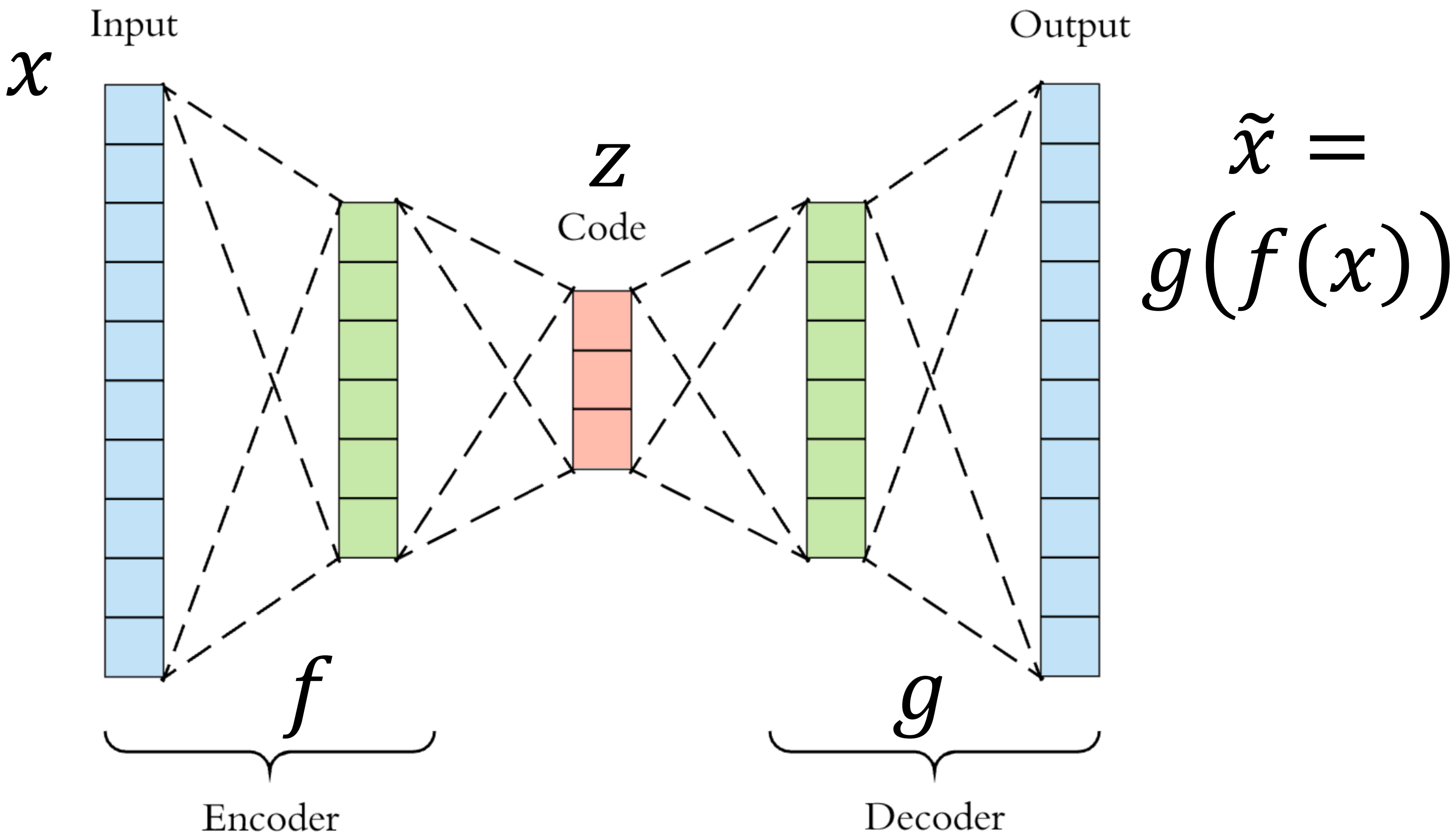
Autoencoders

ECE57000: Artificial Intelligence

David I. Inouye

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



https://towardsdatascience.com/generating-images-with-autoencoders-77fd3a8dd368

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

- More formally, the autoencoder objective is:

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

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

- One example is using Mean Squared Error loss

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

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

- Suppose $f(x) = x$ and $g(x) = x$

- Then we know that

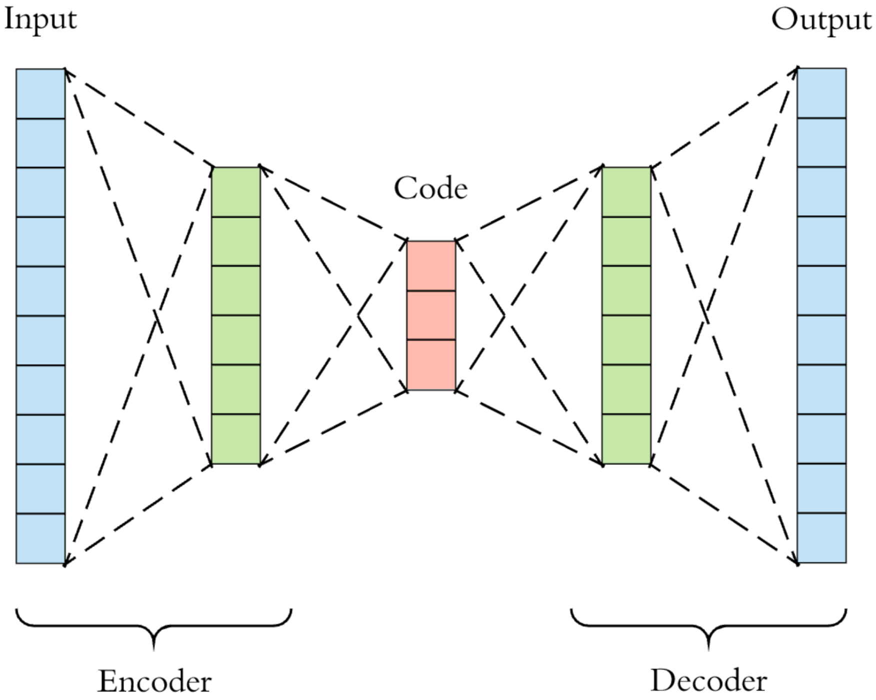
$$\min_{f,g} \mathbb{E}\left[\left\|x - g\big(f(x)\big)\right\|_2^2\right]$$
$$= \min_{f,g} \mathbb{E}[\|x - x\|_2^2] = 0$$

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

- Trivial/useless…What can we do?

Adding constraints to f, g or z can often produce interesting properties of z

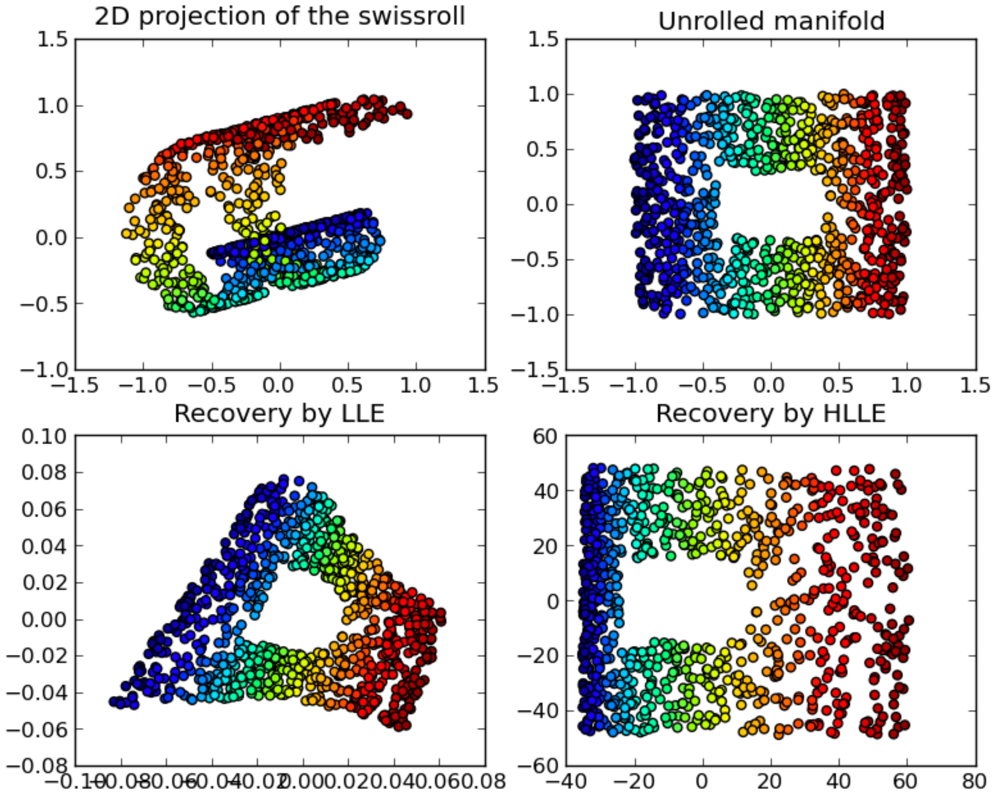
- Undercomplete autoencoders assume that the latent space has lower dimension, i.e., $k < d$

