ECE 57000

Generative models & Variational Autoencoder

Chaoyue Liu

Fall 2024

Generative models

Generative models:

machine learning models that aim to **learn the underlying patterns/distributions** of data, in order to **generate new, similar data**

topics include:

- Variational Autoencoder (VAE)
- Generative Adversarial Network (GAN)
- Diffusion model [time permitting]

Motivations

Why study generative models?

- Sketching realistic photos

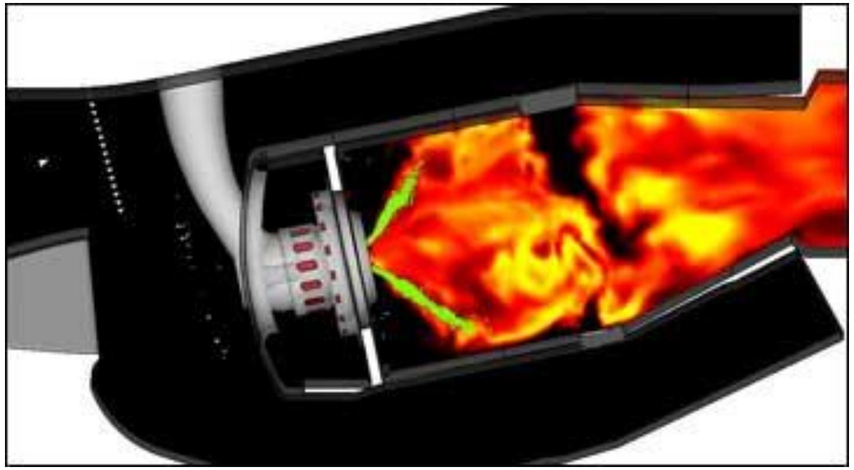
- Style transfer

- Super resolution

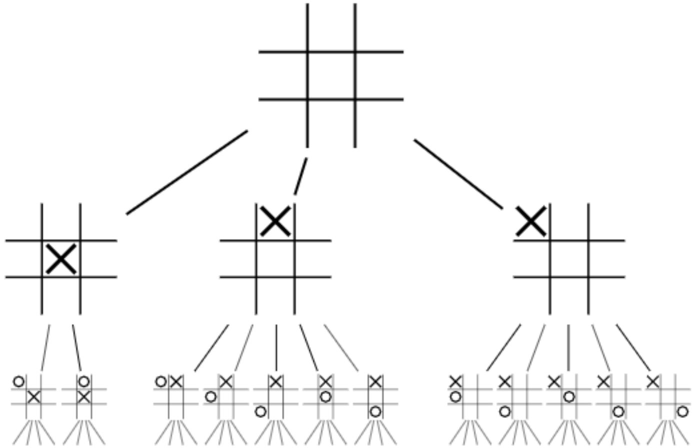
Motivations

<u>Why</u> study generative models?

- Emulate complex physics simulations to be faster



- Reinforcement learning - Attempt to model the real world so we can simulate possible futures



Highlights

StyleGAN

Picture: *These people are not real – they were produced by our generator that allows control over different aspects of the image.*

source: https://arxiv.org/abs/1406.2661

Highlights

DALL-E

Source: OpenAI official Website

Highlights

Stable Diffusion

Source: https://parental-control.flashget.com/how-do-you-use-stable-diffusion

Generative models

We have a training set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{n}$ (without labels), which are i.i.d. sampled from an unknown distribution $P(\mathbf{x})$

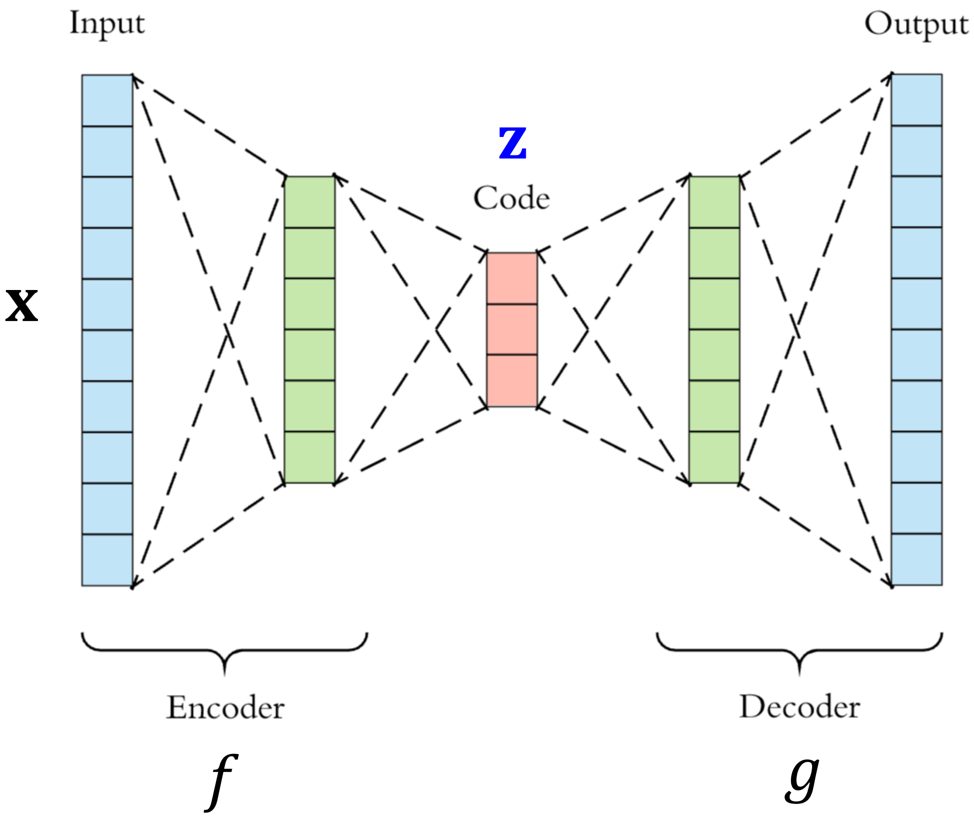
Ideally, we would like to recover/learn distribution $P(\mathbf{x})$ from the training set $\mathcal{D}$, and sample from $P(\mathbf{x})$ to generate new data.
--- Gaussian Mixture model

Alternatively, we would like to generate new data so that its distribution matches $P(\mathbf{x})$

Key: the model learns underlying pattern or distribution

Autoencoder

Autoencoders map an input to a latent code (**encoder**) and map this latent code back to the input (**decoder**)



$$\tilde{\mathbf{x}} = g\big(f(\mathbf{x})\big)$$

Autoencoder

The optimization problem is to fit the encoder and decoder simultaneously to reconstruct output

- More formally, the autoencoder objective is:

$$\min_{f,g} \mathbb{E}[L(\mathbf{x}, \tilde{\mathbf{x}})]$$

$$\min_{f,g} \mathbb{E}\left[L\left(\mathbf{x}, g\big(f(\mathbf{x})\big)\right)\right]$$

- One example is using Mean Squared Error loss

$$\min_{f,g} \mathbb{E}\left[\left\|\mathbf{x} - g\big(f(\mathbf{x})\big)\right\|_2^2\right]$$

Autoencoder

If there are no constraints on the encoder and decoder than the identity function works perfectly…

- Suppose $f(\mathbf{x}) = \mathbf{x}$ and $g(\mathbf{x}) = \mathbf{x}$

- Then we know that

$$\min_{f,g} \mathbb{E}\left[\left\|\mathbf{x} - g\big(f(\mathbf{x})\big)\right\|_2^2\right]$$
$$= \min_{f,g} \mathbb{E}[\|\mathbf{x} - \mathbf{x}\|_2^2] = 0$$

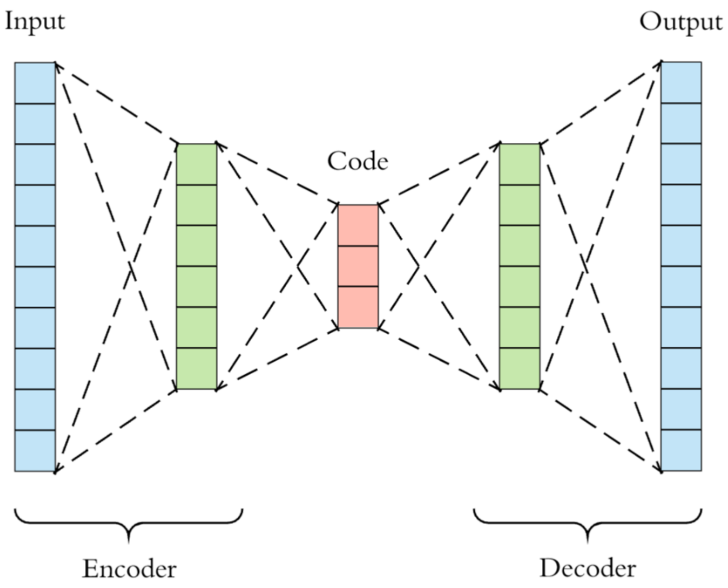
  - And since all terms are positive, this is the global minimum

- Trivial/useless…What can we do?

