Deep Learning & Engineering Applications

12. Variational Autoencoders (VAE)

JIDONG J. YANG

COLLEGE OF ENGINEERING

UNIVERSITY OF GEORGIA

Supervised vs. unsupervised learning

Most of the supervised learning algorithms are inherently **discriminative**, which means they learn how to model the conditional probability distribution function, $\boldsymbol{p(y|x)}$.

Although we could make predictions with this probability distribution function, we are not allowed to sample new instances from the input distribution directly.

We need generative models!

Taxonomy of Generative Models

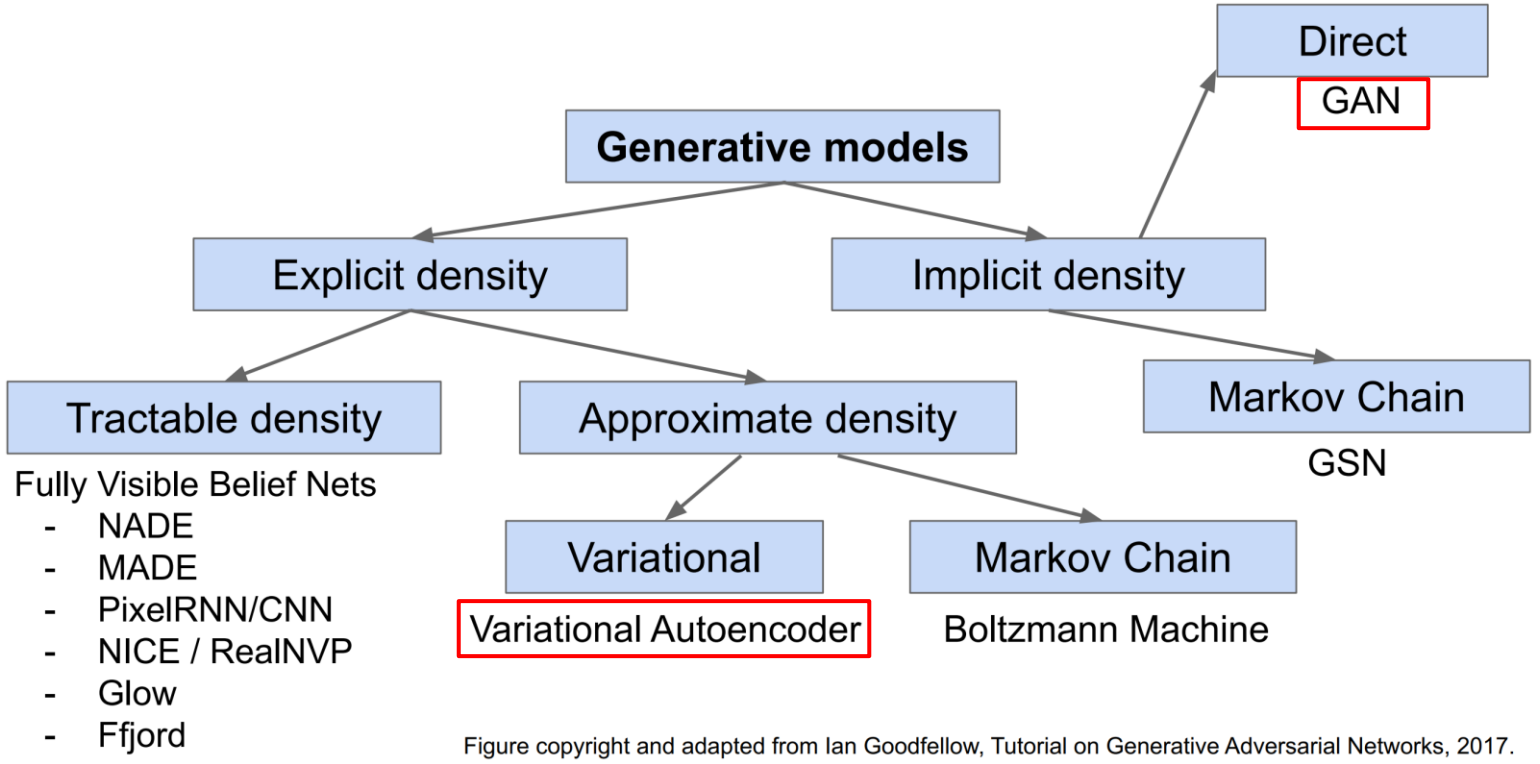
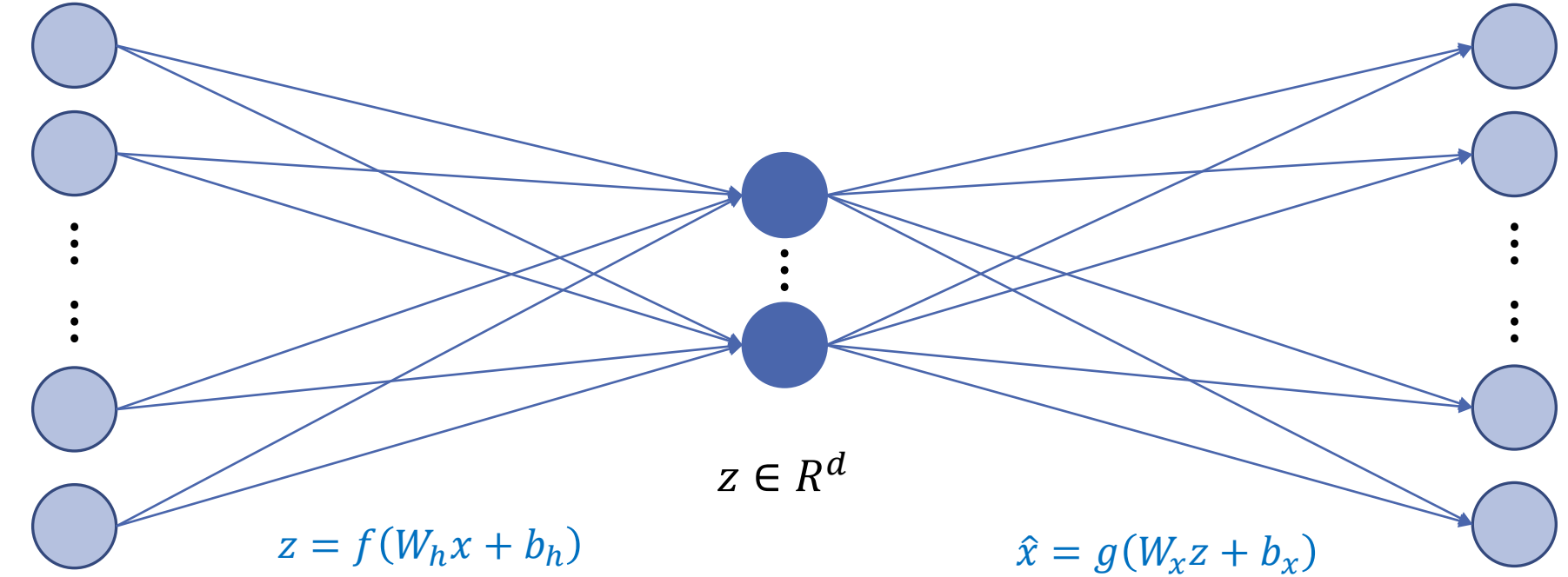


