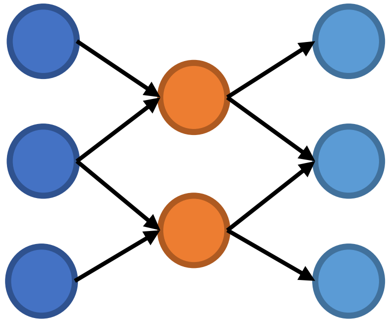
Setup for a “vanilla” autoencoder



- Input: $x_i \in \mathbb{R}^n$
- Basic autoencoder: a feedforward, two layer net such that
  - the first layer “encodes” $\mathrm{x_i} \to z_i \in \mathbb{R}^p, \quad z = f_\theta(x) = \mathrm{sigmoid}(W'x + b')$
  - the second later “decodes” $z_i \to y_i \in \mathbb{R}^n, \quad y = g_\theta(z) = \mathrm{sigmoid}(Wz + b)$

- “Autoencoder” objective:

  find W, b, W', b' such that $\sum_i \|x_i - y_i\|^2$ is minimized

- Note $\mathrm{y} = g_\theta(z) = g_\theta\big(f_\theta(x)\big)$ is trained to be the identity map!

The simplest autoencoder…

- Linear AE: instead of sigmoid functions pick $f_\theta = g_\theta = I$ (identity)
- If we further choose $b = b' = 0$, then
  - Encoding: $z = f_\theta(x) = W'x$
  - Decoding: $y = g_\theta(z) = Wz = WW'x$
- The linear AE cost is: $\mathrm{C} = \sum_i \|x_i - WW'x_i\|^2$
- Can show the optimal W' depends on W:
  - Writing $z = W'x$, consider $F(z) = \|x - Wz\|^2 = xx^T - x^TWz - z^TW^Tx + z^TW^TWz$
  - $0 = \frac{\partial F}{\partial z} \;\rightarrow\; 0 = -2W^Tx + 2W^TWz^* \;\rightarrow\; W^Tx = W^TWz^* \;\rightarrow\; W^Tx = W^TWW'^*x$
  - To hold for all x, we must have $W'^* = (W^TW)^{-1}W^T$ which is the pseudoinverse $W^\dagger$ of W
- So the cost simplifies to

$$\mathrm{C} = \sum_i \left\| x_i - WW^\dagger x_i \right\|^2$$

Ref: http://www.vision.jhu.edu/teaching/learning/datascience18/assets/Baldi_Hornik-89.pdf

The simplest autoencoder… acts like PCA

- Recall PCA: $Y = W^T X$ dimension reduction $x_i \in \mathbb{R}^n \to y_i \in \mathbb{R}^p$

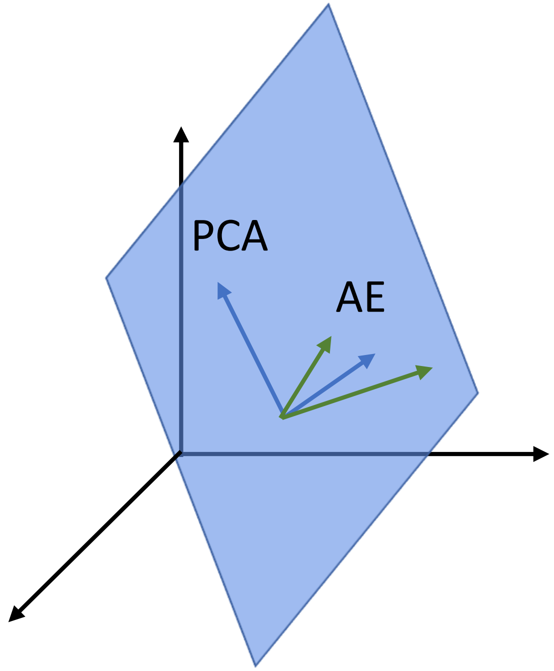
- Goal: find $W \in \mathbb{R}^{n \times p}$ such that each new basis vector maximally describes variance in the data