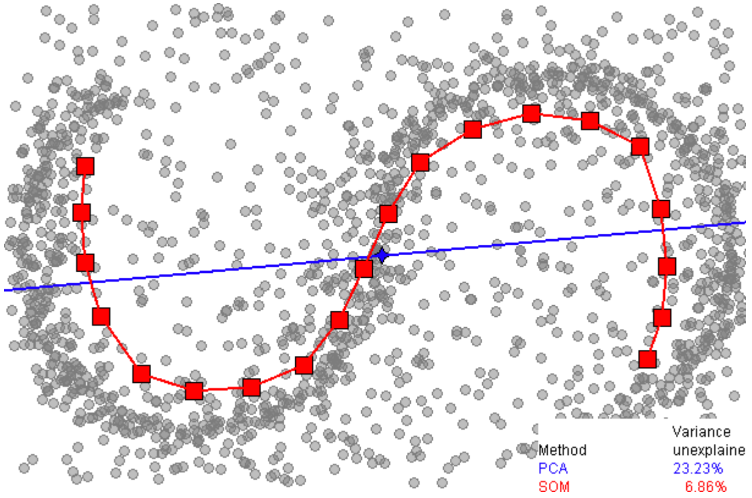


The **undercomplete** and **linear** autoencoder is closely related to PCA

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

Even undercomplete autoencoders can be uninteresting if **non-linear**

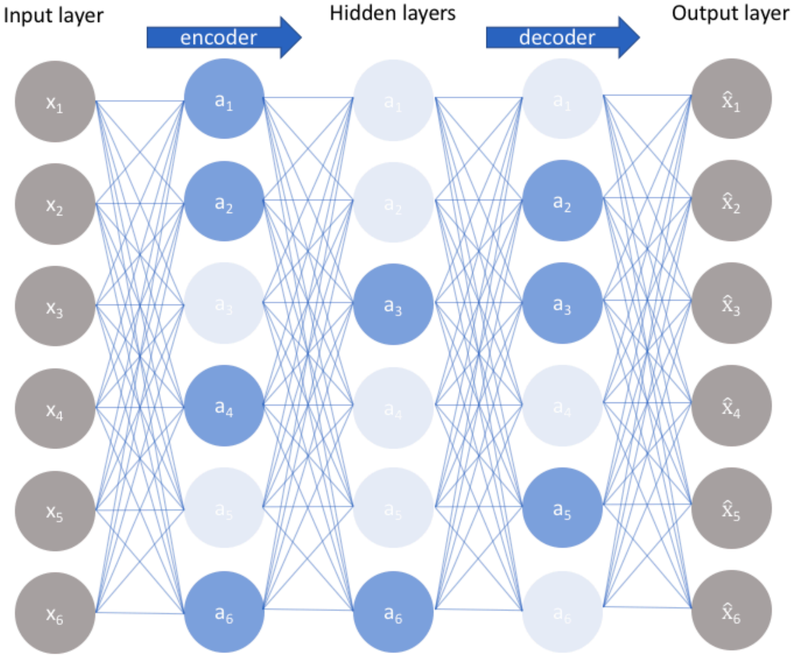
- With no other constraints, again an uninteresting solution exists where x_i is a training instance
    - $z = f(x) = \begin{cases} i, & x = x_i \\ 0, & \text{otherwise} \end{cases}$
    - $g(z) = \begin{cases} x_i, & z = i \\ 0, & \text{otherwise} \end{cases}$
- While this doesn't necessarily happen in practice, it demonstrates that more constraints are usually needed