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

Image credit: Fei-Fei Li, Ranjay Krishna, Danfei Xu, Lecture 12: Generative Models, May 11, 2021.
The Original Figure is from Ian Goodfellow, NIPS 2016 Tutorial: Generative Adversarial Networks https://arxiv.org/pdf/1701.00160.pdf

Autoencoder (AE)

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



Under-complete vs. Over-complete Hidden Layer

Does hidden dimension have to be less than the input/output dimension?

What if the hidden dimension is same as or greater than the input dimension?

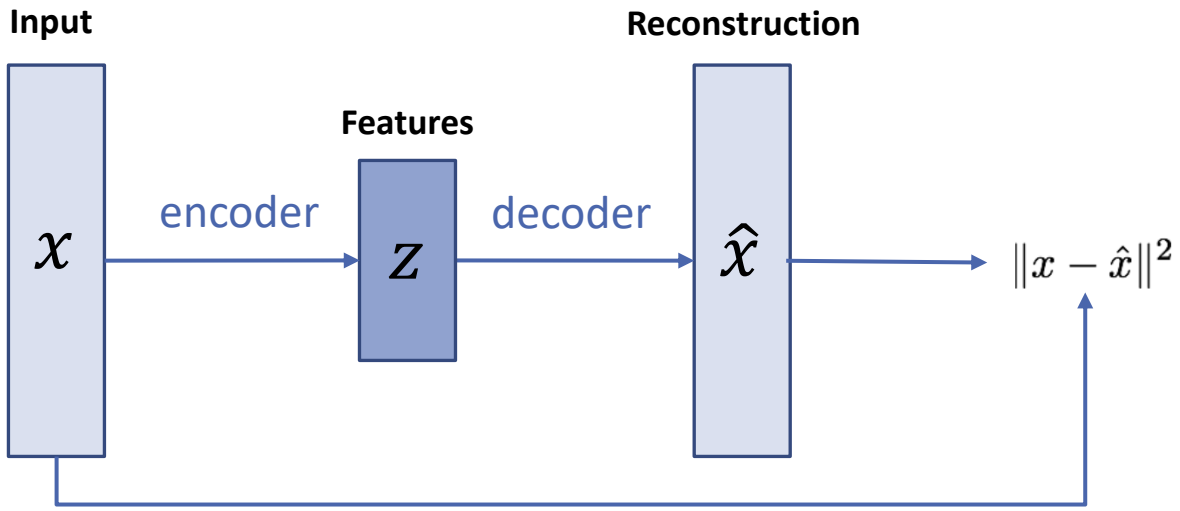
[Ng, lecture note] https://web.stanford.edu/class/cs294a/sparseAutoencoder.pdf
[Rolfe & LeCun, 2013] https://arxiv.org/pdf/1301.3775.pdf

Autoencoder (AE)

Learn representative feature in a lower dimension space.

Train the AE with bottleneck to capture “representative features” in low-dimensional space (compression), which can be used to reconstruct original data.

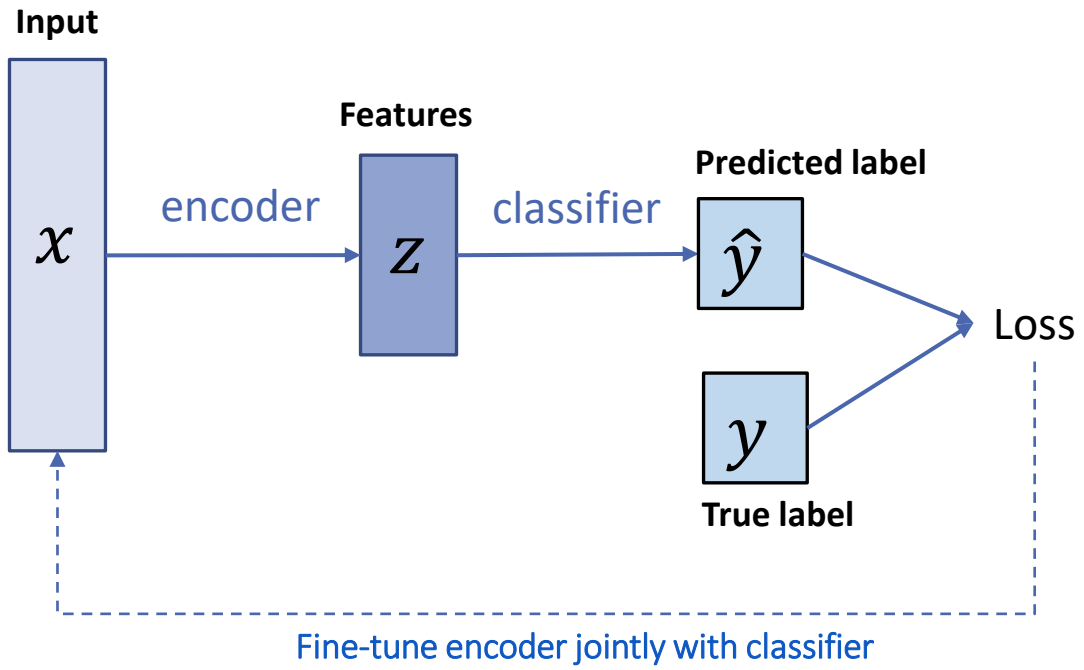
“Autoencoder”: Encode input itself.



