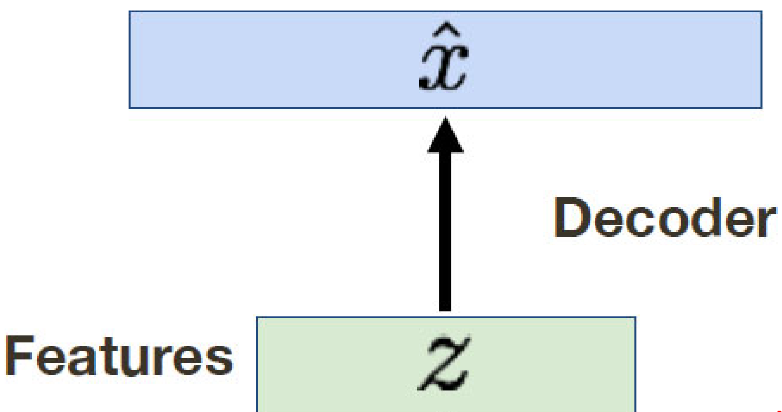
https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoders

Allow us to generate data!

Assume training data is generated from underlying unobserved **latent representation z**

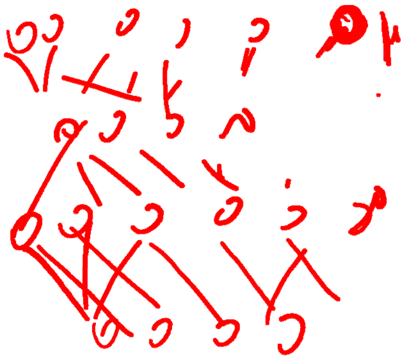
At test time:



Sample from simple distribution
P(z)
(a Gaussian)

$$P(z) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(z-\mu)^2/2\sigma^2}$$

$$\frac{1}{\sqrt{2\pi^D|\Sigma|}} e^{-\frac{1}{2} z^T I z}$$



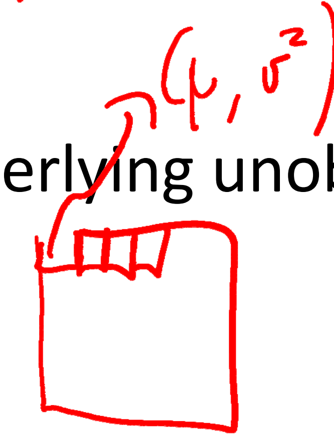
$\mu(z)$ $\sigma^2(z)$ $\max \mathbb{E}[p(x_1) \cdot p(x_2) \cdot p(x_3) \cdot \ldots]$

Variational Autoencoders

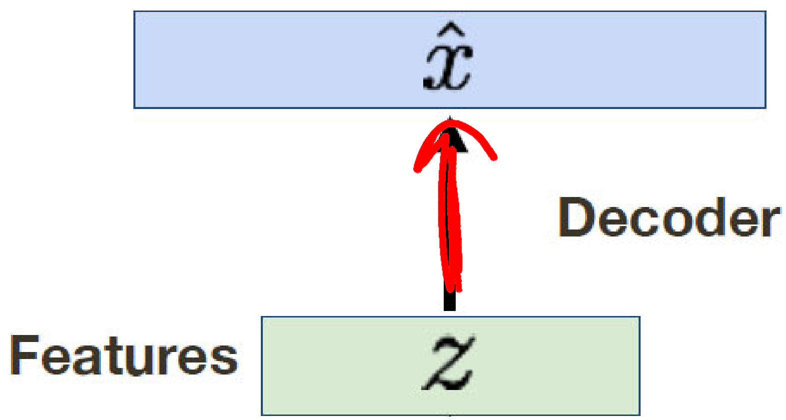
$z = [0.01, -1.5, 0.9, \ldots]$

Allow us to generate data!

Assume training data is generated from underlying unobserved **latent representation z**



At test time: $\cancel{P(x) \cdot P(z|x)}$



Sample from complex cond. distribution
$P(x \mid z)$
(a neural network with learned param θ)

Sample from simple distribution
$P(z)$
(a Gaussian)

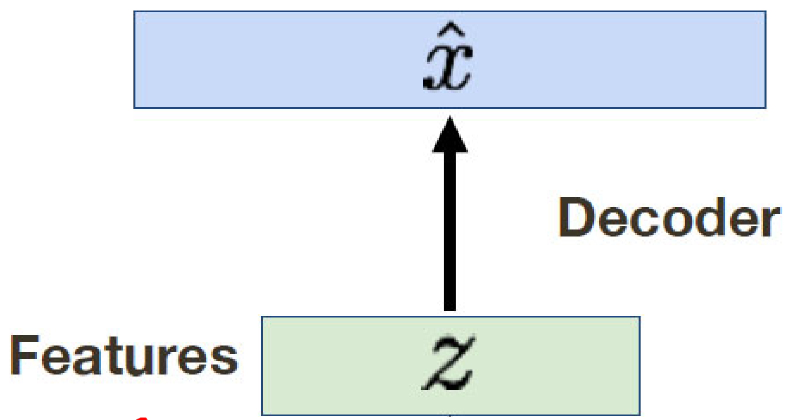
$\rightarrow P(x, z) = \underbrace{P(z)}_{\text{Gaussian}} \cdot \underbrace{P(x|z)}_{\text{Gaussian}}$

Variational Autoencoders

How to train this model?

Maximum likelihood:

$$p_\theta(x_i) = \int_z p_\theta(z)p_\theta(x|z)dz$$



Sample from complex cond. distribution
$P(x \mid z)$
(a neural network with learned param θ)

Sample from simple distribution
$P(z)$
(a Gaussian)

$$p(x) = \int_z p(z) \cdot p(x|z)\, d_z$$

Variational Autoencoders

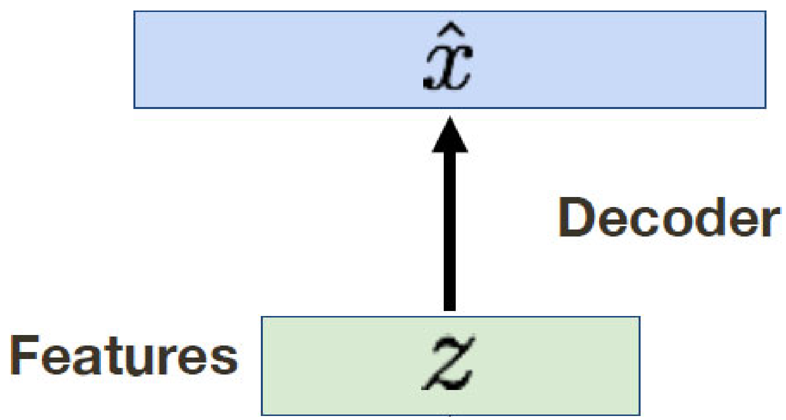
How to train this model?

