



Variational Autoencoders (VAEs)

DL4DS – Spring 2025

DS542 Gardos

Prince, *Understanding Deep Learning*,

[Rocca, "Understanding Variational Autoencoders \(VAEs\)", 2019](#)

Other Content Cited

April/May Dates



Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
		April 1	2	3 Xformers Part 2	4	5
6	7	8 Industry Talk	9	10 Improving LLMs	11	12
13	14	15 GANs	16	17 VAEs	18	19
20	21	22 Diffusion Models	23	24 Graph NNs	25	26
27	28	29 ★ Project Presentations 1 ★	30	May 1 ★ Project Presentations 2 ★	2	3
4	5 Project Reports Due	6	7	8	8	10
			Finals Week			

Diederik P. Kingma

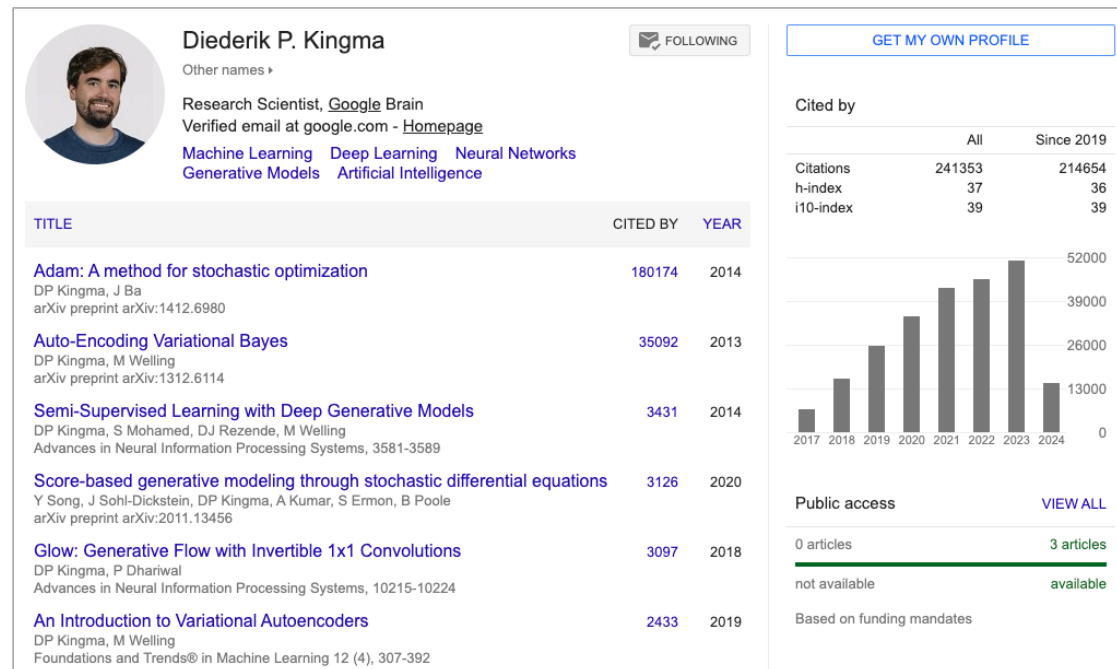


2016 OpenAI, founding member
2017 PhD U. of Amsterdam
2018– Google DeepMind

Auto-Encoding Variational Bayes

Diederik P. Kingma
Machine Learning Group
Universiteit van Amsterdam
dpkingma@gmail.com

Max Welling
Machine Learning Group
Universiteit van Amsterdam
welling.max@gmail.com



Variational Autoencoder



Variational Inference: A method from machine learning that approximates probability densities through optimization.

Autoencoder: A type of artificial neural network used to learn efficient codings of unlabeled data in an unsupervised manner.

VAE is an autoencoder whose encodings distribution is regularized during the training to ensure that its latent space has good properties allowing us to generate new data.

Auto-Encoding Variational Bayes

Autoencoder: A type of artificial neural network used to learn efficient codings of unlabeled data in an unsupervised manner.

Variational Inference: A method from machine learning that approximates probability densities through optimization.