Autoencoder

Adding constraints to f, g or $\mathbf{z}$ can often produce interesting properties of $\mathbf{z}$

- Undercomplete autoencoders assume that the latent space has lower dimension, i.e., $k < d$



Autoencoder

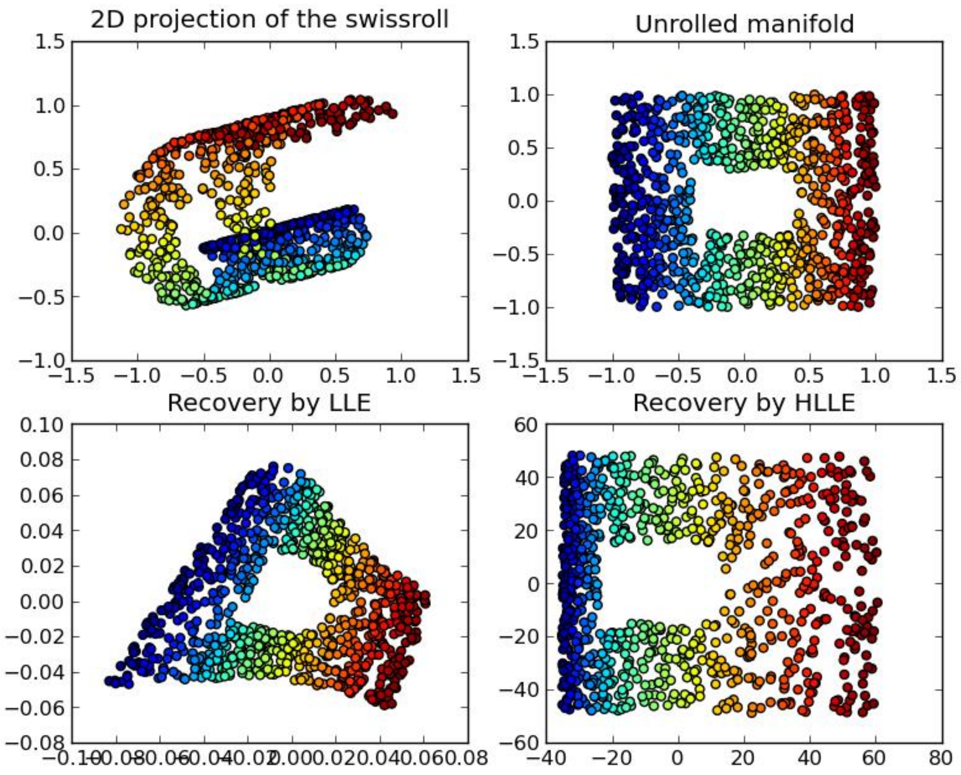
The **undercomplete** <u>and</u> **linear** autoencoder is closely related to PCA

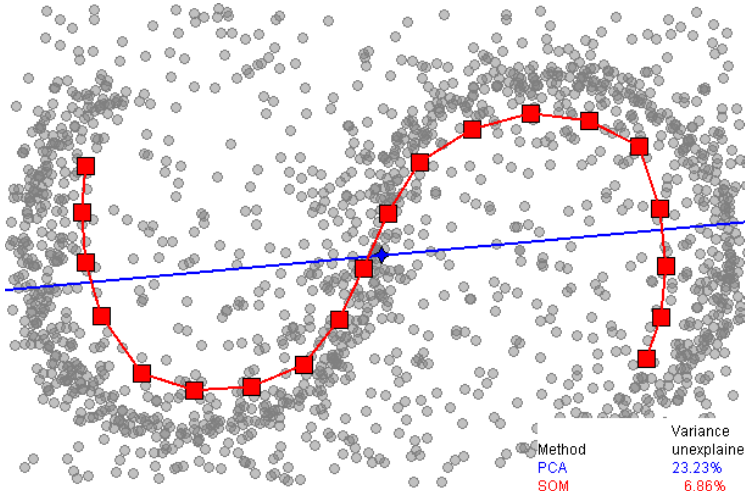
- Formally
  - Let $\mathbf{z} = f(\mathbf{x}) = A\mathbf{x} + b, \mathbf{z} \in \mathbb{R}^k$
  - Let $\tilde{\mathbf{x}} = g(\mathbf{z}) = B\mathbf{z} + c$
  - Let $L(\mathbf{x}, \tilde{\mathbf{x}}) = \mathbb{E}[||\mathbf{x} - \tilde{\mathbf{x}}||_2^2]$

- One solution can be derived from PCA though other (closely-related) solutions exist

- Autoencoders are “non-linear” PCA

Why might we want a **non-linear** autoencoder?
Non-linear dimensionality reduction





https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction

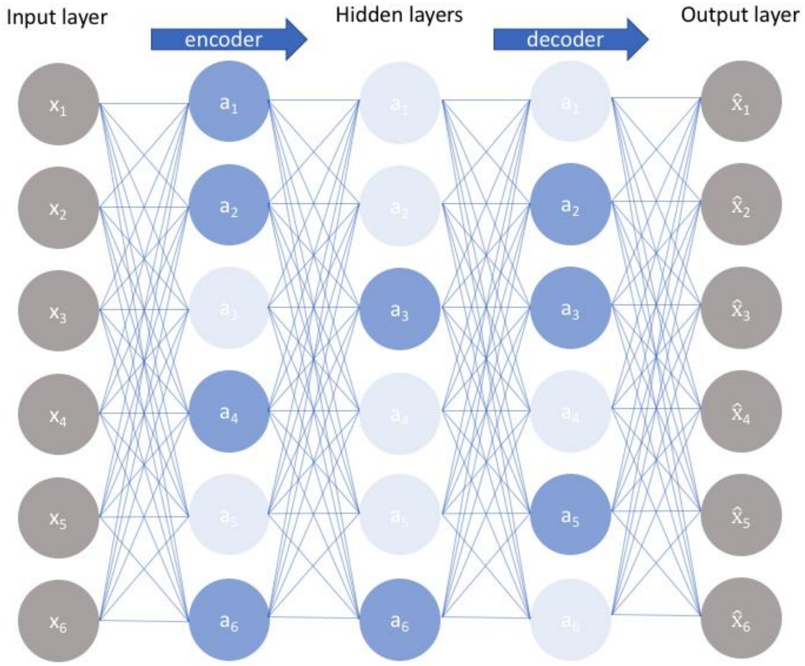
Sparse Autoencoder

Sparse autoencoders add a penalty that the latent space is sparse

- Add a regularization term to latent variables

$$\min_{f,g} \mathbb{E}\left[L\left(\mathbf{x}, g\left(f(\mathbf{x})\right)\right) + \lambda \|f(\mathbf{x})\|_1\right]$$