Maximum likelihood:

$$p_\theta(x) = \int_z \boxed{p_\theta(z)} p_\theta(x|z)dz$$

🙂

Simple **Gaussian** Prior



Sample from complex cond. distribution

$P(x \mid z)$

(a neural network with learned param θ)

Sample from simple distribution

$P(z)$

(a Gaussian)

Variational Autoencoders

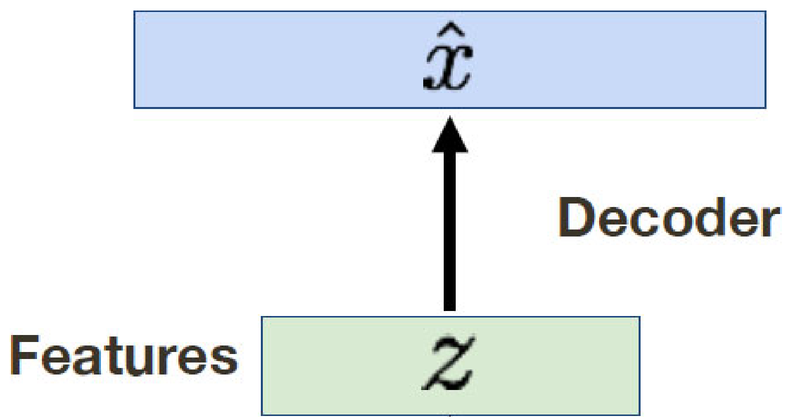
How to train this model?

Maximum likelihood:

Decoder Neural Network 🙂

$$p_\theta(x) = \int_z p_\theta(z)\boxed{p_\theta(x|z)}dz$$

θ



Sample from complex cond. distribution
$P(x \mid z)$
(a neural network with learned param θ)

Sample from simple distribution
$P(z)$
(a Gaussian)

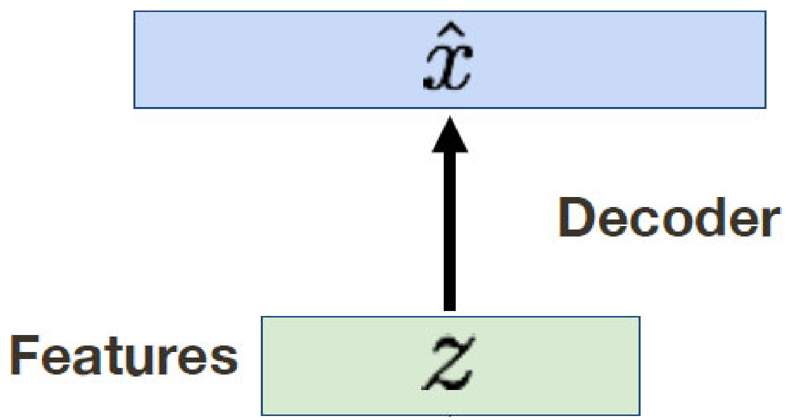
Variational Autoencoders

How to train this model?

Maximum likelihood:

$$\cancel{p_\theta(x)} = \boxed{\int_{\vec{z}}} p_\theta(z) p_\theta(x|z) dz$$

Intractable to compute for every z ☹️



Sample from complex cond. distribution
$P(x \mid z)$
(a neural network with learned param θ)

Sample from simple distribution
$P(z)$
(a Gaussian)

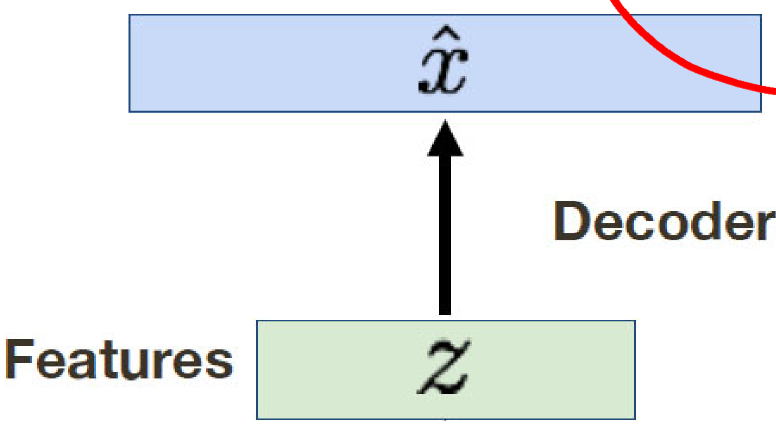
Variational Autoencoders

How to train this model?

Maximum likelihood:

$$p_\theta(x) = \boxed{\int_z} p_\theta(z)p_\theta(x|z)dz$$

Intractable to compute for every z



Sample from complex cond. distribution
$P(x \mid z)$
(a neural network with learned param θ)

Sample from simple distribution
$P(z)$
(a Gaussian)

latent | data

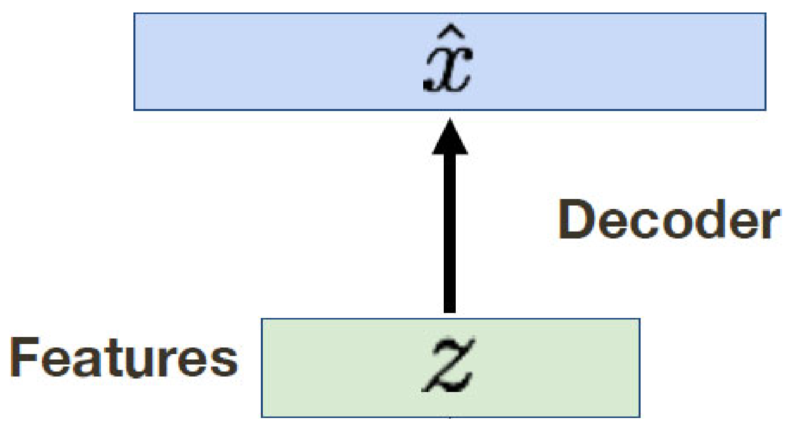
Posterior density is also intractable: $p_\theta(z|x) = p_\theta(x|z)p_\theta(z) / \boxed{p_\theta(x)}$

Variational Autoencoders

How to train this model?