Common use of AE

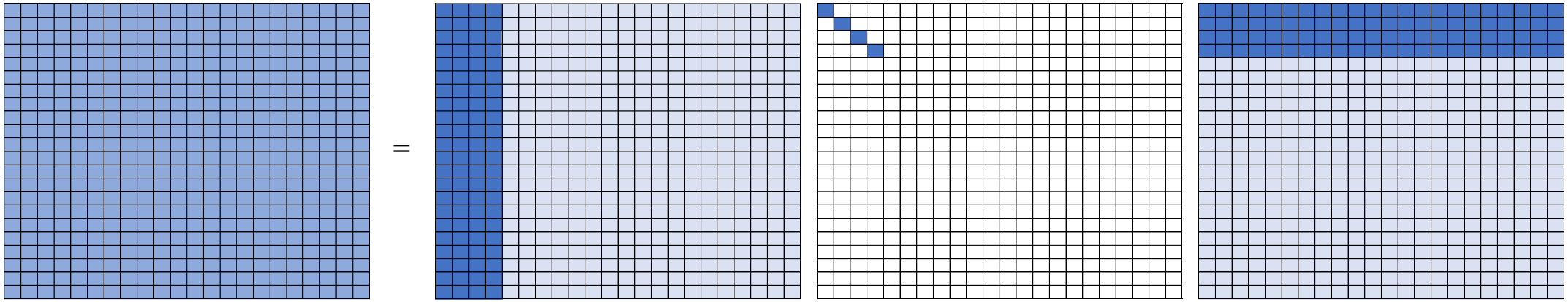
After training, throw away decoder.

Use encoder to initialize a supervised model.

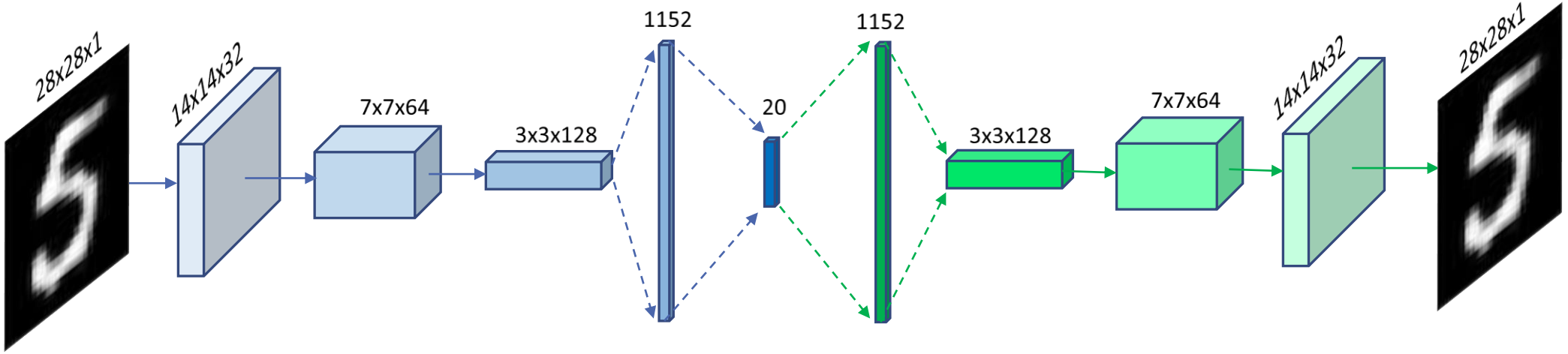


Recall PCA

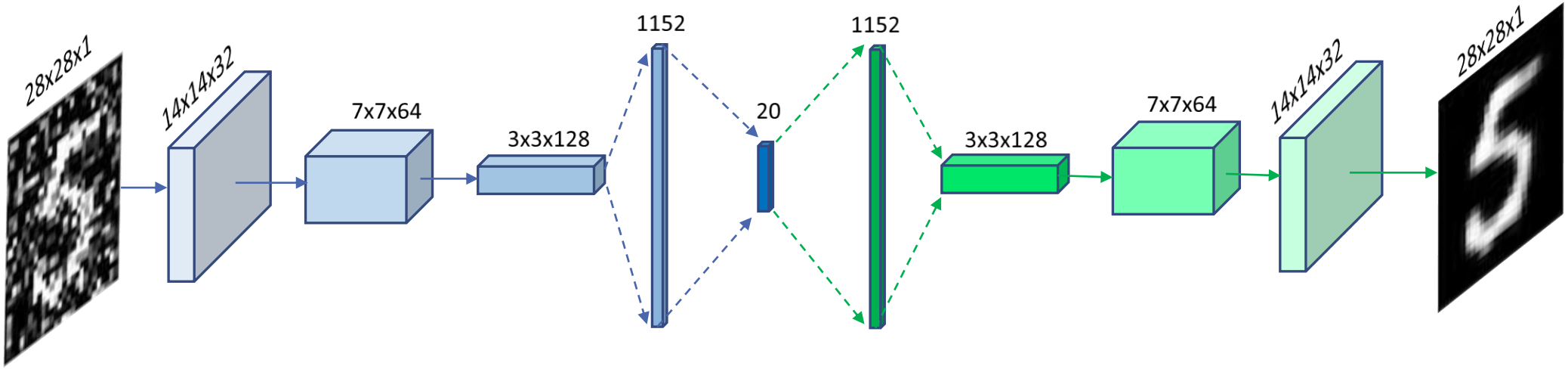
Express the original matrix as a linear combination of k low-rank matrices associated with first k largest singular values.