- This creates **data-dependent** sparsity

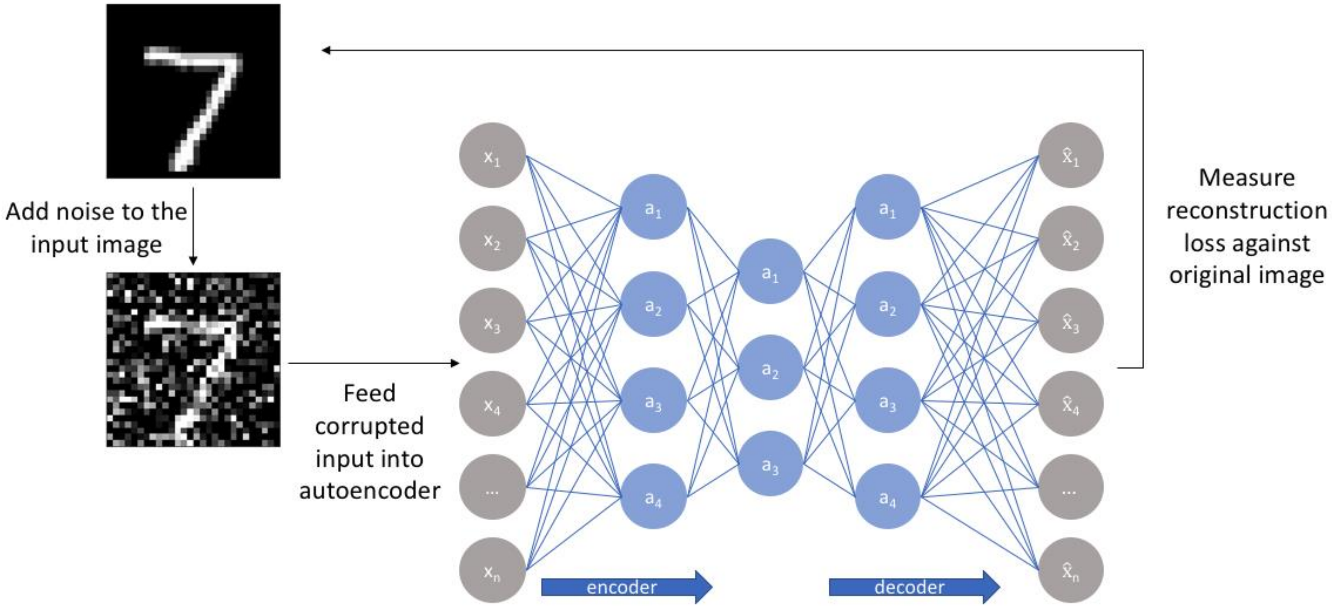


Denoising Autoencoder

<u>Denoising autoencoders</u> force functions to learn to remove noise rather than copy the input

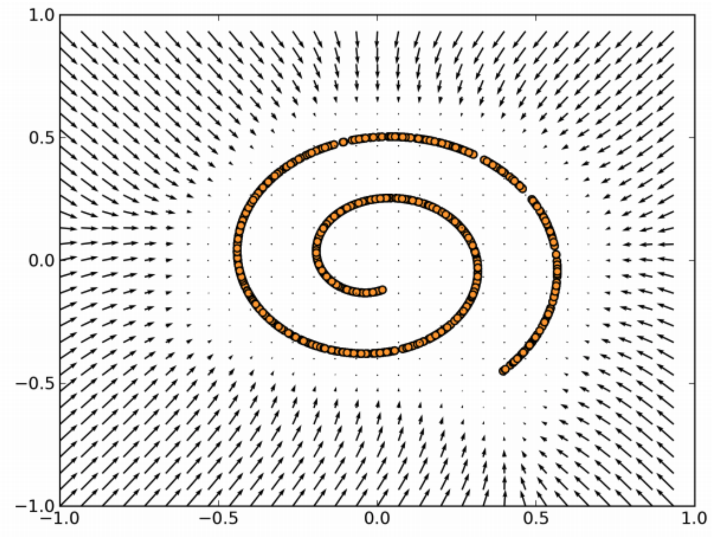
- Add noise to the input so that copying input is not possible

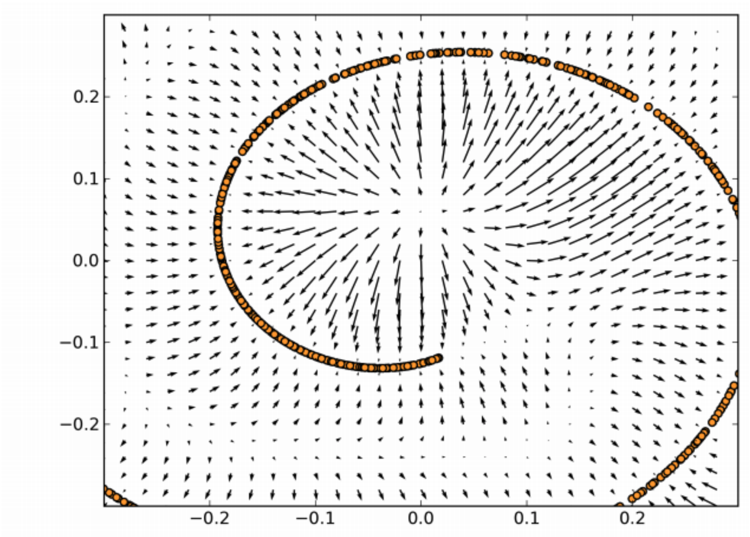
$$\min_{f,g} \mathbb{E}_{\mathbf{x},\epsilon}\left[L\left(\mathbf{x}, g\big(f(\mathbf{x}+\epsilon)\big)\right)\right], \text{where } \epsilon \sim \mathcal{N}(\mu, \sigma I)$$



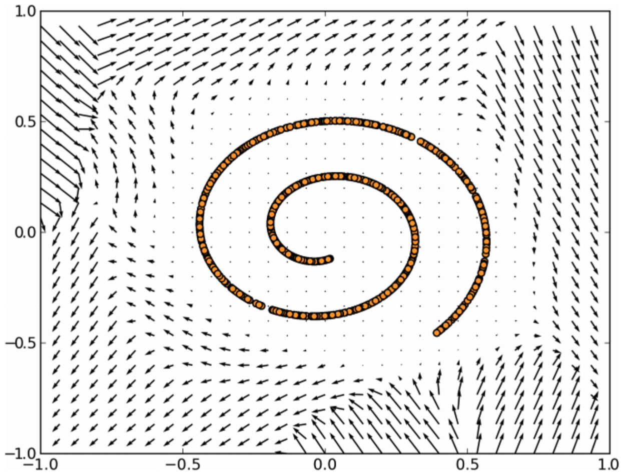
Denoising Autoencoder

Denoising autoencoders can be shown to learn the structure of the distribution





Vector field that pushes corrupted data back to manifold as learned by DAE



Zoomed in view

May misbehave outside data area in practice

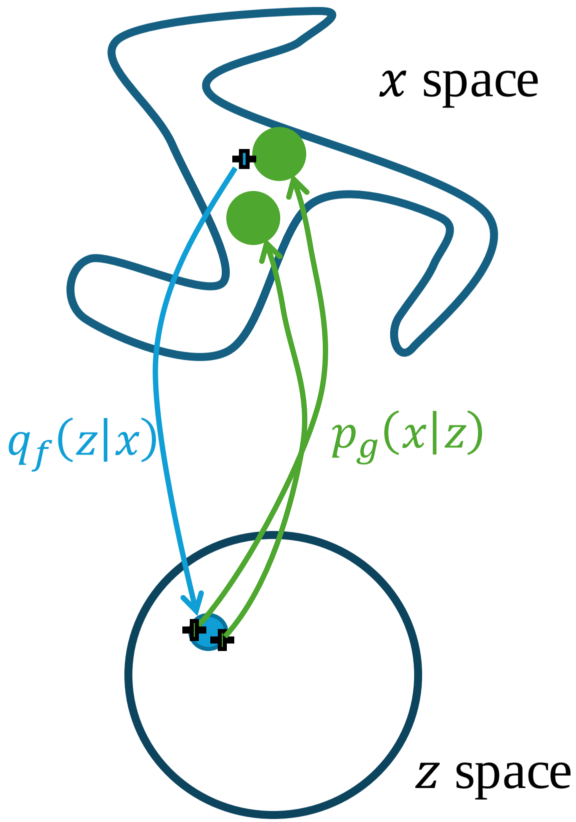
https://arxiv.org/pdf/1211.4246.pdf

Probabilistic Autoencoder

Autoencoders can also use **non-deterministic** or **probabilistic mappings**

- The outputs are **distributions** instead of a points
  - Encoder/decoder output the **parameters** of distribution

- Probabilistic mappings
  - Replace encoder $f(x)$ with $q_f(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}\,; \boldsymbol{\mu} = f(\mathbf{x}), \Sigma = I)$
  - Replace decoder $g(z)$, with $p_g(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}\,; \boldsymbol{\mu} = g(\mathbf{z}), \Sigma = I)$