Maximum likelihood: $$p_\theta(x) = \boxed{\int_z} p_\theta(z) p_\theta(x|z) dz$$

Intractable to compute for every z



Sample from complex cond. distribution $P(x \mid z)$ (a neural network with learned param θ)

Sample from simple distribution $P(z)$ (a Gaussian)

Solution: approximate $p_\theta(z \mid x)$ with a tractable distribution $\boxed{q_\varphi(z \mid x)}$

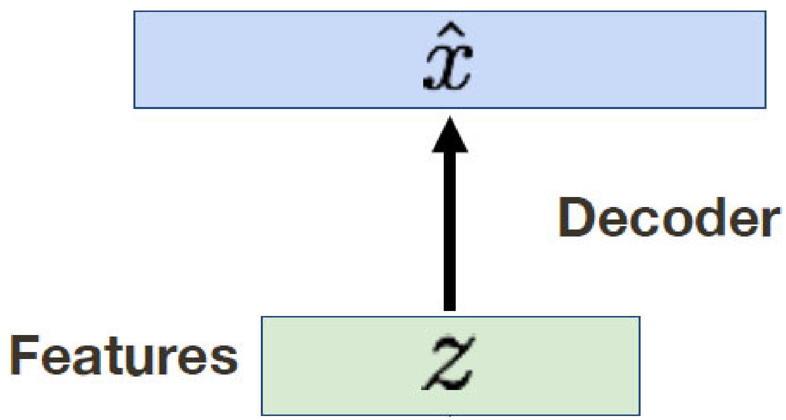
Posterior density is also intractable: $p_\theta(z|x) = p_\theta(x|z)p_\theta(z)/p_\theta(x)$

Variational Autoencoders

How to train this model?

Maximum likelihood: $$p_\theta(x) = \boxed{\int_z} p_\theta(z) p_\theta(x|z) dz$$

Intractable to compute for every z



Sample from complex cond. distribution
$P(x \mid z)$
(a neural network with learned param θ)

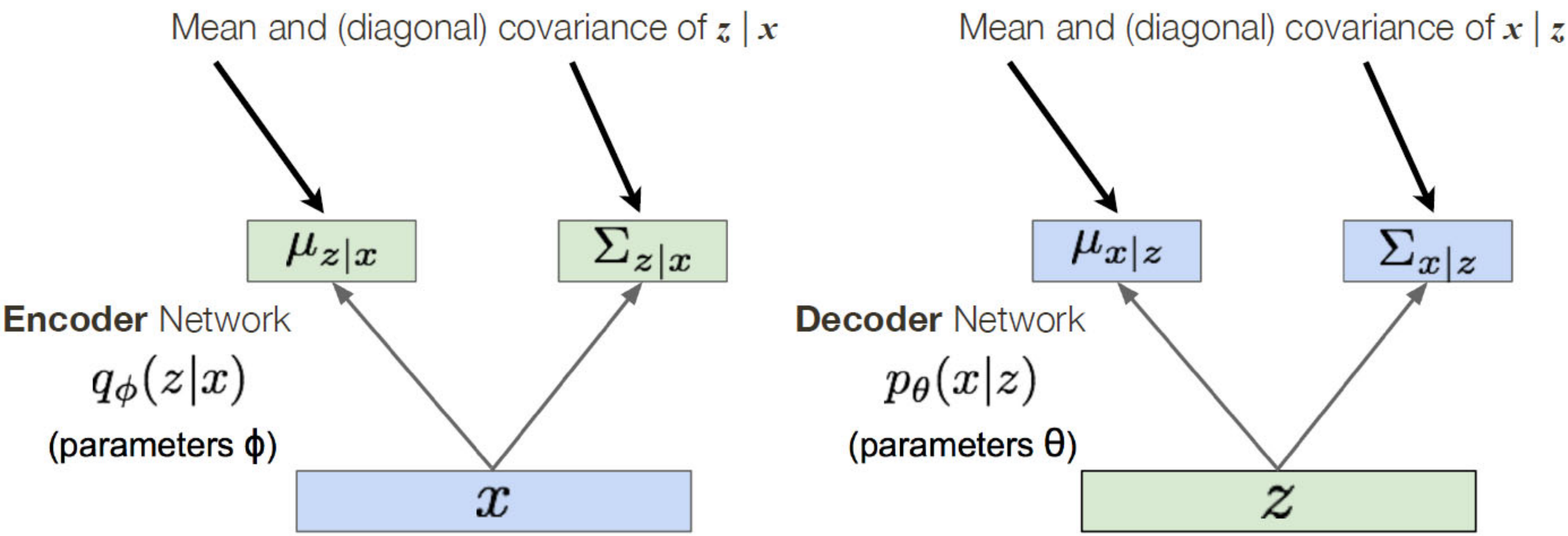
Sample from simple distribution
$P(z)$
(a Gaussian)

Solution: approximate $p_\theta(z \mid x)$ with a **neural network** $q_\varphi(z \mid x)$ **[encoder]**

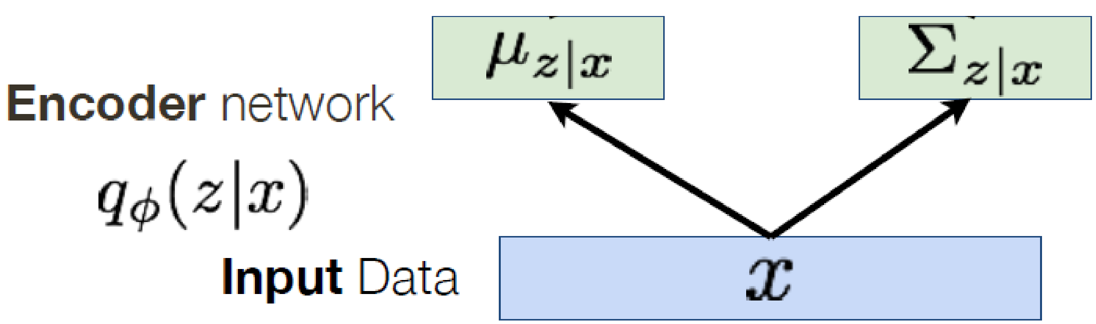
Posterior density is also intractable: $p_\theta(z|x) = p_\theta(x|z)p_\theta(z)/p_\theta(x)$