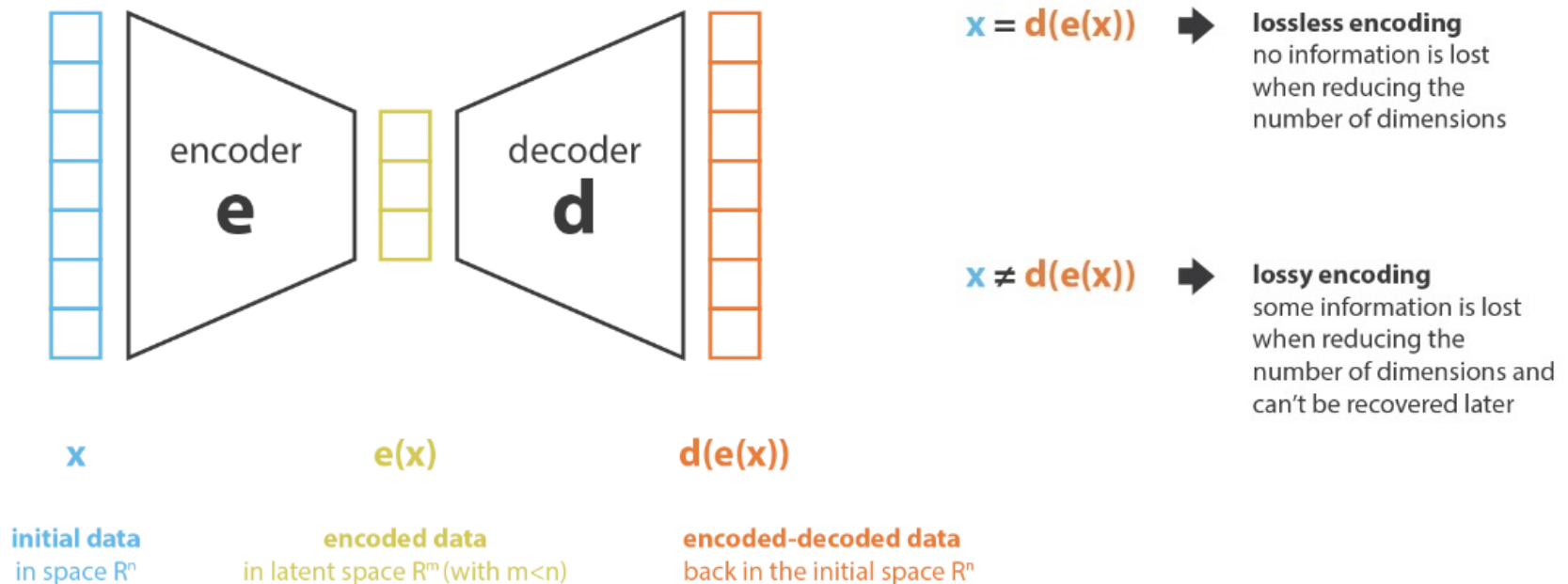
Bayesian since joint density is decomposed into prior and posterior density distributions using Bayes Rule:

$$p(\mathbf{z}, \mathbf{x}) = p(\mathbf{x}|\mathbf{z}) p(\mathbf{z})$$

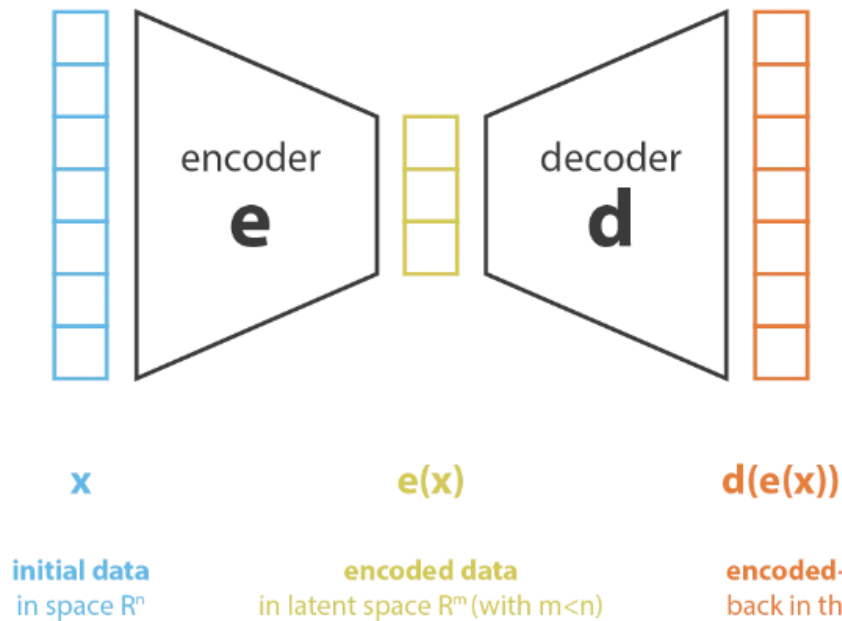
Outline

- Autoencoder and its limitations
- Intuition behind VAEs
- Derivation of VAE
- Example applications

Dimensionality reduction with an autoencoder



Dimensionality reduction with an autoencoder



We want to find the best encoder, e , and decoder, d , to minimize the error between x and $d(e(x))$.