AE architecture example



Denoising Autoencoder (DAE)



Variational Autoencoder (VAE)

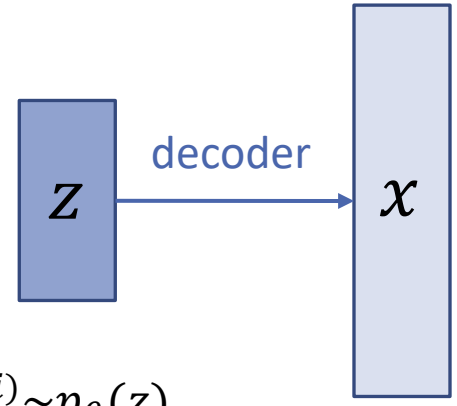
In summary, autoencoders can

- reconstruct data and learn features to initialize a supervised model.
- learned features capture factors of variation in training data.
- use as a denoiser by adding noises to inputs.
- Can we generate new data (e.g., images) from an autoencoder?

Variational Autoencoders

- Probabilistic spin on autoencoders
- Allow to generate data by sampling in the latent space.

Can we generate new x by sampling z?



$z^{(i)} \sim p_\theta(z)$

Prior

$x \sim p_\theta(x|z^{(i)})$

Conditional

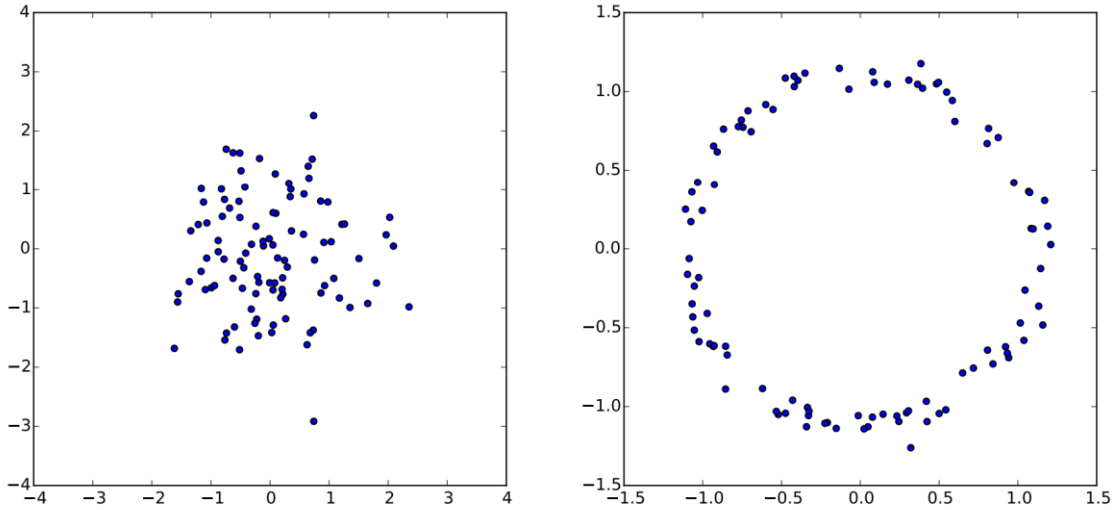
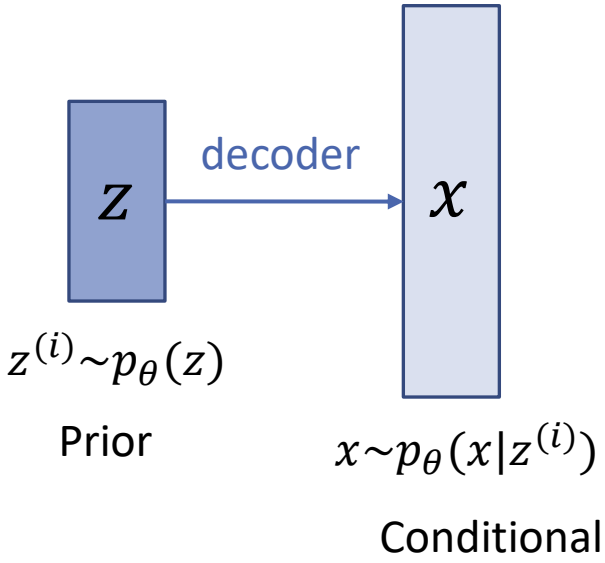


Figure 2: Given a random variable z with one distribution, we can create another random variable $X = g(z)$ with a completely different distribution. Left: samples from a gaussian distribution. Right: those same samples mapped through the function $g(z) = z/10 + z/||z||$ to form a ring. This is the strategy that VAEs use to create arbitrary distributions: the deterministic function g is learned from data.