Variational Autoencoders

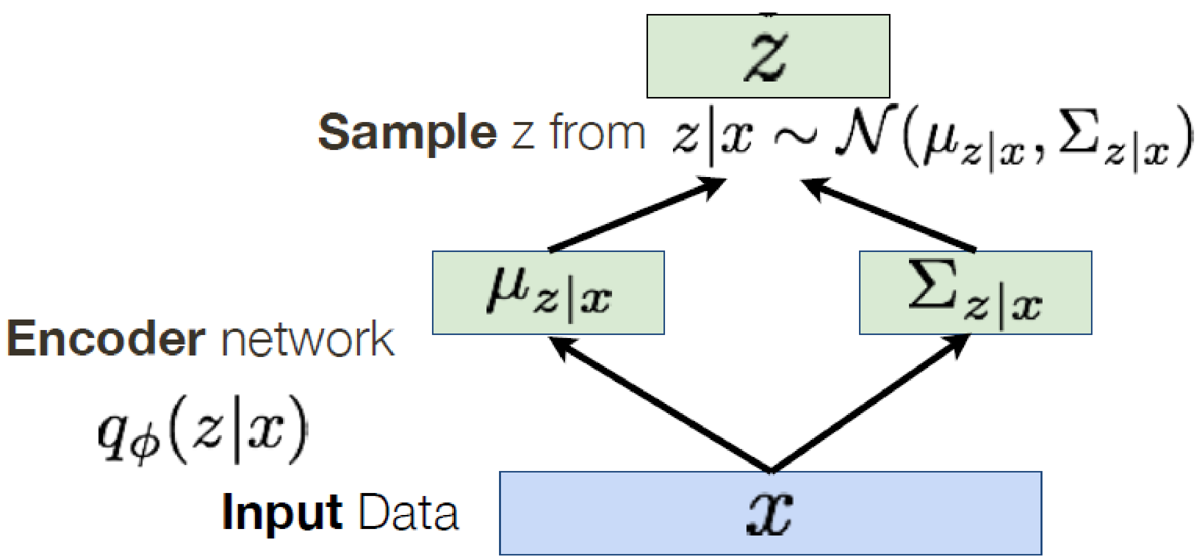
Since we are modeling probabilistic generation of data, encoder and decoder networks are probabilistic (they model Gaussian distributions)