- Can frame PCA as minimization problem: $\sum_i \|x_i - W W^T x_i\|^2$

- PCA solution: choose $\mathrm{W} = U_p$, the first p eigenvectors from $XX^T = \mathrm{U}\Lambda\mathrm{U}^T$

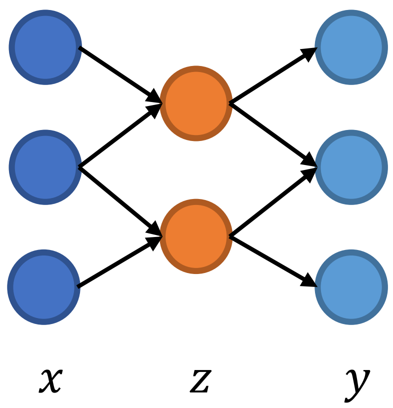
- Thus $W = U_p$ is a minimizer of $\mathrm{C} = \sum_i \left\| x_i - W W^{\dagger} x_i \right\|^2$
  - Could try to show this directly by taking tricky derivative $0 = \frac{\partial C}{\partial W} = \cdots$
  - Note it is not unique solution: any change-of-basis of U_p also works (e.g. non-orthogonal variants)

- Linear autoencoders project data onto the PCA subspace (with diff basis)
  - Suppose AE finds minimizer W, then its orthogonalization is U_p (up to rotation)
  - i.e. $W W^{\dagger} = W(W^T W)^{-1} W^T = U_p U_p^T$



Ref: https://arxiv.org/pdf/1804.10253.pdf, https://arxiv.org/pdf/1812.06775.pdf , see also Bengio et al, 2013 review Ch 7

Dimension reduction with autoencoders



- Classic “bottleneck” AE:
  - Restrict latent dimension $p < n$
  - performs dimension reduction on the data while explicitly preserving the data points
- Counterintuitive: same principle as unsupervised learning (e.g. MDS)
  - Unlike many unsupervised methods, a trained autoencoder offers a map to the latent space (and back) for new input data
- Remark:
  - Simple autoencoders reportedly perform poorly on untrained data
  - See 14.3 of Bengio/Goodfellow online text
- Extensions to address this:
  - adding more layers to the encoding and decoding (multilayer AEs)
  - fancier objective functions / constraints

http://www.deeplearningbook.org/contents/autoencoders.html

Variational Autoencoders (VAEs)

Input: $\{x_i\}_{i=1}^{M} \in \mathbb{R}^n$

Assume the data was created by:

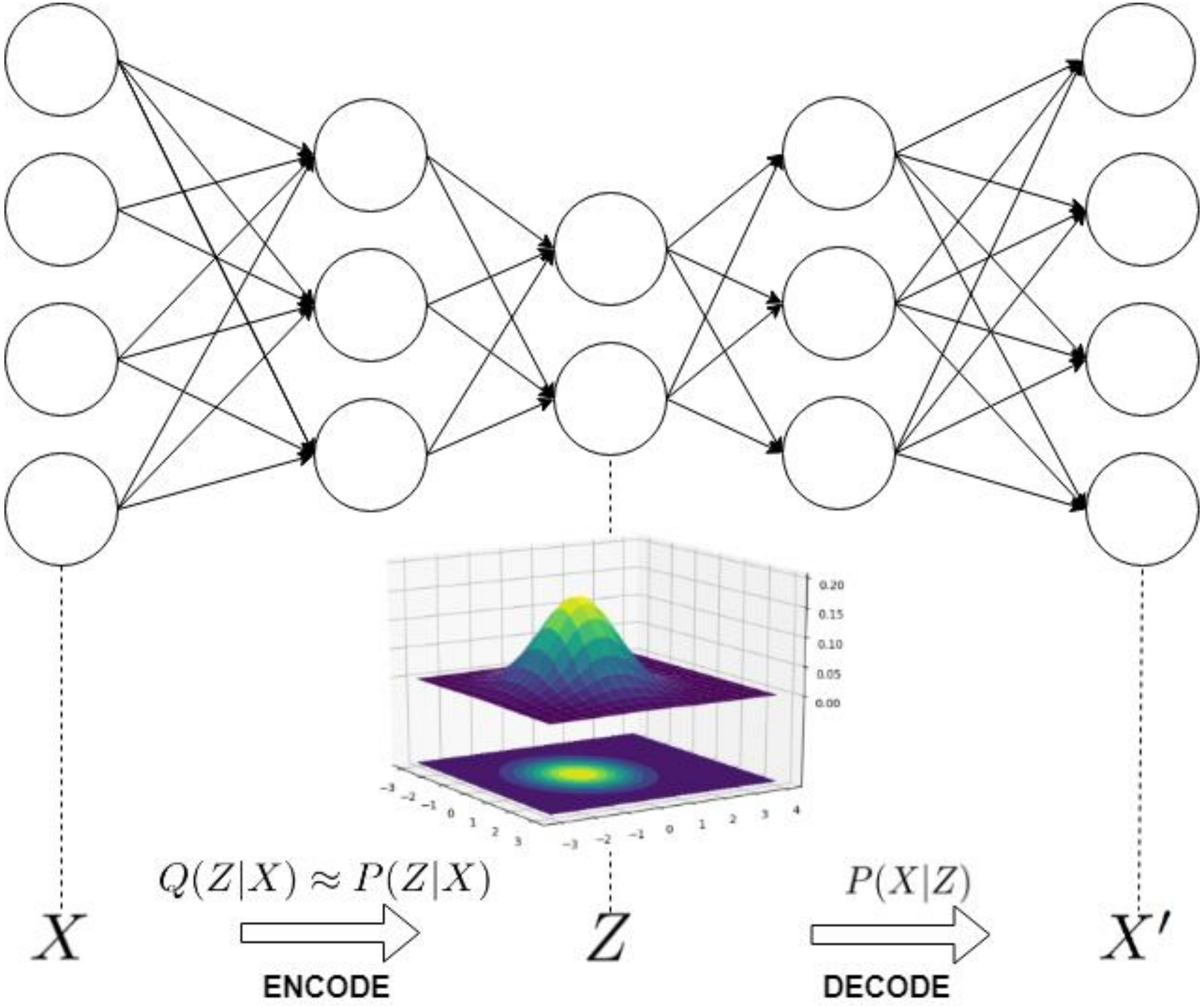
1. Sampling latent variables z_i from distribution $p_\theta(z)$
2. Sampling data x_i from conditional distribution $p_\theta(x|z)$

We want to find the “encoder” $q_\phi(z|x)$ and “decoder” $p_\theta(x|z)$ by optimizing with respect to parameters ϕ, θ

Cost function: $\boxed{L(x_i, \theta, \phi) = -D_{KL}\left[q_\phi(z|x_i) \middle| p_\theta(z)\right] + E_{q_\phi(z|x_i)}\left[\log\left(p_\theta(x_i|z)\right)\right]}$

- This is called the variational lower bound on $p(x_i)$ because $\log\left(p(x_i)\right) \geq L$
- We actually want to *maximize* L for this technique; imagine maximizing the log-likelihood of our data