[Doersch 2016] https://arxiv.org/pdf/1606.05908.pdf

Generate new x by sampling in z



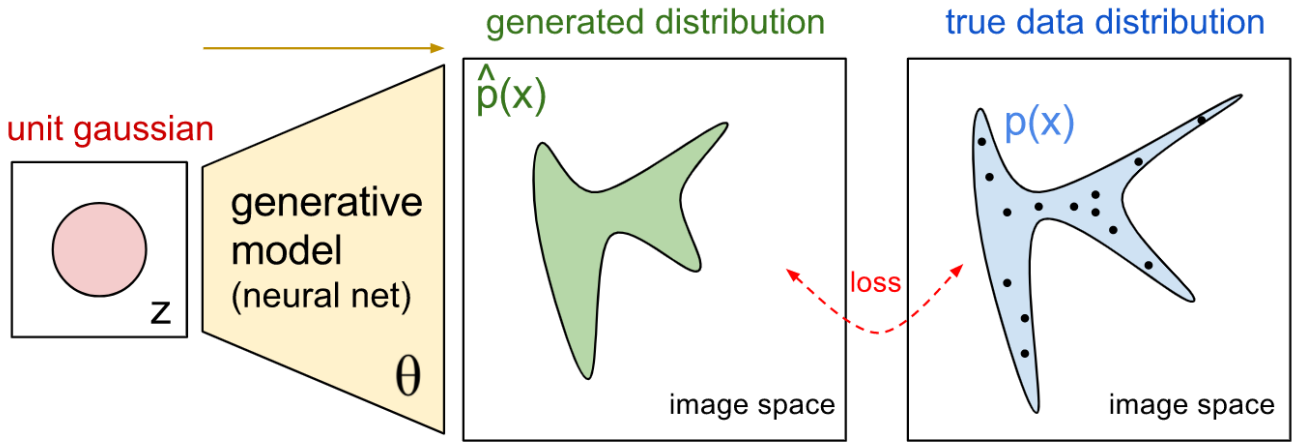
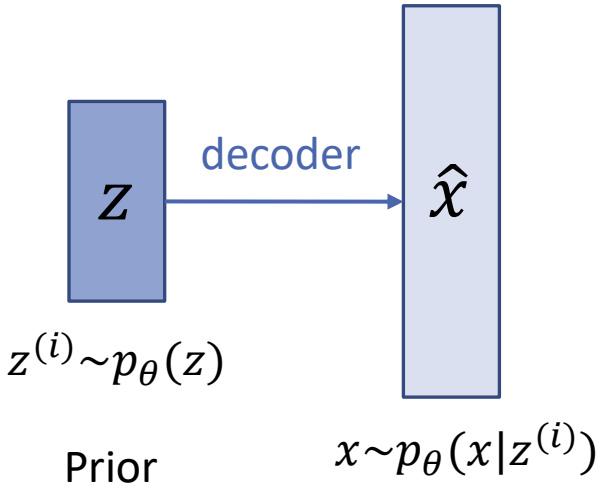


Image credit: https://openai.com/blog/generative-models/

We would like to estimate θ given training data x.

How to formulate the problem?



We would like to estimate θ given training data x.

Maximize likelihood of training data, i.e.,

Data likelihood: $p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$

Gaussian NN

Monte-Carlo estimation: $\log p_\theta(x) \approx log \left[\frac{1}{K} \sum_{i=1}^{K} p_\theta(x|z^{(i)}) \right]$

Posterior density: $p_\theta(z|x) = \frac{p_\theta(x|z) p_\theta(z)}{p_\theta(x)}$

Both data likelihood and posterior density are intractable.

$$p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz$$

$$p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)}$$

Solutions?

"Decouple" the above two equations by learning $\boldsymbol{q_\phi(z|x)}$ to approximate the posterior $\boldsymbol{p_\theta(z|x)}$

i.e., approximate the unknown posterior distribution from the observed data x (**variational inference**).

We will see this allows us to derive a lower bound on the data likelihood that is tractable and optimizable.

$$\log p_\theta\left(x^{(i)}\right) = \boldsymbol{E}_{z\sim q_\phi(z|x^{(i)})}[\log p_\theta(x)]$$

This will be handled by sampling from z parametrized by the encoder network.
We will talk about this shortly.

[Kingma & Welling, 2014] https://arxiv.org/pdf/1312.6114.pdf

Variational Autoencoders (VAE)

$$\log p_\theta\left(x^{(i)}\right) = \boldsymbol{E}_{z\sim q_{\boldsymbol{\phi}}\left(z|x^{(i)}\right)}[\log p_\theta(x)]$$

$$= \boldsymbol{E}_z\left[\log \frac{p_\theta\left(x^{(i)}|z\right)p_\theta(z)}{p_\theta\left(z|x^{(i)}\right)}\right] \qquad \text{Bayes' Rule}$$

$$= \boldsymbol{E}_z\left[\log \frac{p_\theta\left(x^{(i)}|z\right)p_\theta(z)}{p_\theta\left(z|x^{(i)}\right)}\frac{q_{\boldsymbol{\phi}}\left(z|x^{(i)}\right)}{q_{\boldsymbol{\phi}}\left(z|x^{(i)}\right)}\right]$$

$$= \boldsymbol{E}_z\left[\log g\left(p_\theta\left(x^{(i)}|z\right)\frac{p_\theta(z)}{q_{\boldsymbol{\phi}}\left(z|x^{(i)}\right)}\frac{q_{\boldsymbol{\phi}}\left(z|x^{(i)}\right)}{p_\theta\left(z|x^{(i)}\right)}\right)\right]$$

$$= \boldsymbol{E}_z\left[\log g\, p_\theta\left(x^{(i)}|z\right)\right] - \boldsymbol{E}_z\left[\log \frac{q_{\boldsymbol{\phi}}\left(z|x^{(i)}\right)}{p_\theta(z)}\right] + \boldsymbol{E}_z\left[\log \frac{q_{\boldsymbol{\phi}}\left(z|x^{(i)}\right)}{p_\theta\left(z|x^{(i)}\right)}\right]$$

$$= \boldsymbol{E}_z\left[\log g\, p_\theta\left(x^{(i)}|z\right)\right] - \boldsymbol{D_{KL}}\left(q_{\boldsymbol{\phi}}\left(z|x^{(i)}\right)||\, p_\theta(z)\right) + \boldsymbol{D_{KL}}\left(q_{\boldsymbol{\phi}}\left(z|x^{(i)}\right)||\, p_\theta\left(z|x^{(i)}\right)\right)$$

Sampling via decoder

Gaussians for encoder & z prior

Intractable; KL ≥ 0

Likelihood Lower Bound

$$\log p_\theta\left(x^{(i)}\right) = \boldsymbol{E}_z\left[\log p_\theta\left(x^{(i)}|z\right)\right] - \boldsymbol{D_{KL}}\left(q_{\boldsymbol{\phi}}\left(z|x^{(i)}\right)||\, p_\theta(z)\right) + \boldsymbol{D_{KL}}\left(q_{\boldsymbol{\phi}}\left(z|x^{(i)}\right)||\, p_\theta\left(z|x^{(i)}\right)\right)$$

Tractable lower bound, which can be optimized. Both terms are differentiable.