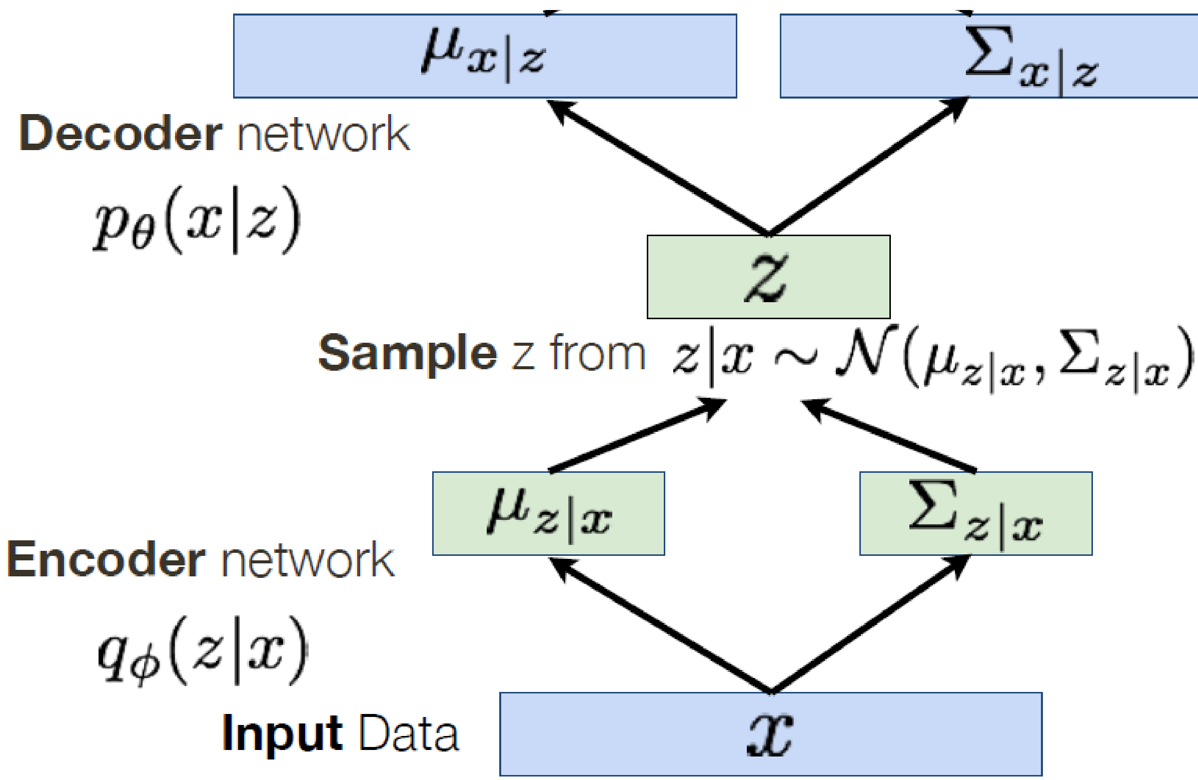
Forward pass during training



Forward pass during training

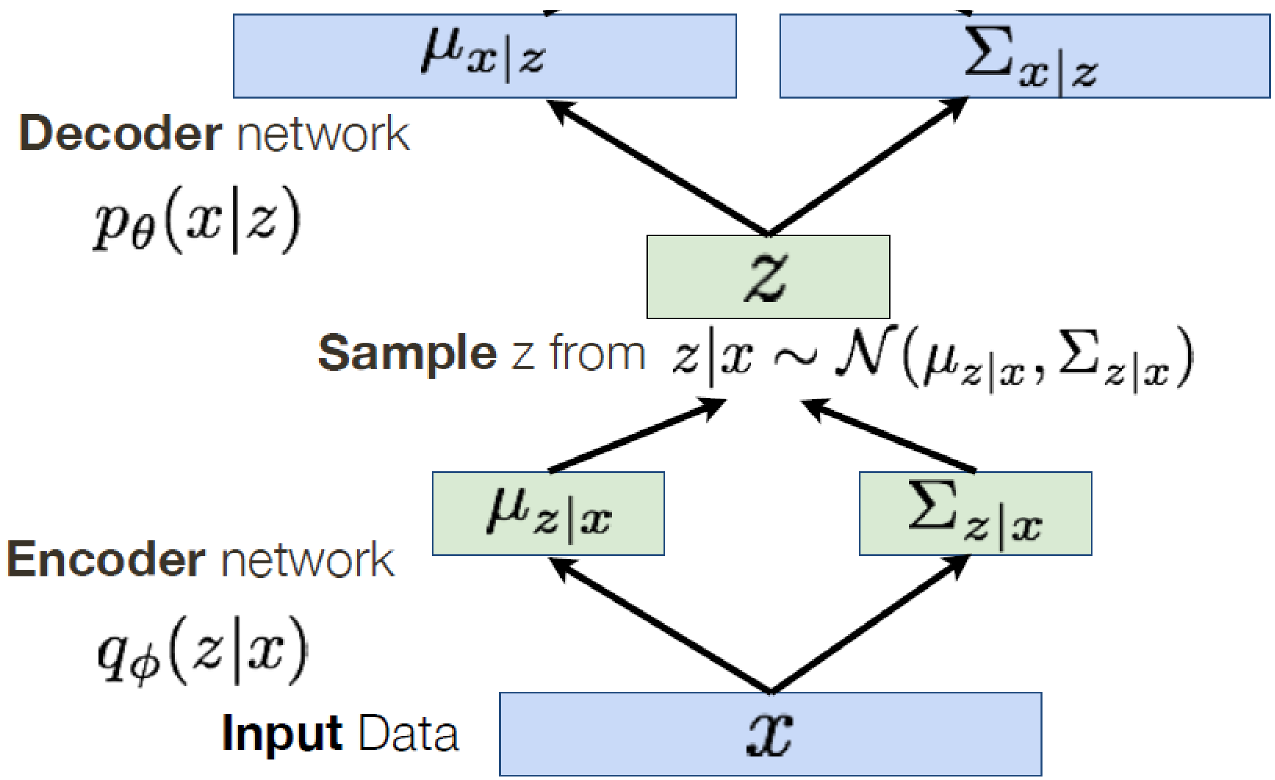


Forward pass during training



VAE Loss

$$-\mathbf{E}_{z\sim q_\phi(z|x)} \log\left(p_\theta(x|z)\right) + KL\left(q_\phi(z|x) \| p_\theta(z)\right)$$



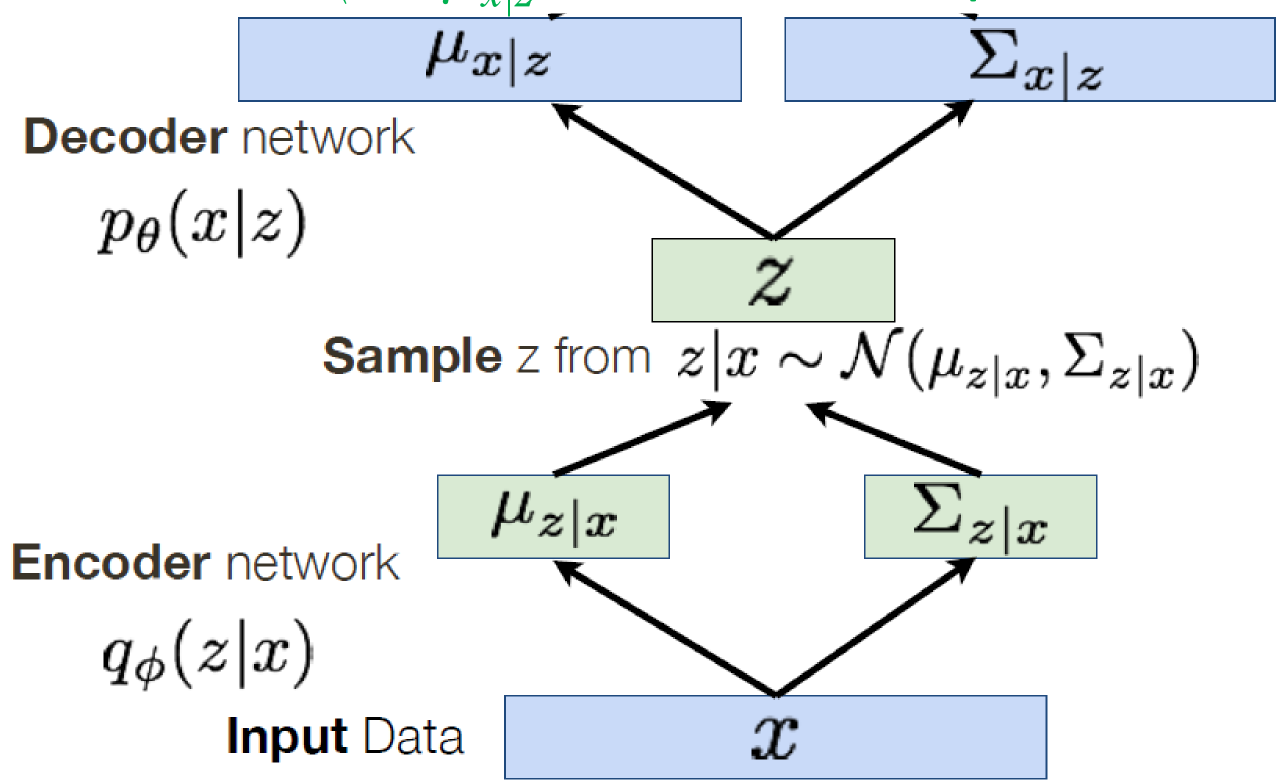
VAE Loss

$$\boxed{-\mathbf{E}_{z\sim q_\phi(z|x)} \log\left(p_\theta\left(x|z\right)\right)} + KL\left(q_\phi\left(z|x\right)\|p_\theta(z)\right)$$