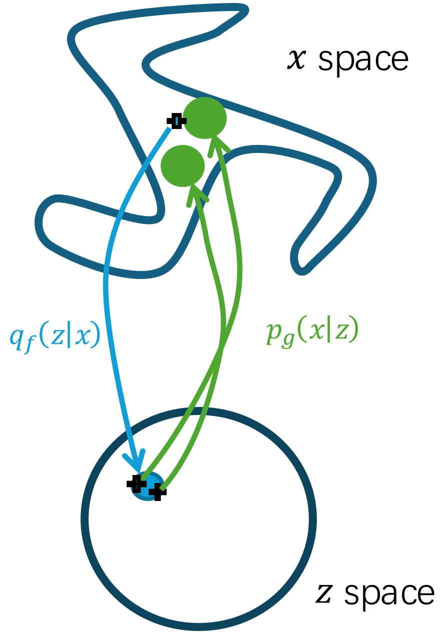


Probabilistic Autoencoder

Vanilla probabilistic autoencoder could minimize expected negative log likelihood of training data

$$\min_{f,g} \mathbb{E}_{p_{\text{data}}(x)}\left[\mathbb{E}_{q_f(z|x)}[-\log p_g(x|z)]\right]$$

- Fact: If $\epsilon \sim \mathcal{N}(0, I)$ and $z_\ell = \mu + \epsilon$, then $z_\ell \sim \mathcal{N}(\mu, I)$.
- $\widehat{\mathbb{E}}_{p_{\text{data}}(x)}\left[\widehat{\mathbb{E}}_{q_f(z|x)}[-\log p_g(x|z)]\right]$ (Empirical expectation)
- $= \frac{1}{n}\sum_i \frac{1}{m}\sum_\ell -\log p_g\left(x_i | z_i^\ell\right)$ $\quad p_g(x_i|z_i^\ell) = \mathcal{N}(x\,; \mu_i^\ell = g(z_i^\ell), \Sigma = I)$
- $= \frac{1}{n}\sum_i \frac{1}{m}\sum_\ell -\log\exp\left(-\frac{1}{2}\|x_i - \mu_i^\ell\|_2^2 - \frac{d}{2}\log 2\pi\right)$
- $= \frac{1}{n}\sum_i \frac{1}{m}\sum_\ell \frac{1}{2}\|x_i - \mu_i^\ell\|_2^2 + c$ (c is constant)
- $= \frac{1}{n}\sum_i \frac{1}{m}\sum_\ell \frac{1}{2}\|x_i - g(z_i^\ell)\|_2^2 + c$ $\quad q_f(z_i^\ell|x_i) = \mathcal{N}(z\,; \mu_i = f(x_i), \Sigma = I)$
- $= \frac{1}{n}\sum_i \frac{1}{m}\sum_\ell \frac{1}{2}\|x_i - g(f(x_i) + \epsilon_i^\ell)\|_2^2 + c$ (Remember fact above)
- $= \frac{1}{n}\sum_i \frac{1}{2}\|x_i - g(f(x_i) + \epsilon_i)\|_2^2 + c$ (let $m = 1$, where $\epsilon_i \sim \mathcal{N}(0, I)$)



Notice the reconstruction term is like AE except for added noise if Gaussian is used.

Comparison between autoencoders

- MSE autoencoder (AE)

$$\min_{f,g} \frac{1}{n} \sum_i \left\| \mathrm{x}_i - g\big(f(\mathbf{x}_i)\big) \right\|_2^2$$

- Sparse autoencoder

$$\min_{f,g} \frac{1}{n} \sum_i \left\| \mathbf{x}_i - g\big(f(\mathbf{x}_i)\big) \right\|_2^2 + \lambda \| f(\mathbf{x}_i) \|_1$$

- Gaussian denoising autoencoder (DAE)

$$\min_{f,g} \frac{1}{n} \sum_i \left\| \mathbf{x}_i - g\big(f(\mathbf{x}_i + \epsilon_i)\big) \right\|_2^2, \qquad \epsilon_i \sim \mathcal{N}(\mu, \sigma \mathrm{I})$$

- Vanilla Gaussian probabilistic autoencoder (with 1 sample in latent space)

$$\min_{f,g} \frac{1}{n} \sum_i \| \mathbf{x}_i - g(f(\mathbf{x}_i) + \epsilon_i) \|_2^2, \qquad \epsilon_i \sim \mathcal{N}(0, I)$$

- Regularized Gaussian probabilistic autoencoder

$$\min_{f,g} \frac{1}{n} \sum_i \| \mathbf{x}_i - g(f(\mathbf{x}_i) + \epsilon_i) \|_2^2 + \lambda \| f(\mathbf{x}_i) \|_2^2, \qquad \epsilon_i \sim \mathcal{N}(0, I)$$

Variational Autoencoder (VAE)

Variational Autoencoders (VAE) are one of the most common probabilistic autoencoders

- Method produces both
  - Probabilistic encoder/decoder for dimensionality reduction/compression
  - **Generative model** for the data
    (AEs don’t provide this)

- **Generative model** can produce fake data

Variational Autoencoder (VAE)

VAEs have inference and generative networks with an assumed prior distribution on $\boldsymbol{z}$

- Generative model

$$\mathbf{z} \sim p_g(\mathbf{z}) = \mathcal{N}(0, I)$$
$$\mathbf{x} \sim p_g(\mathbf{x}|\mathbf{z})$$

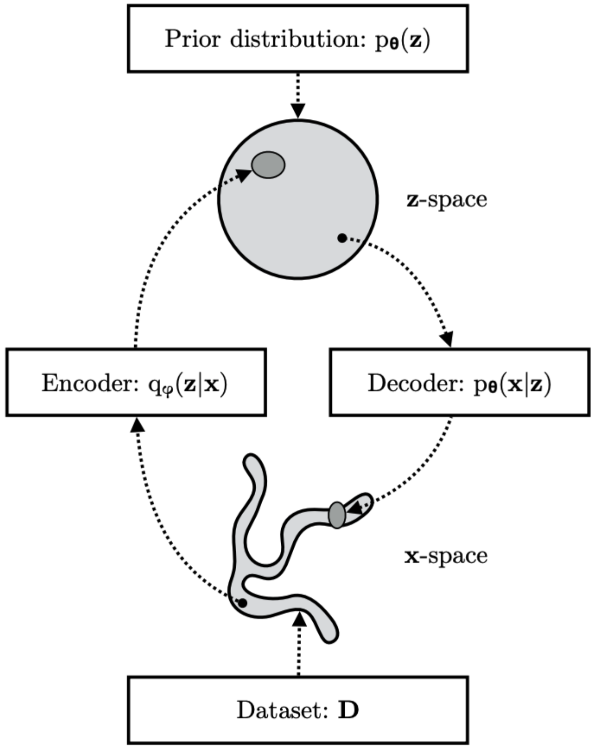
- MLE is intractable

$$\log p(\mathbf{x}; g) = \log \int_{\mathbf{z}} p_g(\mathbf{z}) p_g(\mathbf{x}|\mathbf{z}) dz$$