Sparse autoencoders add a penalty that the latent space is sparse

- Add a regularization term to latent variables

$$\min_{f,g} L\left(x, g(f(x))\right) + \lambda \|f(x)\|_1$$

- This creates **data-dependent** sparsity

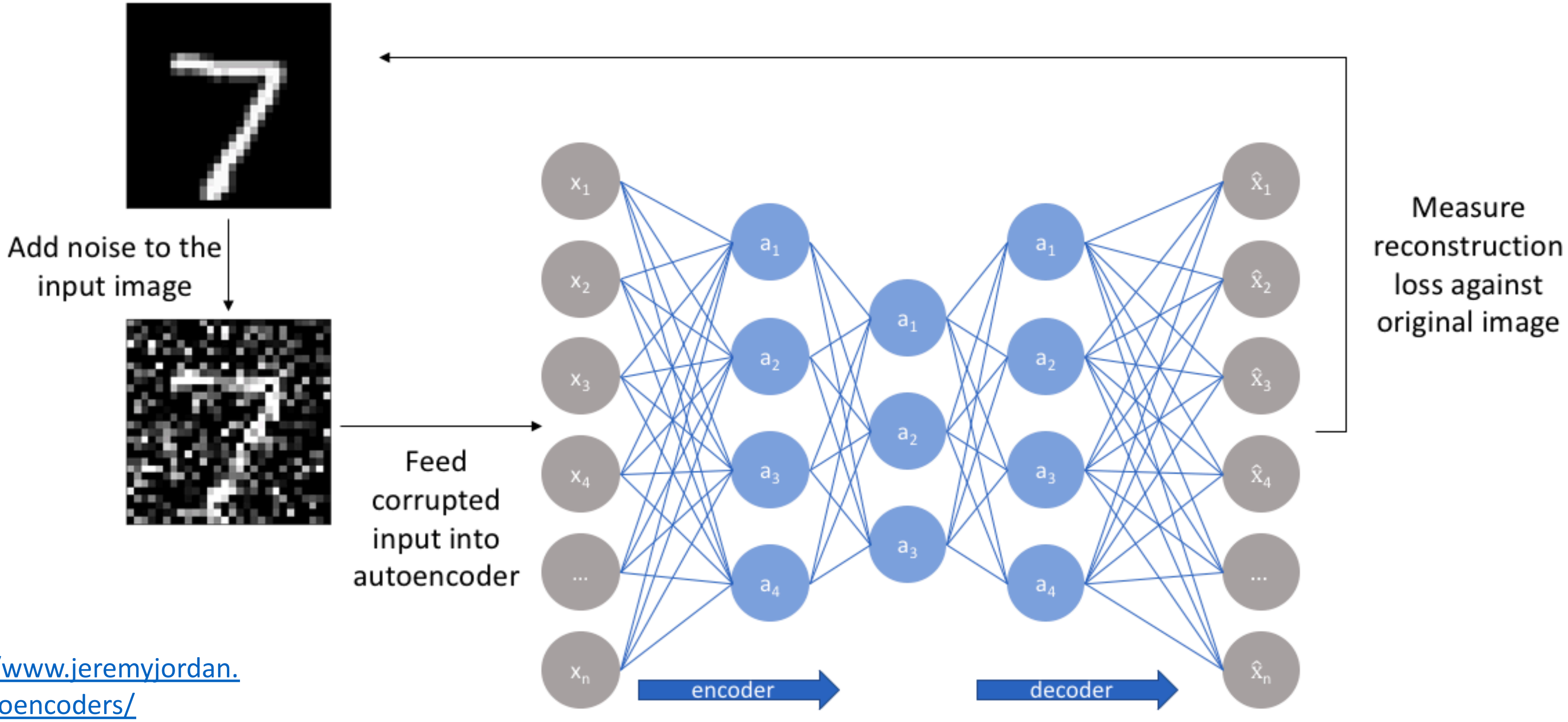


https://www.jeremyjordan.me/autoencoders/

<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

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



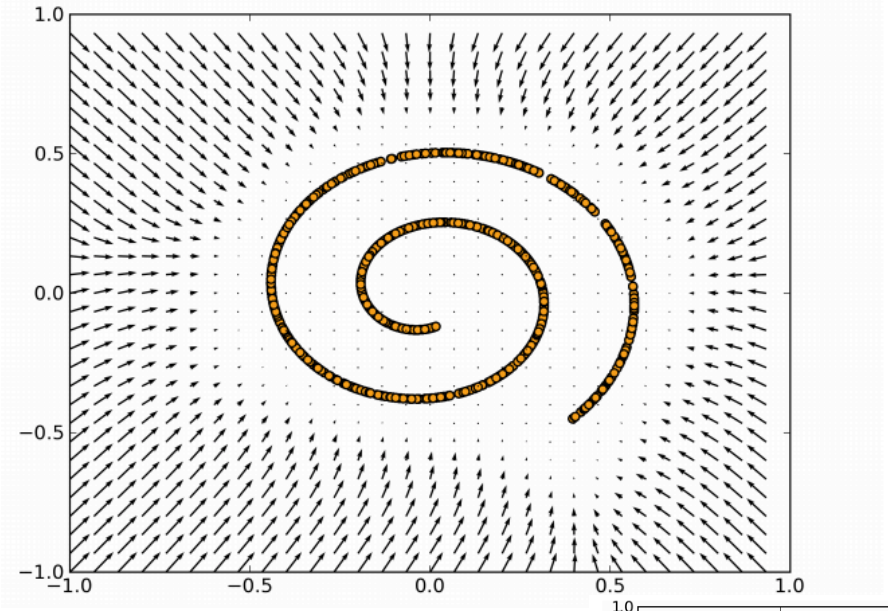
Add noise to the input image

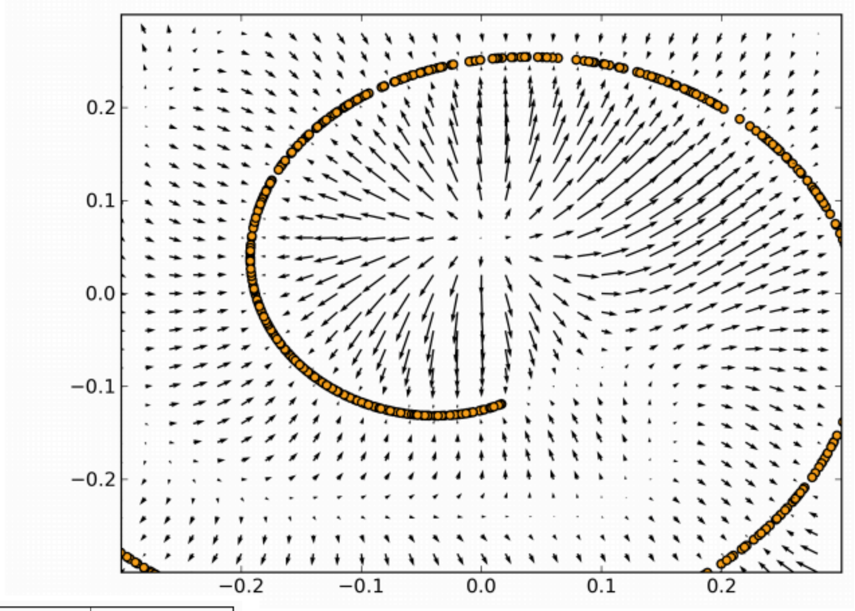
Feed corrupted input into autoencoder

Measure reconstruction loss against original image

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

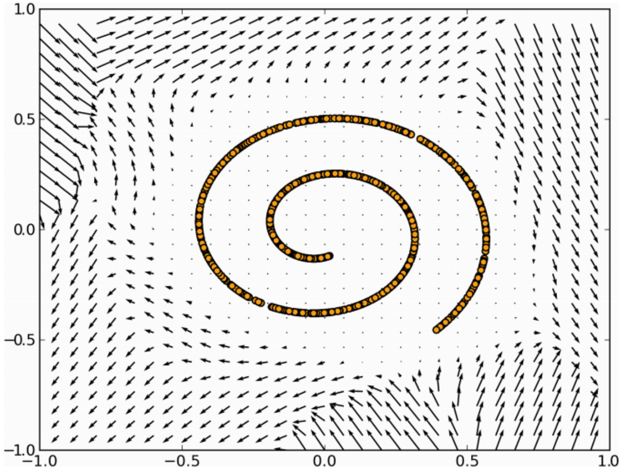
Denoising autoencoders can be shown to learn the structure of the distribution





Zoomed in view

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



May misbehave outside data area in practice

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

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