Reconstruction Loss

(outputs should be as close as possible to input)

reduces to $(x - \mu_{x|z})^2$ for fixed output covariance

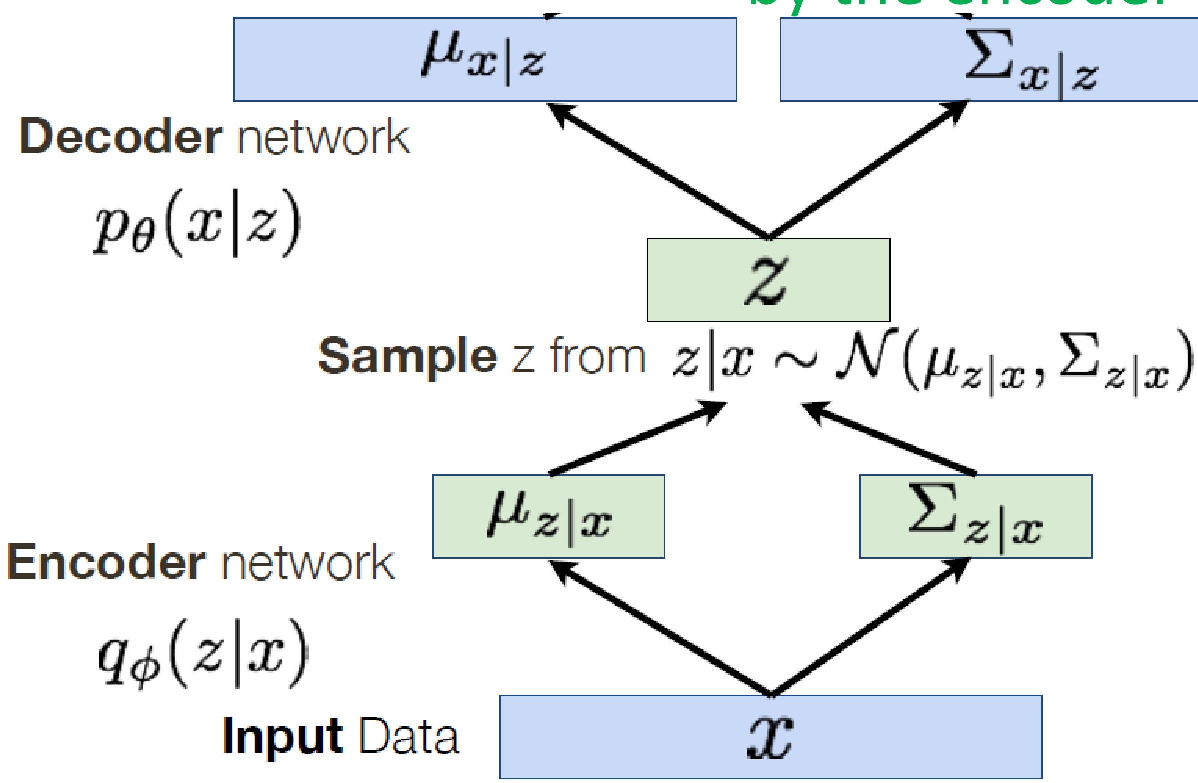


VAE Loss

$$-\mathbf{E}_{z\sim q_\phi(z|x)}\log\left(p_\theta\left(x|z\right)\right)+\boxed{KL\left(q_\phi\left(z|x\right)\|p_\theta(z)\right)}$$

Regularization term

Make distribution of the latent space produced by the encoder close to a standard Gaussian.



KL divergence

A measure of how one probability distribution is different from a second one:

$$KL\left(q_\phi\left(z|x\right)\|p_\theta(z)\right)=\int_z q_\phi\left(z|x\right)\log\frac{q_\phi\left(z|x\right)}{p_\theta(z)}$$

In our case, we want our latent space $p_\theta(z)$ to be $N(0, I)$

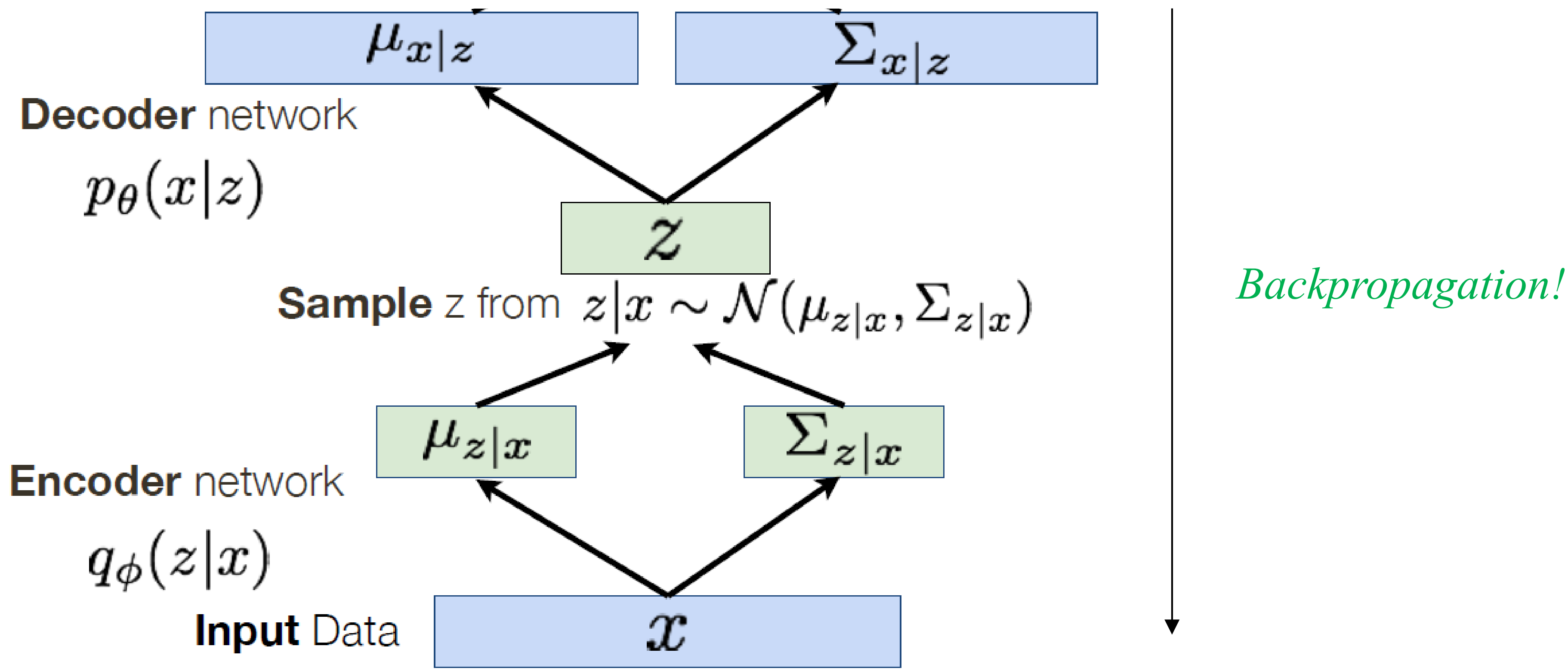
https://medium.com/data-science/light-on-math-machine-learning-intuitive-guide-to-understanding-kl-divergence-2b382ca2b2a8