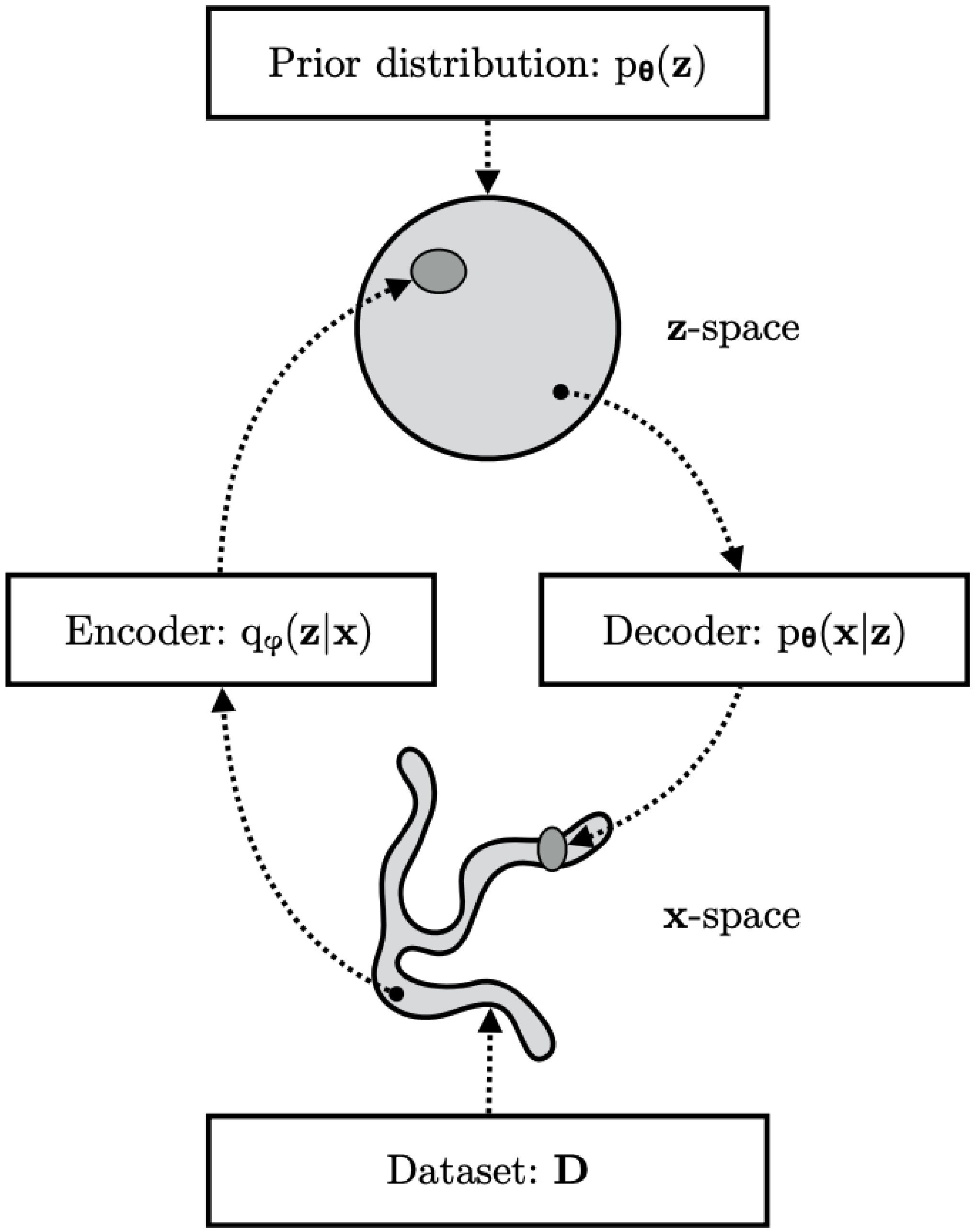
Observed likelihood intractable

- Add encoder/inference model to help

$$\mathbf{x} \sim p_{\text{data}}(\mathbf{x})$$
$$\mathbf{z} \sim q_f(\mathbf{z}|\mathbf{x})$$

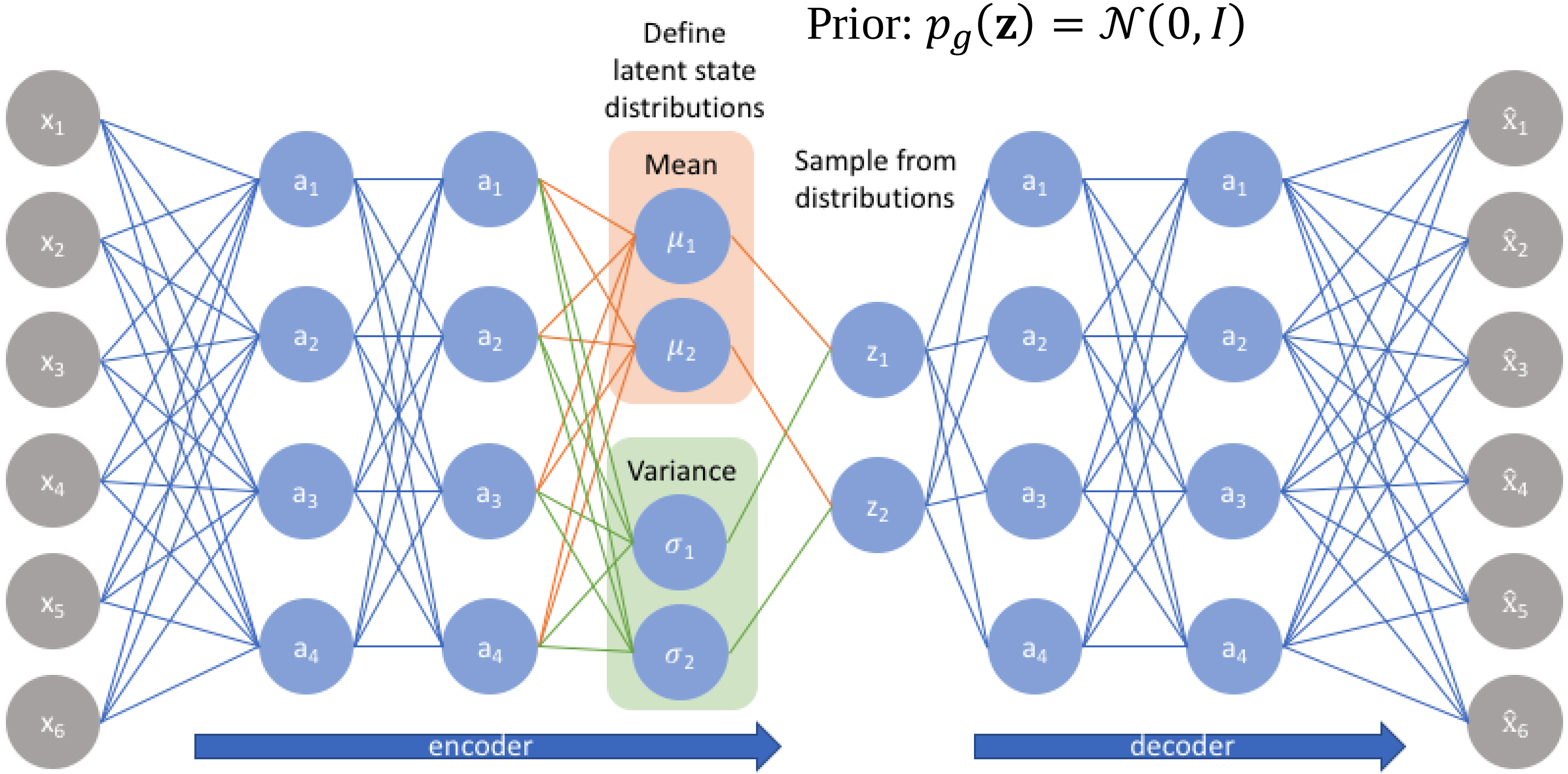


Kingma, D. P., & Welling, M. (2019). An Introduction to Variational Autoencoders. *Foundations and Trends® in Machine Learning*, *12*(4), 307-392.


Prior distribution: $p_\theta(\mathbf{z})$
$\mathbf{z}$-space
Encoder: $q_\varphi(\mathbf{z}|\mathbf{x})$
Decoder: $p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}$-space
Dataset: $\mathbf{D}$

VAE implementation

Prior: $p_g(\mathbf{z}) = \mathcal{N}(0, I)$



$q_f(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}; diag(\sigma_1, \sigma_2))$

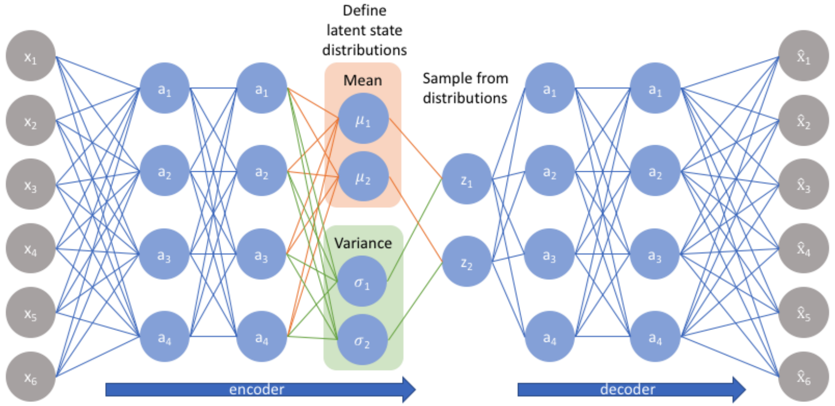
$p_g(\mathbf{x}|\mathbf{z})$

Variational Autoencoder (VAE)

VAEs objective function* (for a given $\mathbf{x}$)