Vanilla probabilistic autoencoder could minimize expected negative log probability 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 expectations)
- $= \frac{1}{n}\sum_i \frac{1}{m}\sum_\ell -\log p_g\left(x_i \middle| z_i^\ell\right)$
- $= \frac{1}{n}\sum_i \frac{1}{m}\sum_\ell -\log\exp\left(-\frac{1}{2}\left\|x_i - \mu_i^\ell\right\|_2^2 - \frac{d}{2}\log 2\pi\right)$
- $= \frac{1}{n}\sum_i \frac{1}{m}\sum_\ell \frac{1}{2}\left\|x_i - \mu_i^\ell\right\|_2^2 + c$ (c is constant)
- $= \frac{1}{n}\sum_i \frac{1}{m}\sum_\ell \frac{1}{2}\left\|x_i - g\left(z_i^\ell\right)\right\|_2^2 + c$
- $= \frac{1}{n}\sum_i \frac{1}{m}\sum_\ell \frac{1}{2}\left\|x_i - g\left(f(x_i) + \epsilon_i^\ell\right)\right\|_2^2 + c$ (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 $\epsilon_i \sim \mathcal{N}(0, I)$)

Comparison between autoencoders

- MSE autoencoder (AE)

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

- Sparse autoencoder

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

- Gaussian denoising autoencoder (DAE)

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

- Vanilla Gaussian probabilistic autoencoder

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

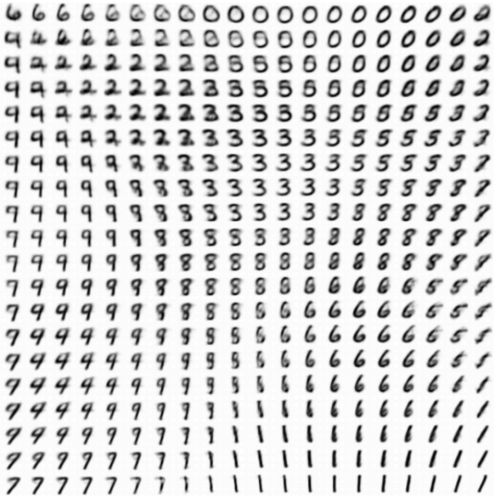
- Regularized Gaussian probabilistic autoencoder

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

This is special case of variational autoencoders (VAE) called β-VAE.

<u>Variational Autoencoders (VAE)</u> are one of the most common probabilistic autoencoders

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



(b) Learned MNIST manifold

- **<u>Generative model</u>** can produce fake data

(a) 2-D latent space (b) 5-D latent space (c) 10-D latent space (d) 20-D latent space

- Derived as a **<u>latent</u> variable** model like GMMs

Figures and reference from Kingma, D. P. and M. Welling (2014). "Auto-Encoding Variational Bayes". *International Conference on Learning Representations*.

VAEs have **inference** and **generative** networks with an **assumed prior** distribution on z