VAE Loss

$$-\mathbf{E}_{z\sim q_\phi(z|x)} \log\left(p_\theta\left(x|z\right)\right) + KL\left(q_\phi\left(z|x\right)\|p_\theta(z)\right)$$

$$\lambda\,(x-\mu_{x|z})^2 + \sum_{d=1}^{D}(\sigma^2_{z|x}[d] + \mu^2_{z|x}[d] - \log\sigma_{z|x}[d] - 1)$$

(where λ is a weighting term)

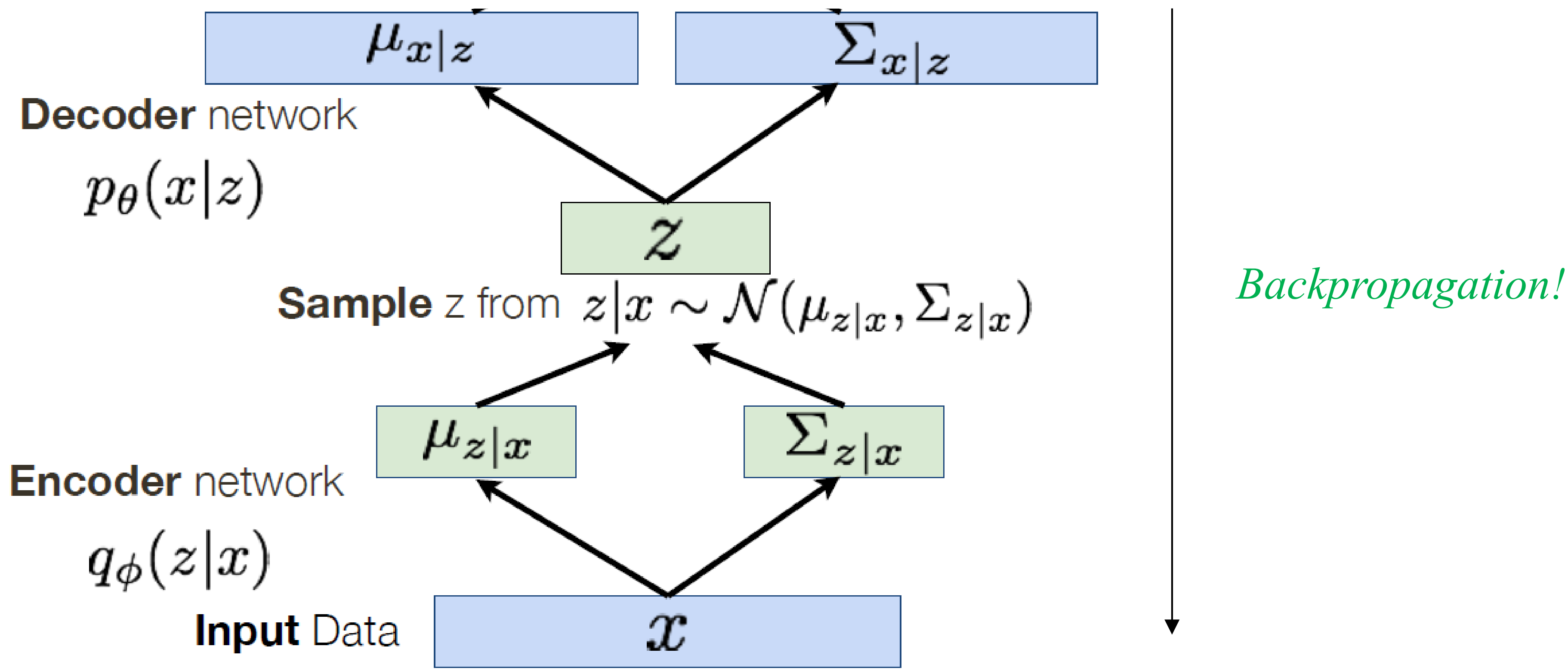


VAE Loss (skipping proofs...)

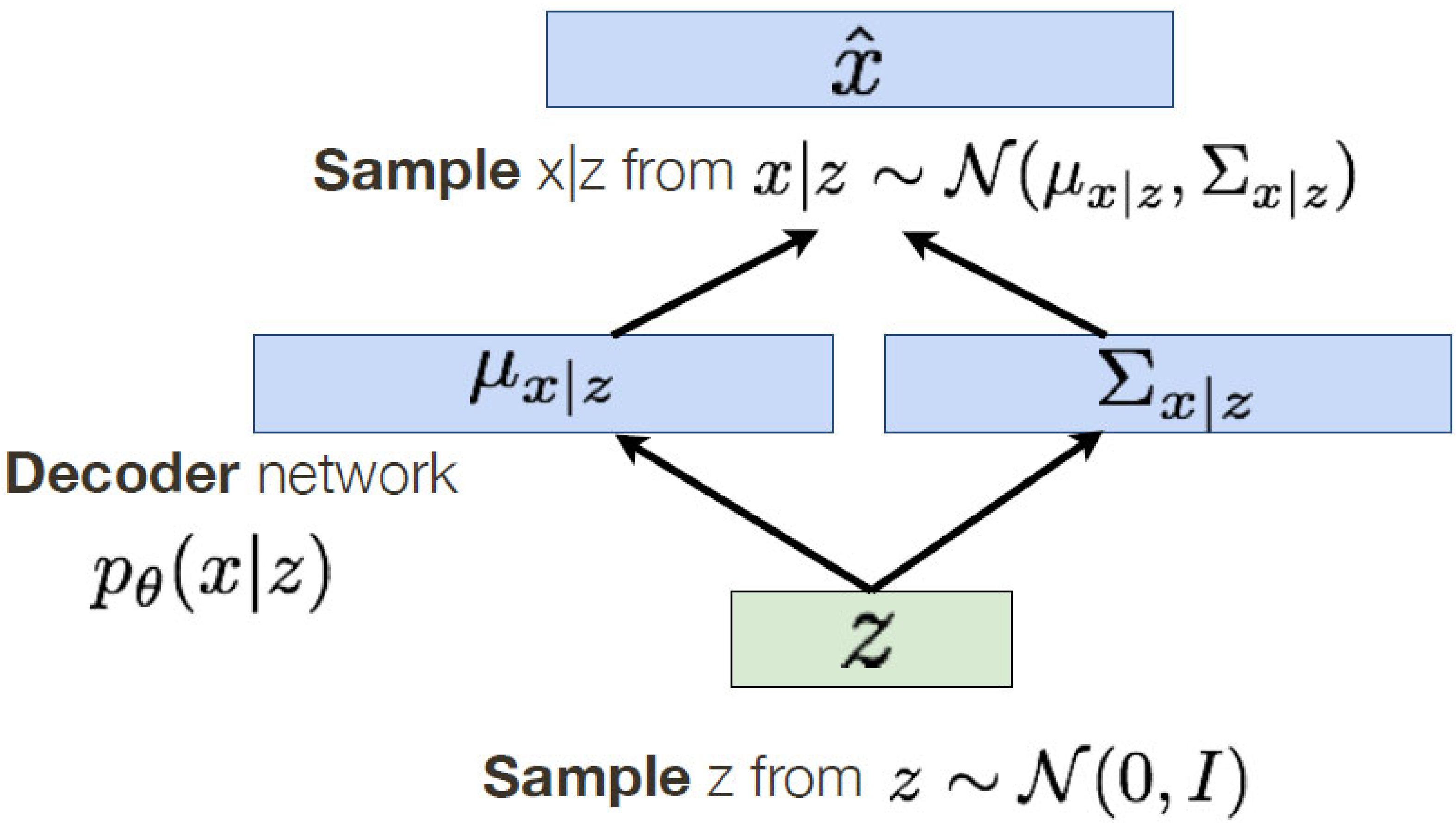
$$-\mathbf{E}_{z\sim q_\phi(z|x)} \log\left(p_\theta\left(x|z\right)\right) + KL\left(q_\phi\left(z|x\right)\|p_\theta(z)\right)$$