Maximize: $\underbrace{\mathbb{E}_{\mathbf{z}\sim q_f}[\log p_g(\mathbf{x}|\mathbf{z})] - KL\left(q_f(\mathbf{z}|\mathbf{x}), p_g(\mathbf{z})\right)}_{\text{Evidence lower bound (ELBO)}}$

Prior: $p_g(\mathbf{z}) = \mathcal{N}(0, I)$



$q_f(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}; diag(\sigma_1, \sigma_2))$

$p_g(\mathbf{x}|\mathbf{z})$

$$\text{ELBO}(\mathbf{x};\, p_g, q_f) = \log p_g(\mathbf{x}) - KL\left(q_f(\mathbf{z}|\mathbf{x}), p_g(\mathbf{z}|\mathbf{x})\right)$$
$$\leq \log p_g(\mathbf{x})$$

Lower bound! Tight if $q_f(\mathbf{z}|\mathbf{x}) = p_g(\mathbf{z}|\mathbf{x})$

*: For details on the derivation of the objective function, see for example: https://arxiv.org/pdf/1606.05908

Derivation of VAE objective -- ELBO

- $\log p_g(x)$
- $= \mathbb{E}_{q_f}[\log p_g(x)]$ ($\mathbb{E}$ of constant = constant)
- $= \mathbb{E}_{q_f}\left[\log \frac{p_g(x)p_g(z|x)}{p_g(z|x)}\right]$ (inflate)
- $= \mathbb{E}_{q_f}\left[\log \frac{p_g(x,z)}{q_f(z|x)} \frac{q_f(z|x)}{p_g(z|x)}\right]$ (inflate)
- $= \mathbb{E}_{q_f}\left[\log \frac{p_g(x,z)}{q_f(z|x)}\right] + \mathbb{E}_{q_f}\left[\log \frac{q_f(z|x)}{p_g(z|x)}\right]$
- $= \mathrm{ELBO}\left(x;\ p_g, q_f\right) + KL\left(q_f(z|x), p_g(z|x)\right)$
- $\Rightarrow \mathrm{ELBO}\left(x;\ p_g, q_f\right) = \log p_g(x) - KL\left(q_f(z|x), p_g(z|x)\right)$
  - $\leq \log p_g(x)$

Lower bound! Tight if $q_f(z|x) = p_g(z|x)$

Variational Autoencoder (VAE)

VAEs objective function* (for a given *x*)

$$\text{Maximize: } \underbrace{\mathbb{E}_{\mathbf{z}\sim q_f}\left[\log p_g(\mathbf{x}|\mathbf{z})\right] - KL\left(q_f(\mathbf{z}|\mathbf{x}), p_g(\mathbf{z})\right)}_{\text{Evidence lower bound (ELBO)}}$$

Minimizing the negative yields error + regularization

$$\min_{f,g} -\frac{1}{n}\sum_i \mathbb{E}_{q_f}\left[\log p_g(\mathbf{x}_i|\mathbf{z}_i)\right] + KL\left(q_f(\mathbf{z}_i|\mathbf{x}_i), p_g(\mathbf{z}_i)\right)$$

Computable

Computable in closed-form for
Gaussian distributions

*: For details on the derivation of the objective function, see for example: https://arxiv.org/pdf/1606.05908

Variational Autoencoder (VAE)

$$\mathcal{L} = -\frac{1}{n}\sum_i \mathbb{E}_{q_f}[\log p_g(\mathbf{x}_i|\mathbf{z}_i)] + KL\left(q_f(\mathbf{z}_i|\mathbf{x}_i), p_g(\mathbf{z}_i)\right)$$

Sampling $\mathbf{z}_i^{(l)} \sim q_f$

KL divergence between two Gaussian distributions

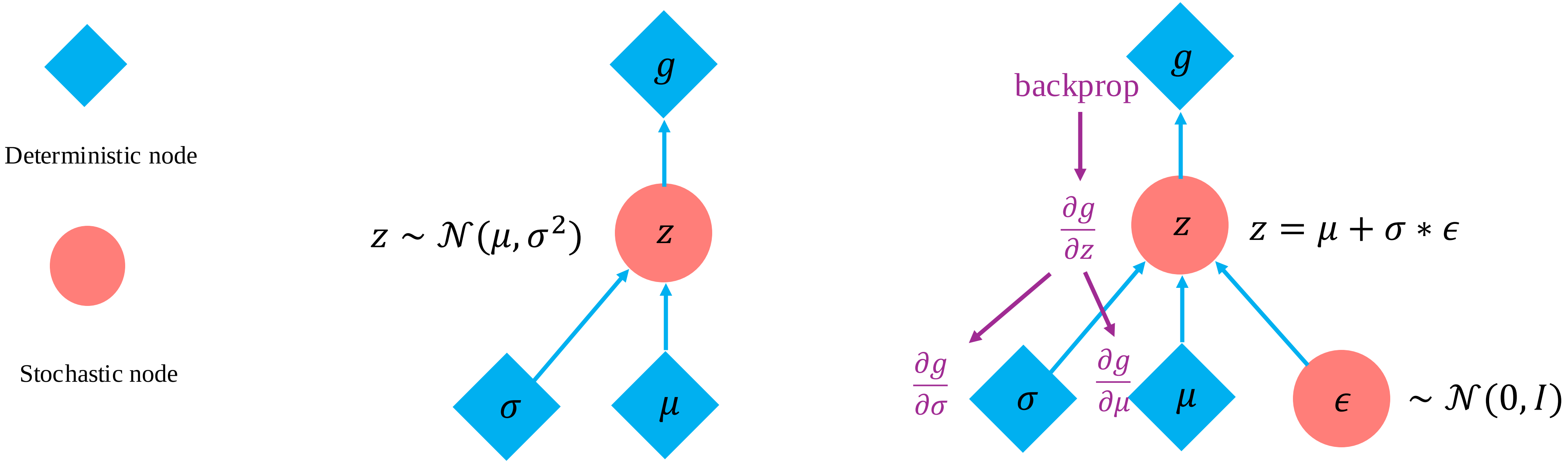
$$\mathcal{L} = -\frac{1}{n}\sum_i \left\{\frac{1}{L}\sum_{l=1}^{L} \log p_g(\mathbf{x}_i|\mathbf{z}_i^{(l)}, \mathbf{w}_g) + \frac{1}{2}\sum_{k=1}^{M}(1 + \log\sigma_{k,i}^2 - \mu_{k,i}^2 - \sigma_{k,i}^2)\right\}$$

Reparameterization trick:
$\mathbf{z}_i^{(l)} = f_\mu(\mathbf{x}_i, \mathbf{w}_f) + f_\sigma(\mathbf{x}_i, \mathbf{w}_f)\epsilon^{(l)}$, where $\epsilon^{(l)} \sim \mathcal{N}(0, I)$

$$\mathcal{L} = -\frac{1}{n}\sum_i \left\{\frac{1}{L}\sum_{l=1}^{L} \log p_g(\mathbf{x}_i|f_\mu(\mathbf{x}_i) + f_\sigma(\mathbf{x}_i)\epsilon^{(l)}, \mathbf{w}_g) + \frac{1}{2}\sum_{k=1}^{M}(1 + \log\sigma_{k,i}^2 - \mu_{k,i}^2 - \sigma_{k,i}^2)\right\}$$

For gaussian p_g, $\frac{1}{L}\sum_{l=1}^{L} \left\|\mathbf{x}_i - g_\mu\left(f^{(l)}(\mathbf{x}_i)\right)\right\|^2 - g_\sigma^2\left(f^{(l)}(\mathbf{x}_i)\right)$