- Generative model

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

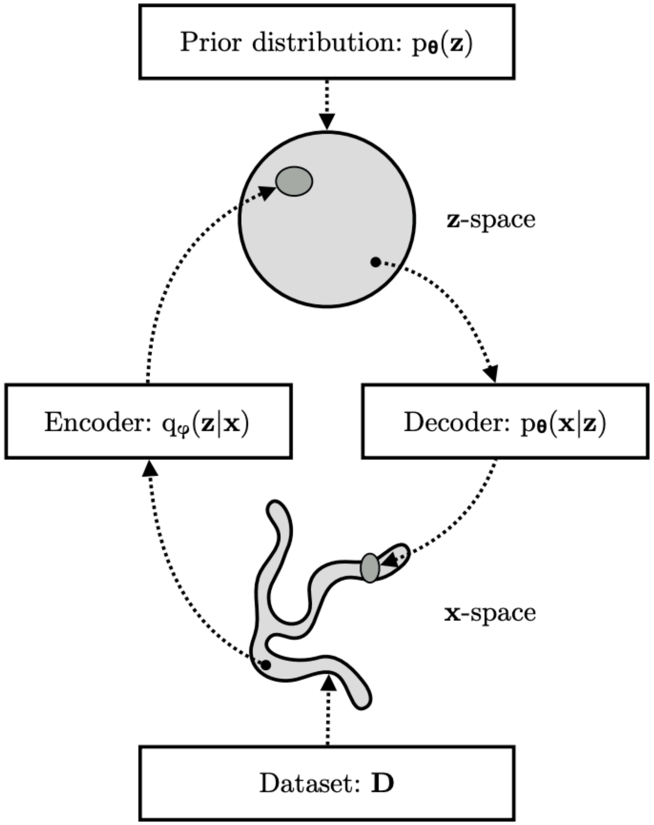
- Encoder/inference model

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

- MLE is intractable

$$\log p(x; g) = \log \int_z p_g(z) p_g(x|z) dz$$

Observed likelihood intractable again like in GMMs



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

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

Derivation of VAE objective (known as the evidence lower bound or ELBO)

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

Tight if $q_f(z|x) = p_g(z|x)$

The ELBO can be interpreted as a reconstruction error term and a regularization term

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

- Minimizing the negative yields error + regularization

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

Computable, see reconstruction error slides

Computable in closed-form for Gaussian distributions

Optimizing the ELBO objective does two things simultaneously

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

- p_g - Optimize the lower bound on the likelihood

- q_f - Improve the lower bound (i.e., make it tighter)

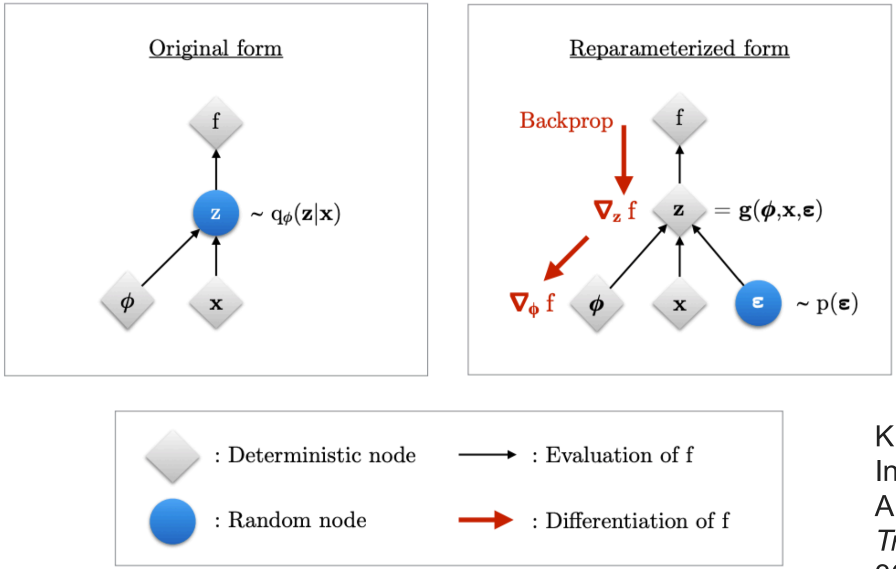
- Like EM algorithm but cannot ensure tightness

Reparameterization trick allows us to compute gradients for q_f

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

Computable in closed-form for Gaussian distributions

$$\min_{f,g} -\frac{1}{n}\sum_i \mathbb{E}_{\epsilon}\left[\log p_g(x_i|z_i = f(x_i) + \epsilon)\right] + \mathbb{E}_{\epsilon}\left[\log \frac{q_f(f(x_i) + \epsilon|x_i)}{p_g(f(x_i) + \epsilon)}\right]$$