≥ 0

We can maximize the loglikelihood lower bound:

$$LL\left(x^{(i)}, \theta, w\right) = \boldsymbol{E}_z\left[\log p_\theta\left(x^{(i)}|z\right)\right] - \boldsymbol{D_{KL}}\left(q_w\left(z|x^{(i)}\right)||\, p_\theta(z)\right)$$

This is equivalent to minimize the following loss function:

$$Loss\left(x^{(i)}, \theta, w\right) = \boldsymbol{E}_z\left[-\log p_\theta\left(x^{(i)}|z\right)\right] + \boldsymbol{D_{KL}}\left(q_w\left(z|x^{(i)}\right)||\, p_\theta(z)\right)$$

$$Loss(x^{(i)}, \theta, w) = \boldsymbol{E}_z[-\log p_\theta(x^{(i)}|z)] + \boldsymbol{D_{KL}}\left(q_w(z|x^{(i)})||\, p_\theta(z)\right)$$

Reconstruction Loss

For continuous data, we could choose Gaussian:

$$p(x|z) \sim N(f(z), \sigma^2 I)$$

So, we have $\log p(x|z) = C - \frac{1}{2\sigma^2}\|x - f(z)\|^2$

$$l_{recon}(x, \hat{x}) = \boldsymbol{E}_z[-\log p_\theta(x^{(i)}|z)] = \frac{1}{2}\|x - f(z)\|^2$$

$$= \frac{1}{2}\|x - \hat{x})\|^2$$

For categorical data, we could use cross-entropy:

$$l_{recon}(x, \hat{x}) = \boldsymbol{E}_z[-\log p_\theta(x^{(i)}|z)] = -\sum_{c=1}^{M} x^{(i)} \log p_\theta(x^{(i)}|z)$$

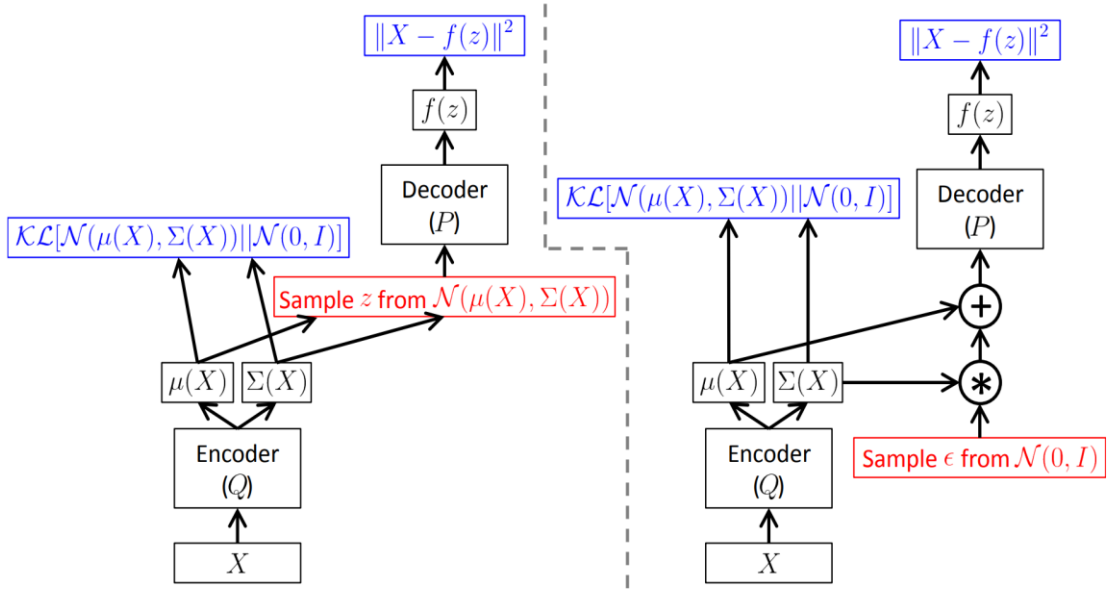


Figure 4: A training-time variational autoencoder implemented as a feed-forward neural network, where $P(X|z)$ is Gaussian. Left is without the "reparameterization trick", and right is with it. Red shows sampling operations that are non-differentiable. Blue shows loss layers. The feedforward behavior of these networks is identical, but backpropagation can be applied only to the right network.

[Doersch 2016] https://arxiv.org/pdf/1606.05908.pdf

KL divergence of Gaussians

KL-divergence between two multivariate Gaussian distributions can be computed in closed form:

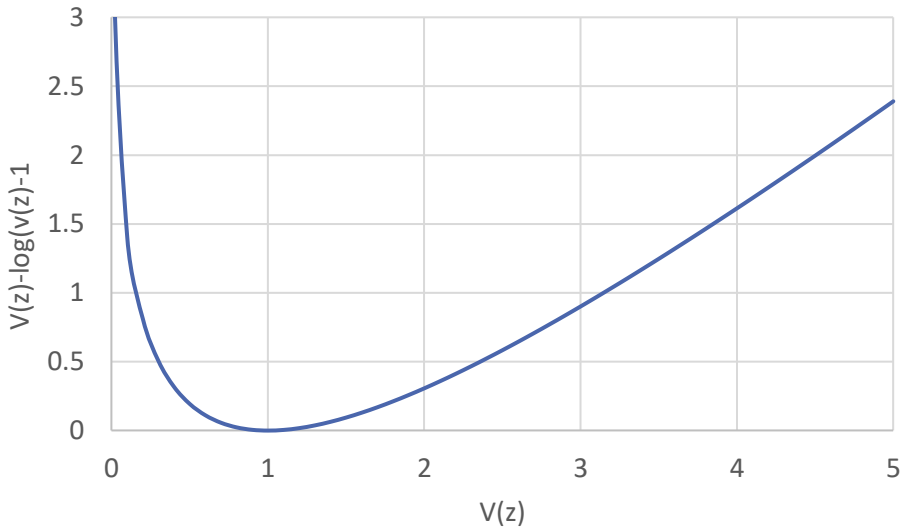
$$\mathcal{D}[\mathcal{N}(\mu_0,\Sigma_0)\|\mathcal{N}(\mu_1,\Sigma_1)] = \frac{1}{2}\left(\mathrm{tr}\left(\Sigma_1^{-1}\Sigma_0\right) + (\mu_1-\mu_0)^\top \Sigma_1^{-1}(\mu_1-\mu_0) - k + \log\left(\frac{\det\Sigma_1}{\det\Sigma_0}\right)\right)$$