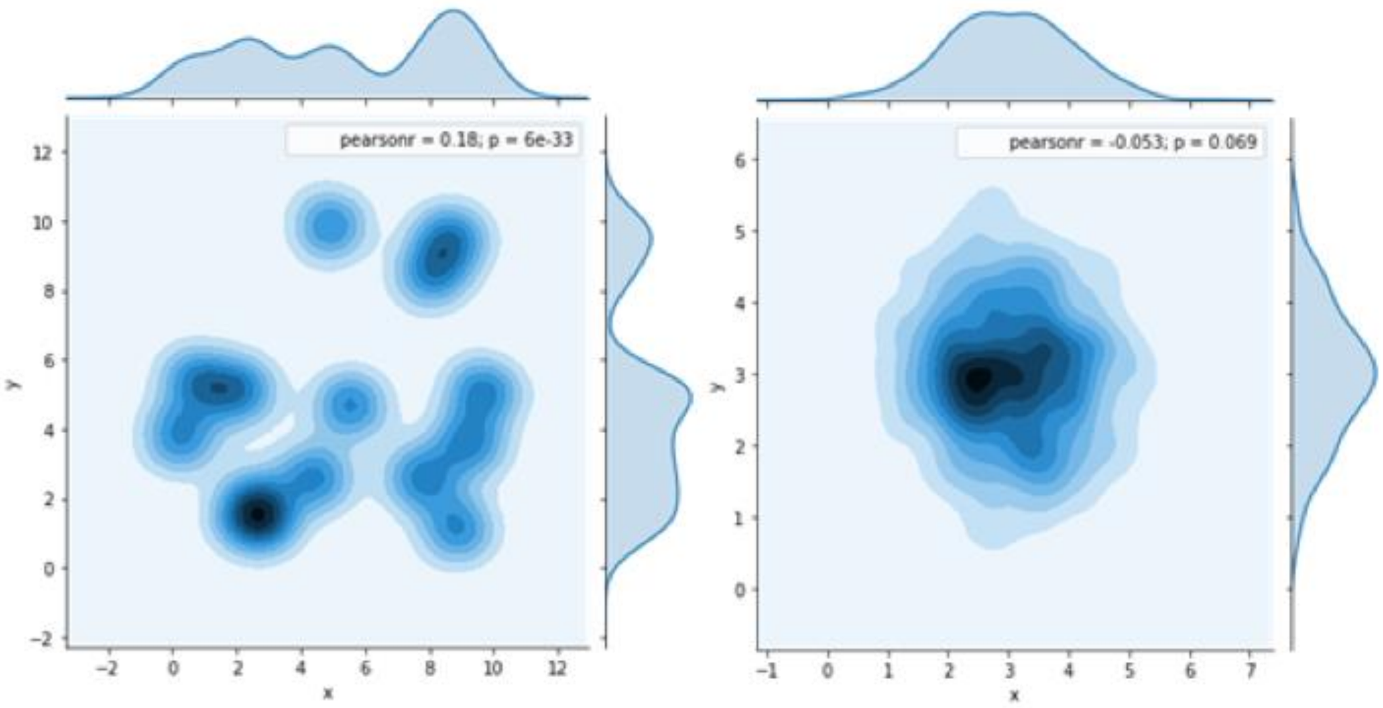
VAEs



https://news.sophos.com/en-us/2018/06/15/using-variational-autoencoders-to-learn-variations-in-data/

VAE Intuition

- The idea behind the VAE is that it allows you to incorporate a prior $p_\theta(z)$ which constrains the latent mapping to be more intuitive



https://news.sophos.com/en-us/2018/06/15/using-variational-autoencoders-to-learn-variations-in-data/

VAE Theory

The cost function can also be written like this:

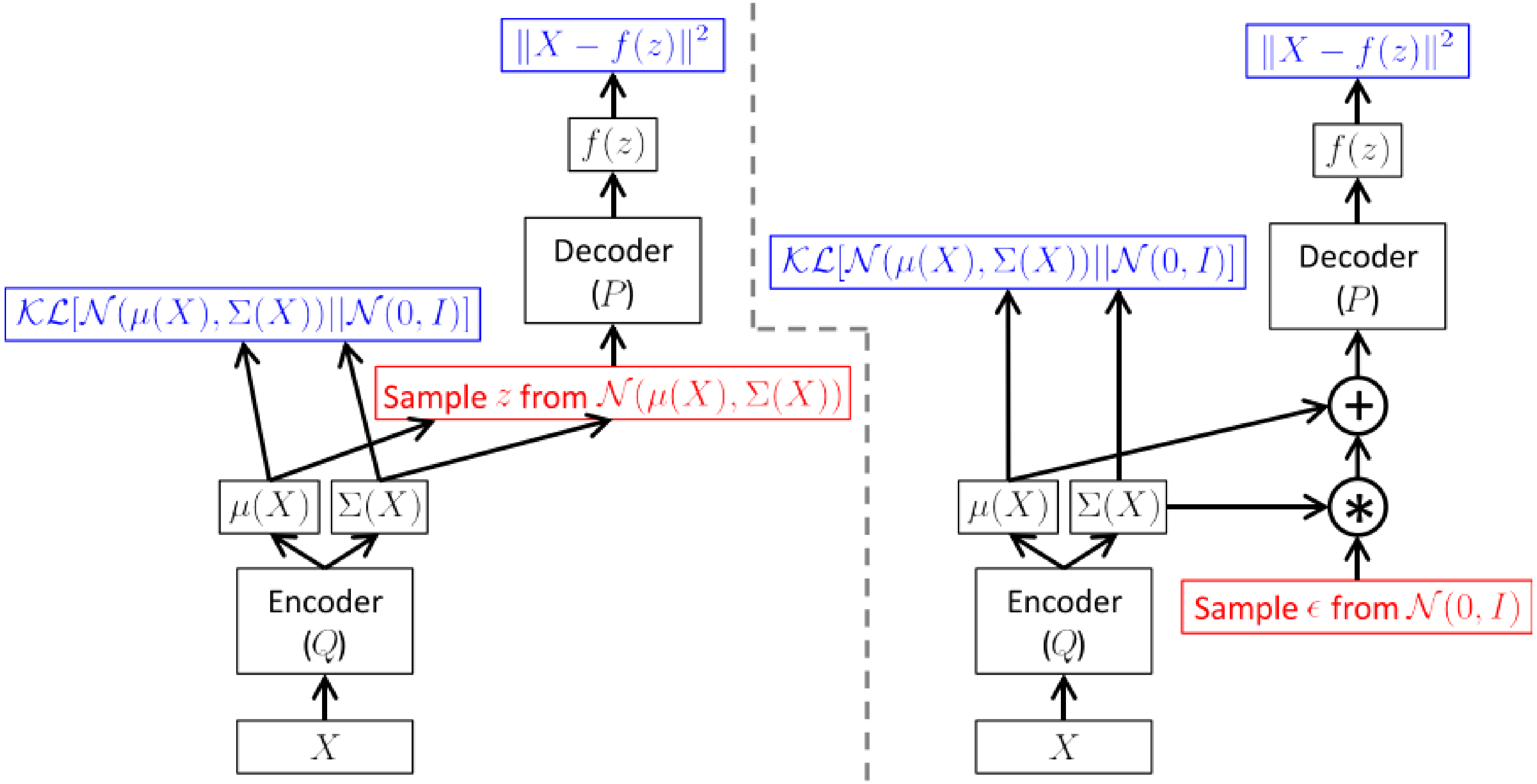
$$p_\theta(x) - \underbrace{D_{KL}[q_\phi(z|x)||p_\theta(z|x)]} = E_{q_\phi(z|x)}[\log(p_\theta(x|z))] - D_{KL}[q_\phi(z|x)|p_\theta(z)]$$

We want to maximize this

We want to minimize this

- The problem with the left-hand side is that it involves unknown distributions $p_\theta(x)$ and $p_\theta(z|x)$

The Reparameterization Trick



References

- Bengio 2013: Ch 6, **7**, 8, 9
- 2018 arxiv VAE PCA: https://arxiv.org/pdf/1812.06775.pdf, p3
- Other PCA ref: https://arxiv.org/pdf/1804.10253.pdf
- http://www.vision.jhu.edu/teaching/learning/datascience18/assets/Baldi_Hornik-89.pdf
- http://www.deeplearningbook.org/contents/autoencoders.html
- Kingma and Welling, 2013