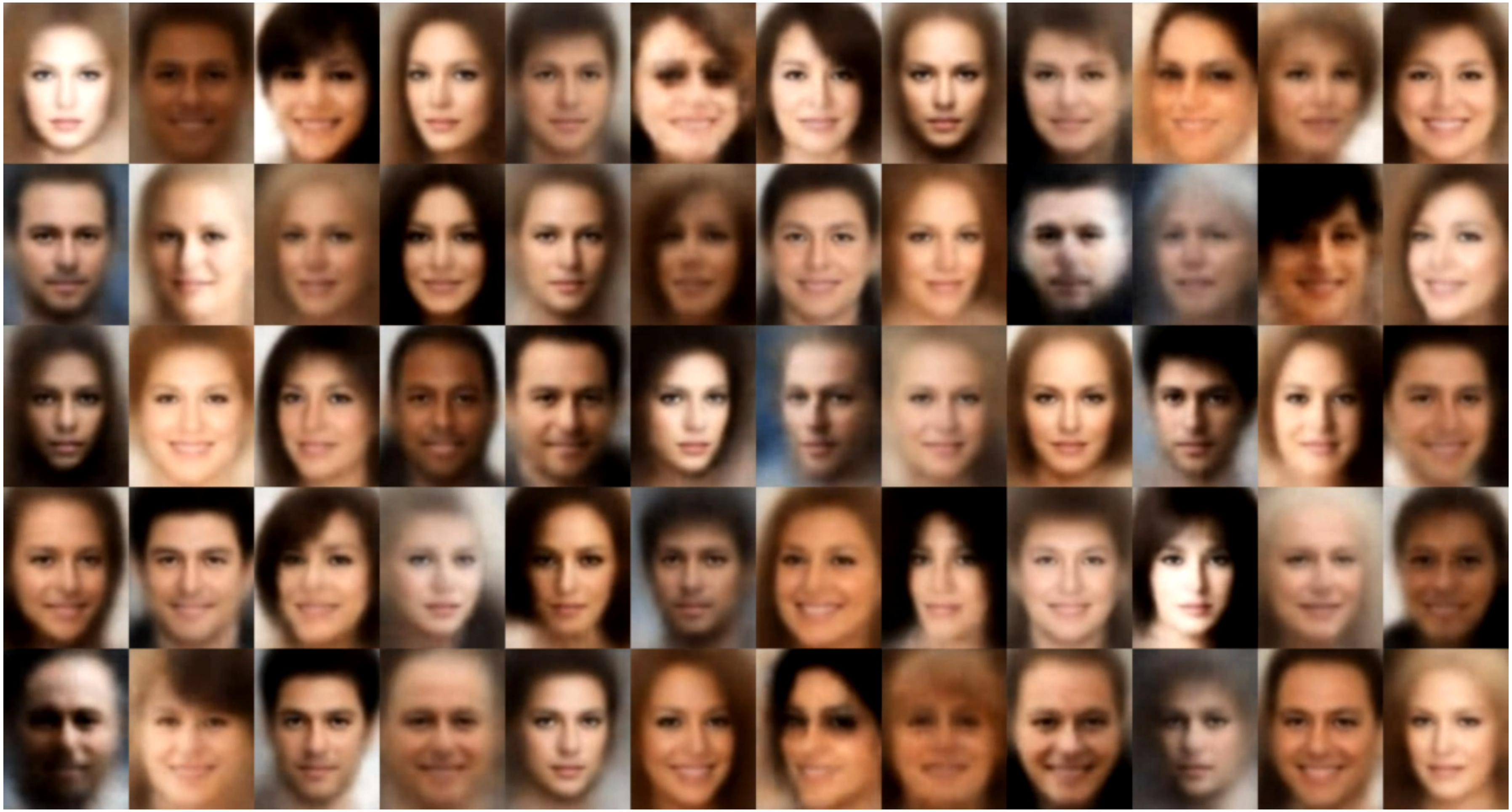
In this figure, f is the loss function, g is the reparameterization function, and ϕ are the parameters of the encoder.

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

Putting it all together:
The VAE algorithm using 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 reparametrization 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

“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).