$$\mathcal{D}[\mathcal{N}(\mu(X),\Sigma(X))\|\mathcal{N}(0,I)] = \frac{1}{2}\left(\mathrm{tr}\left(\Sigma(X)\right) + (\mu(X))^\top(\mu(X)) - k - \log\det\left(\Sigma(X)\right)\right)$$

$$l_{KL}(z,N(0,I_k)) = \frac{1}{2}\sum_{i=1}^{k}\left(V(z_i) + E(z_i)^2 - 1 - \log V(z_i)\right)$$

$$= \frac{1}{2}\sum_{i=1}^{k}\left(V(z_i) - \log V(z_i) - 1 + E(z_i)^2\right)$$

$V(z_i) \to 1$ $\quad$ $E(z_i) \to 0$



Loss Function

$$Loss\left(x^{(i)}, \theta, w\right) = \boldsymbol{E}_z\left[-\log p_\theta\left(x^{(i)}|z\right)\right] + \boldsymbol{D_{KL}}\left(q_w\left(z|x^{(i)}\right)||\, p_\theta(z)\right)$$

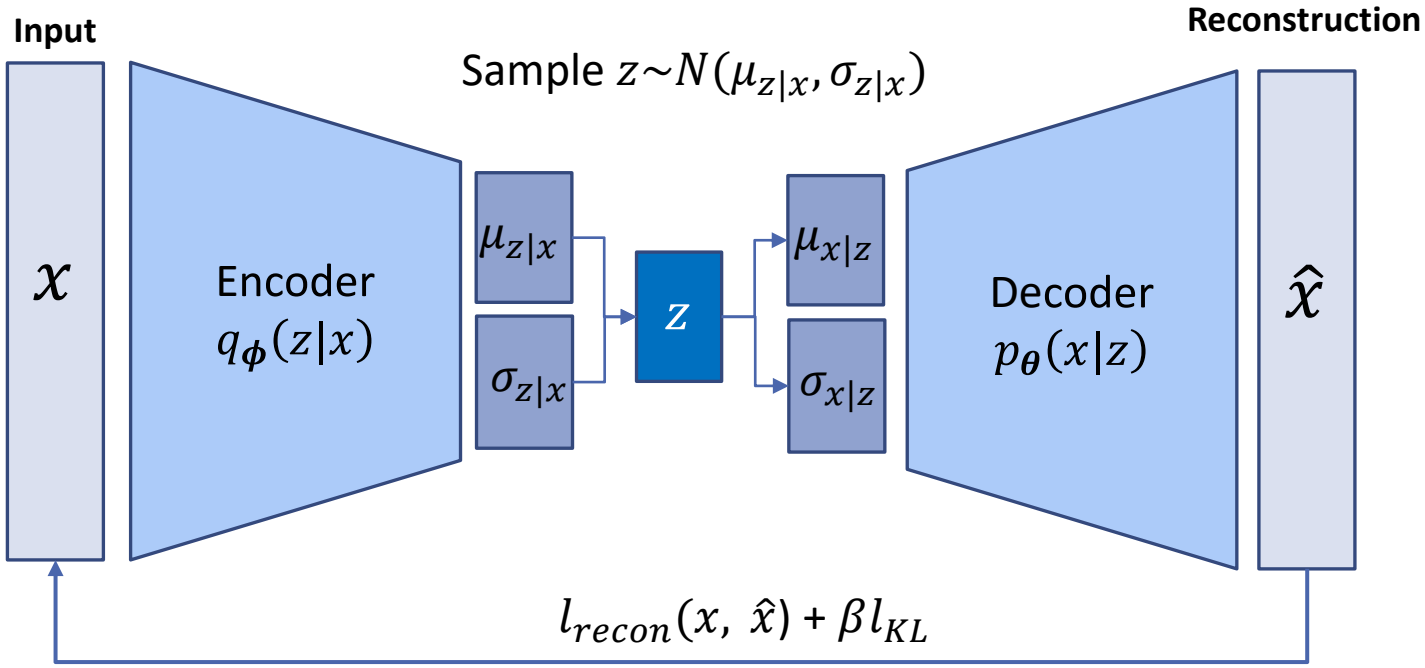
$$Loss\left(x^{(i)}, \theta, w\right) = l_{recon}(x,\ \hat{x}) + \beta l_{KL}(z, N(0, I_k))$$

$$= l_{recon}(x,\ \hat{x}) + \frac{\beta}{2}\sum_{i=1}^{k}\left(V(z_i) - \log V(z_i) - 1 + E(z_i)^2\right)$$

Train a VAE

$$z = \mu_{z|x} + \epsilon\sigma_{z|x}$$

Sample
$\epsilon \sim N(0, I)$

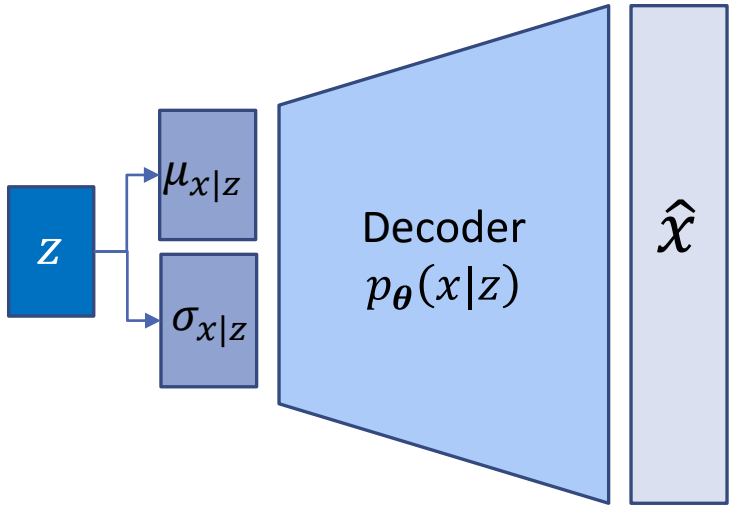


Generate data

Given a trained VAE, we can use decoder network & sample z from the prior to generate data.