Reparameterization trick



Original form

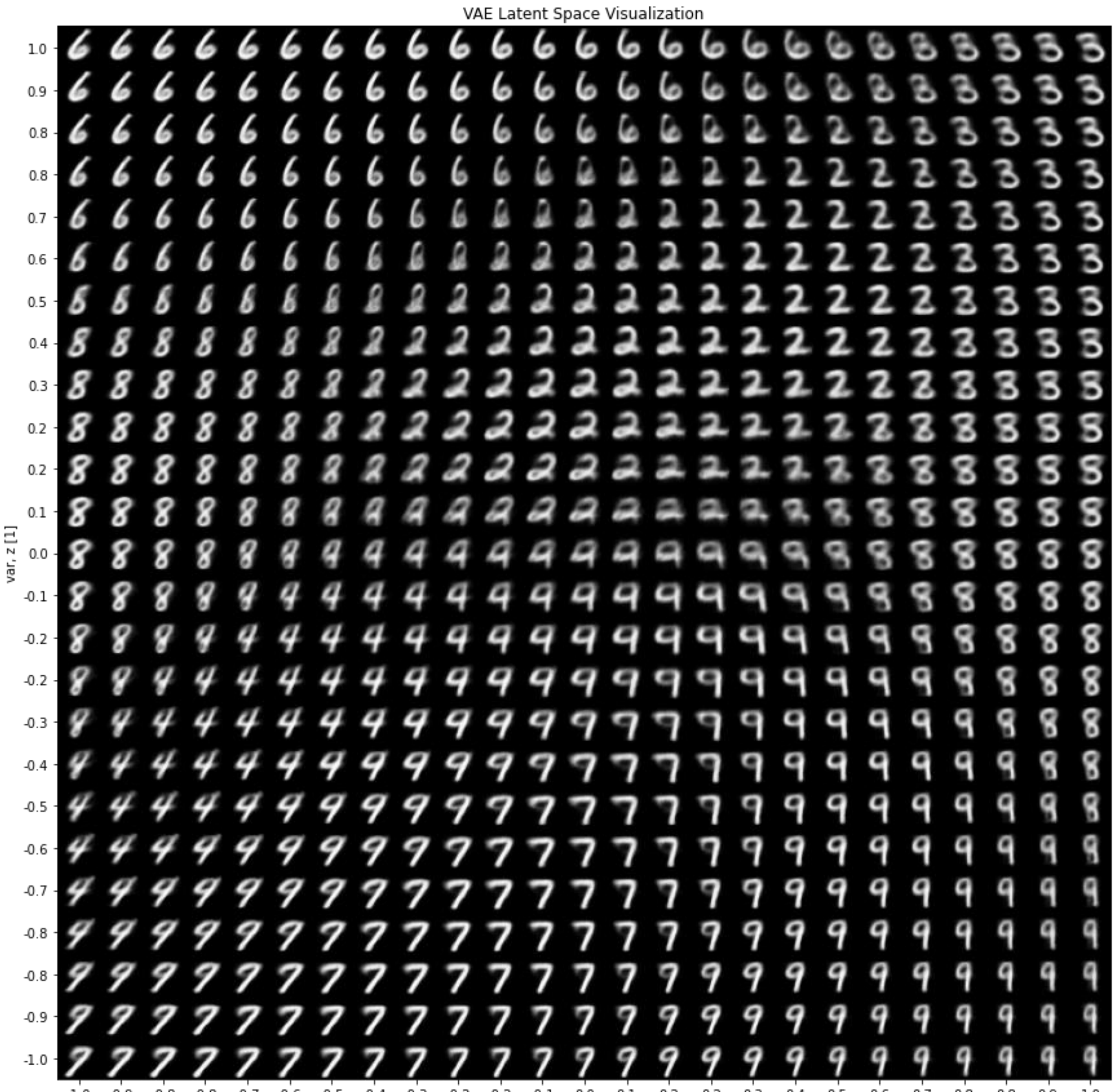
Reparameterization form

$$\frac{\partial g}{\partial \mu} = \frac{\partial g}{\partial z} \cdot I, \quad \frac{\partial g}{\partial \sigma} = \frac{\partial g}{\partial z} * \epsilon$$

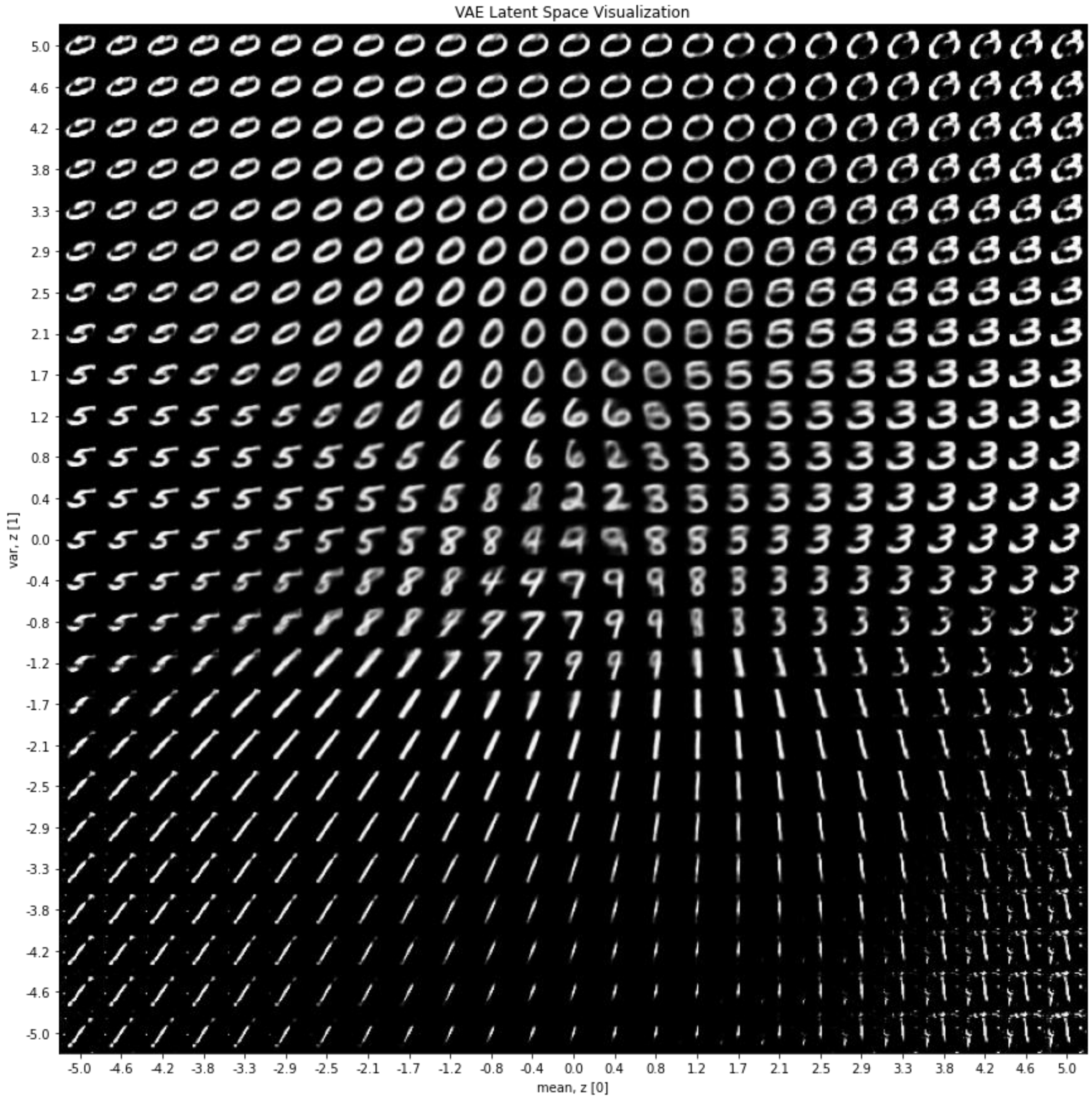
Visualization of latent space:

$$[-1.0, 1.0]^2$$

$$z_1 = \{-1.0, -0.9, -0.8, \cdots, 0.9, 1.0\}$$
$$z_2 = \{-1.0, -0.9, -0.8, \cdots, 0.9, 1.0\}$$



Visualization of latent space: $[-5.0, 5.0]^2$



Putting it all together: The VAE algorithm using (mini-batch) SGD

1. Get minibatch of data x
2. Pass through encoder to get $\mu, \sigma^2 = f(x)$
3. Sample from $z = q_f(z|x, (\mu, \sigma^2) = f(x))$ using reparameterization trick
4. Pass through decoder to get output parameters $\theta = g(z)$
5. Compute log likelihood of $p_g(x|z, \theta = g(z))$
6. Loss is negative log likelihood + KL term
7. Backpropagate to gradients for both g and f and update model

Variational Autoencoder (VAE)

Inference (deploy and generate new samples):

- Encoder network f is discarded
- New data points are generated by
  - sampling from the prior $p_g(\mathbf{z}) = \mathcal{N}(0, I)$
  - forward propagating through the decoder network g