Minimize upper bound on loss we care about!

$$\geq -\log p_\theta(x)$$



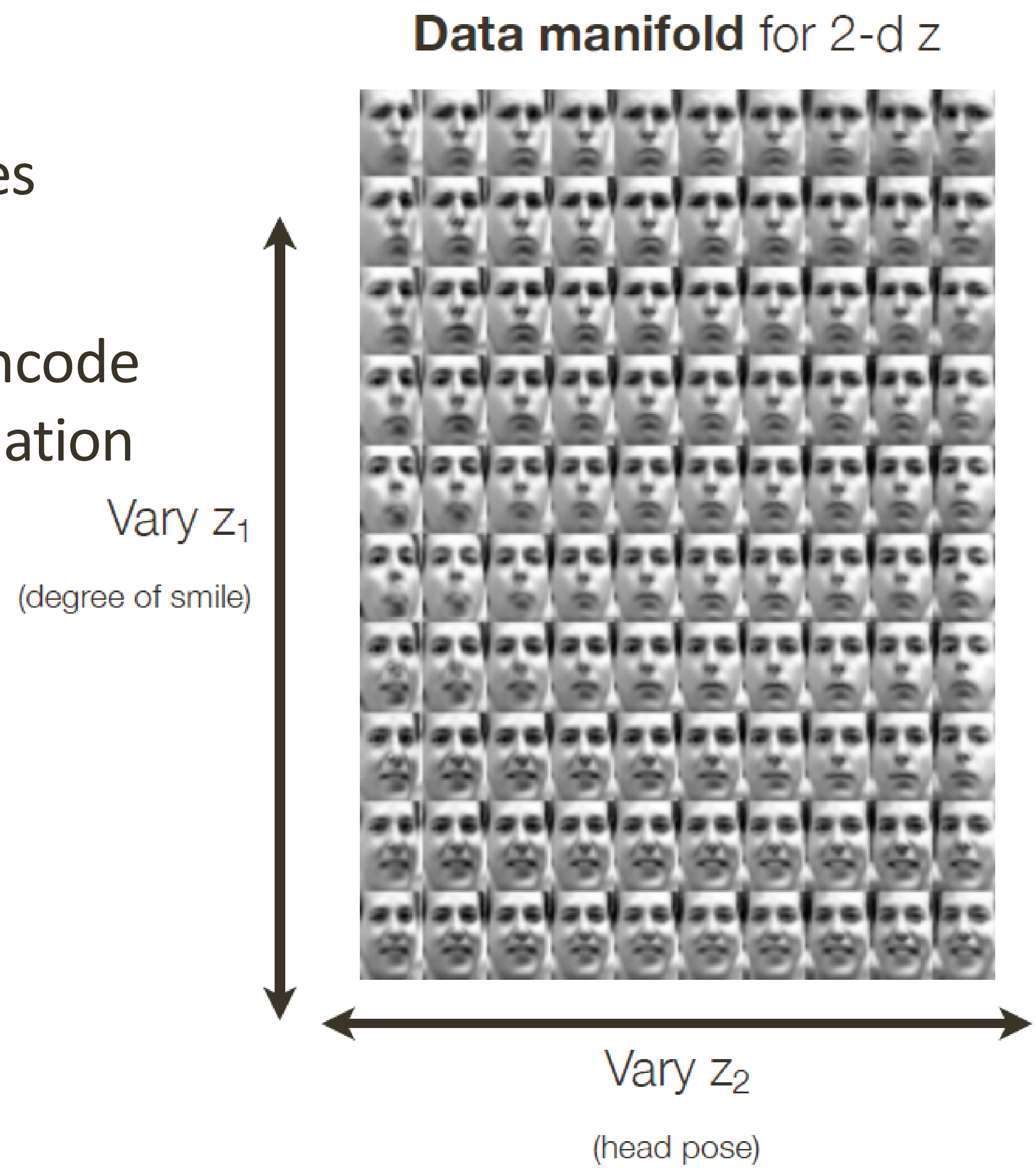
Test time


$\hat{x}$
Sample x|z from $x|z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z})$
$\mu_{x|z}$
$\Sigma_{x|z}$
Decoder network
$p_\theta(x|z)$
z
Sample z from $z \sim \mathcal{N}(0, I)$

VAE Latent space

- Diagonal prior on z => independent latent variables

- Different dimensions of z encode interpretable factors of variation



VAE useful literature

- **Understanding Variational Autoencoders:** **https://www.jeremyjordan.me/variational-autoencoders/**

- **Tutorial on Variational Autoencoders**

  https://arxiv.org/pdf/1606.05908.pdf

- **Today VAEs are mostly used to produce a low-dimensional latent space of data – latent diffusion models operate on this space…**