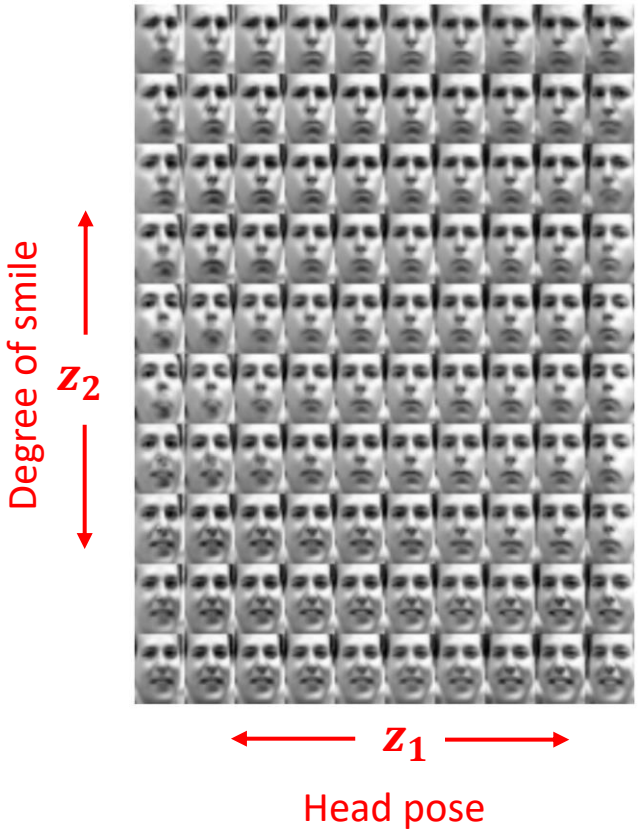
$$z = \mu_{z|x} + \epsilon\sigma_{z|x}$$

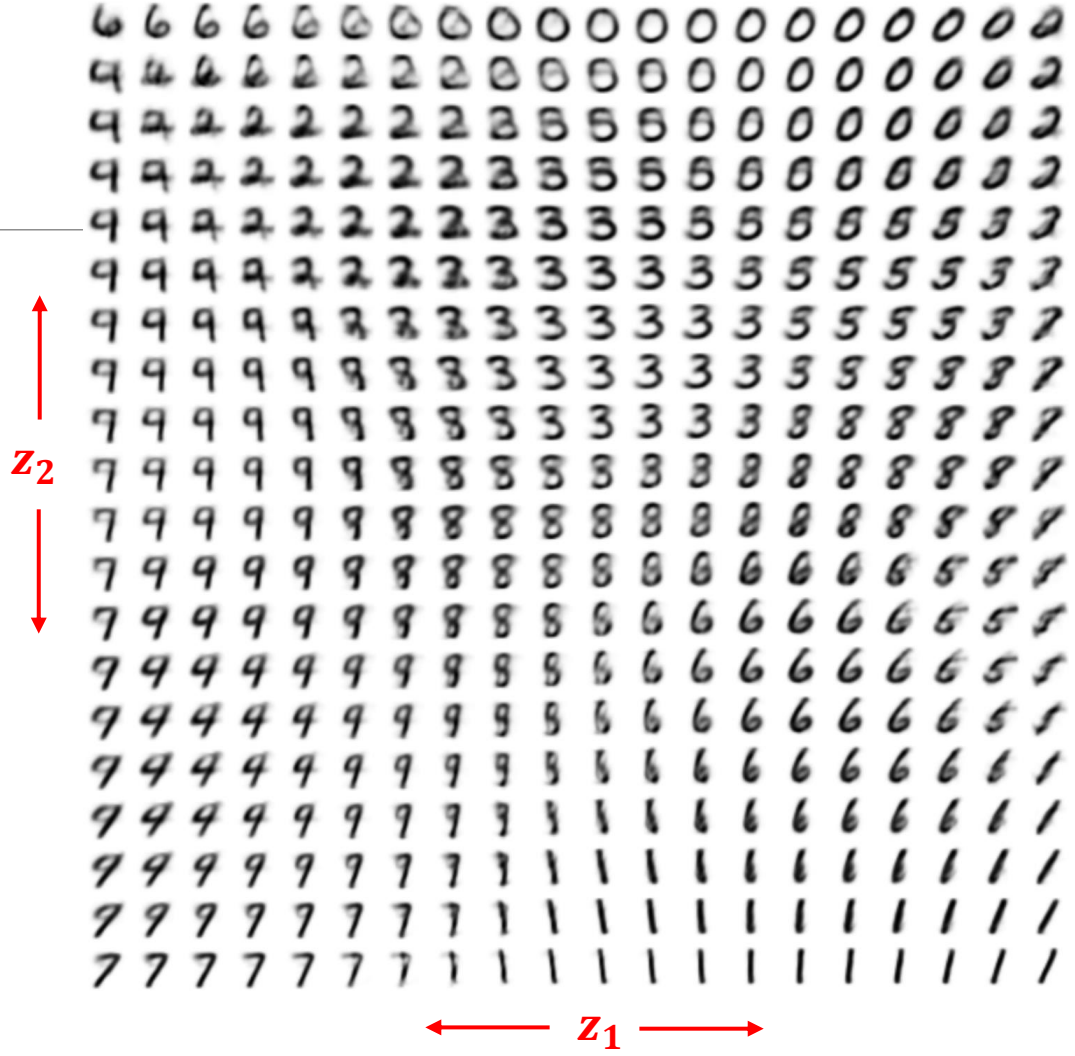
Sample
$\epsilon \sim N(0, I)$



VAE

Learn low-dimensional data manifold in latent space





[Kingma & Welling, 2003] https://arxiv.org/pdf/1312.6114v10.pdf