Testing: a new test point $\hat{\mathbf{x}}$

- Use the encoder network f to estimate the latent variable distribution $q_f(\mathbf{z}|\hat{\mathbf{x}}, \mathbf{w}_f)$
- Sample $\mathbf{z}$ from $q_f(\mathbf{z}|\hat{\mathbf{x}}, \mathbf{w}_f)$, and feed to decoder network g
- Using the EBLO as an approximation

Variational Autoencoder (VAE)

$$\mathbb{E}_{\mathbf{z}\sim q_f}[\log p_g(\mathbf{x}|\mathbf{z})] - KL\left(q_f(\mathbf{z}|\mathbf{x}), p_g(\mathbf{z})\right)$$

Encourages the encoder distribution $q_f(\mathbf{z}|\mathbf{x})$
to be close to the prior $p_g(\mathbf{z})$

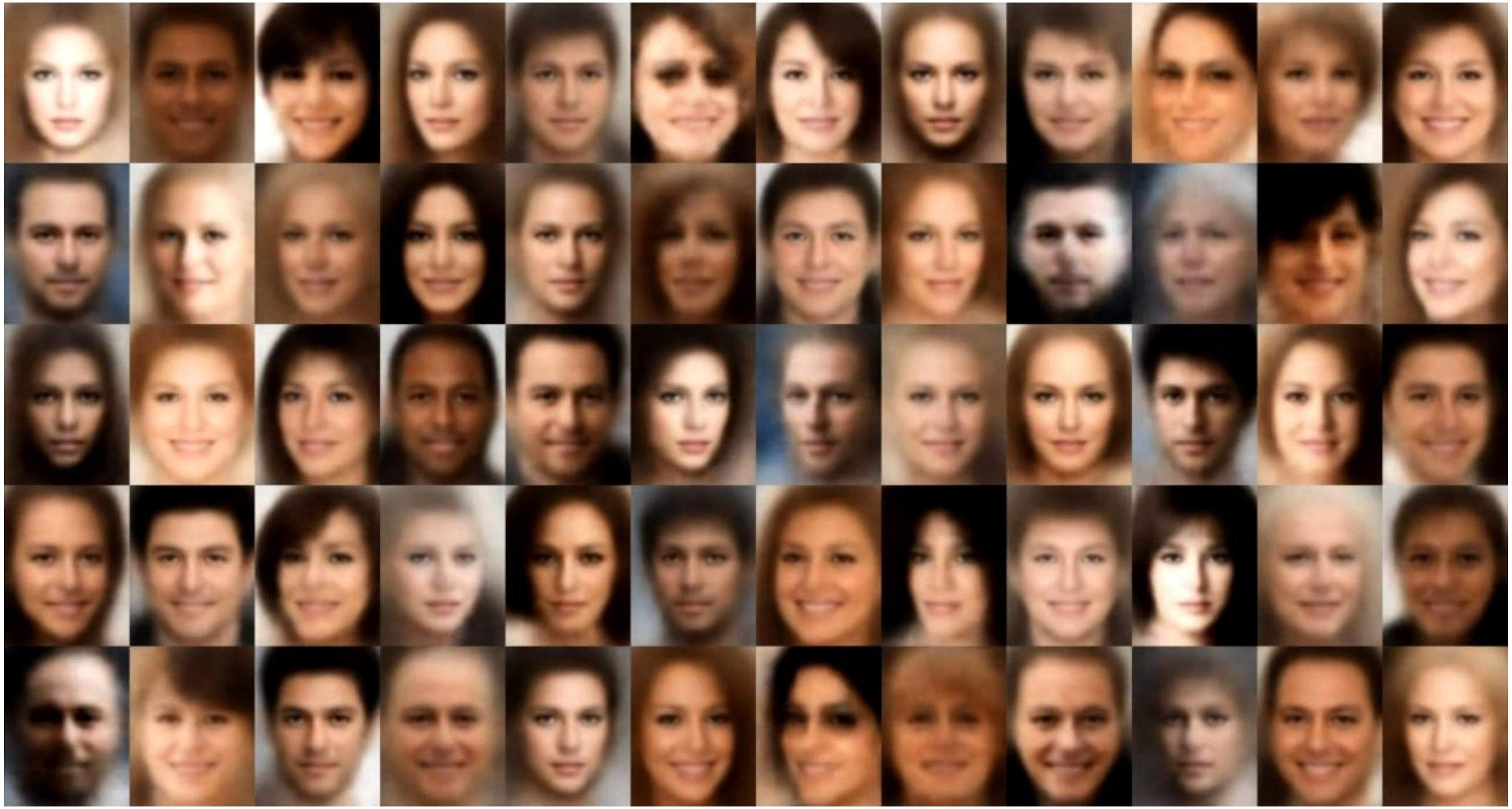
😀 Encoder can produce realistic outputs when sampling from $p_g(\mathbf{z})$

☹️ If too close, $q_f(\mathbf{z}|\mathbf{x}) \approx p_g(\mathbf{z})$, no longer $\mathbf{x}$ dependent. [posterior collapse]
Poor reconstruction of test images

Introduce a hyper-parameter β to balance

$$\mathbb{E}_{\mathbf{z}\sim q_f}[\log p_g(\mathbf{x}|\mathbf{z})] - \beta \cdot KL\left(q_f(\mathbf{z}|\mathbf{x}), p_g(\mathbf{z})\right)$$

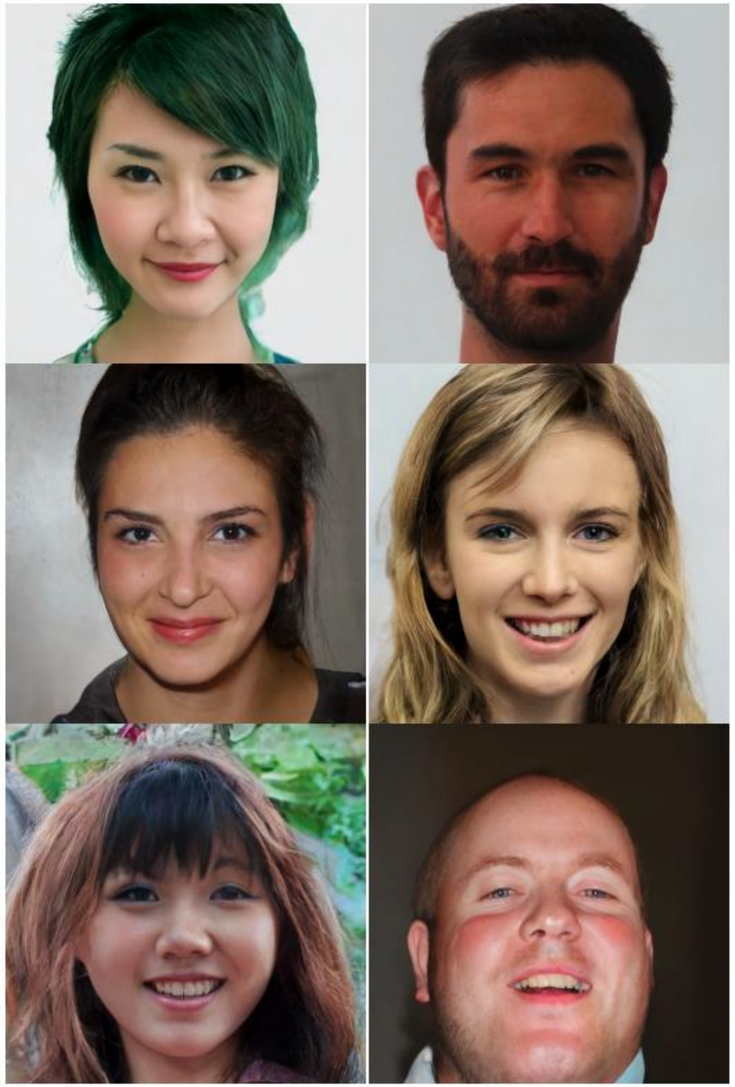
“Traditional” Drawback:
VAEs tend to generate blurry images rather than sharp images



https://github.com/WojciechMormul/vae

Maybe not a drawback…
VQ-VAE-2 at *NeurIPS 2019*

Generated high-quality images (probably don't ask how long it takes to train this though…)



Razavi, A., van den Oord, A., & Vinyals, O. (2019). Generating diverse high-fidelity images with vq-vae-2. In *Advances in Neural Information Processing Systems* (pp. 14866-14876).