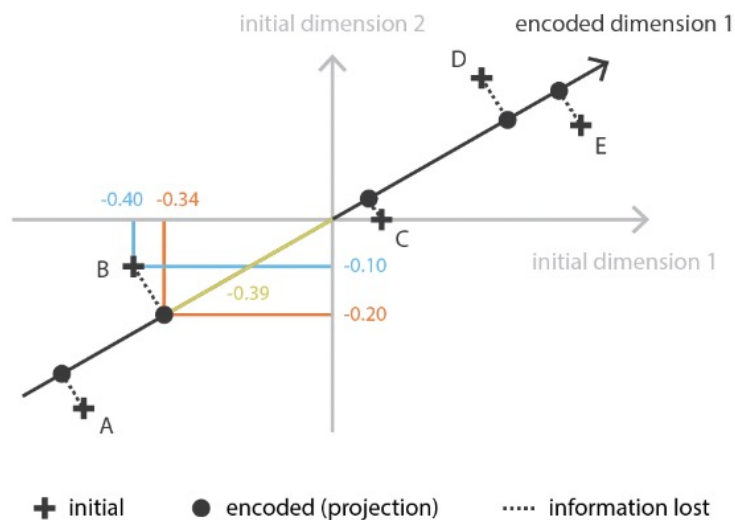
$$(e^*, d^*) = \underset{(e, d) \in E \times D}{\operatorname{argmin}} \epsilon(x, d(e(x)))$$

where

$$\epsilon(x, d(e(x)))$$

is the reconstruction error.

Dimensionality reduction with Principal Component Analysis (PCA)



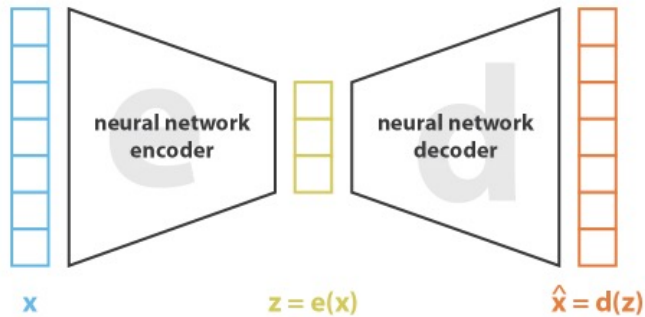
$$n_d = 2 \quad n_e = 1$$

Point	Initial	Encoded	Decoded
A	(-0.50, -0.40)	-0.63	(-0.54, -0.33)
B	(-0.40, -0.10)	-0.39	(-0.34, -0.20)
C	(0.10, 0.00)	0.09	(0.07, 0.04)
D	(0.30, 0.30)	0.41	(0.35, 0.21)
E	(0.50, 0.20)	0.53	(0.46, 0.27)

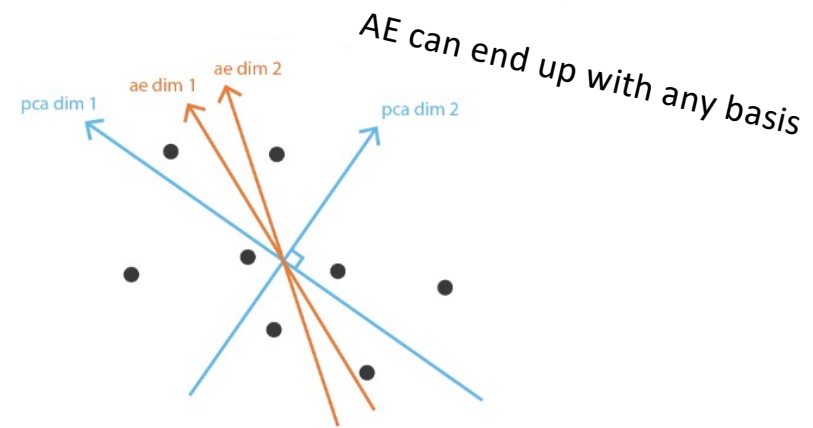
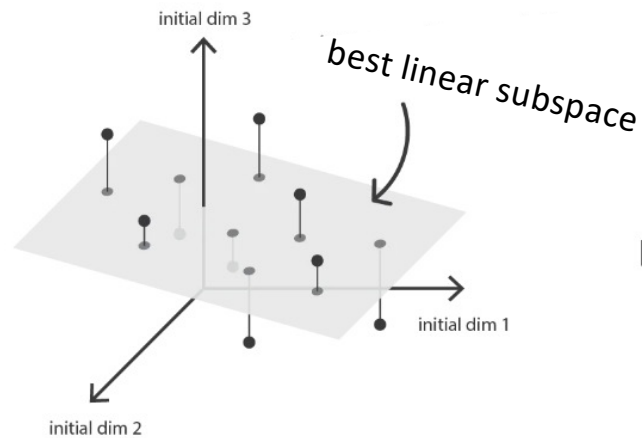
Project the n_d -dimensional features onto an orthogonal n_e -dimensional subspace that minimizes Euclidean distance.

Linear Transformation!!

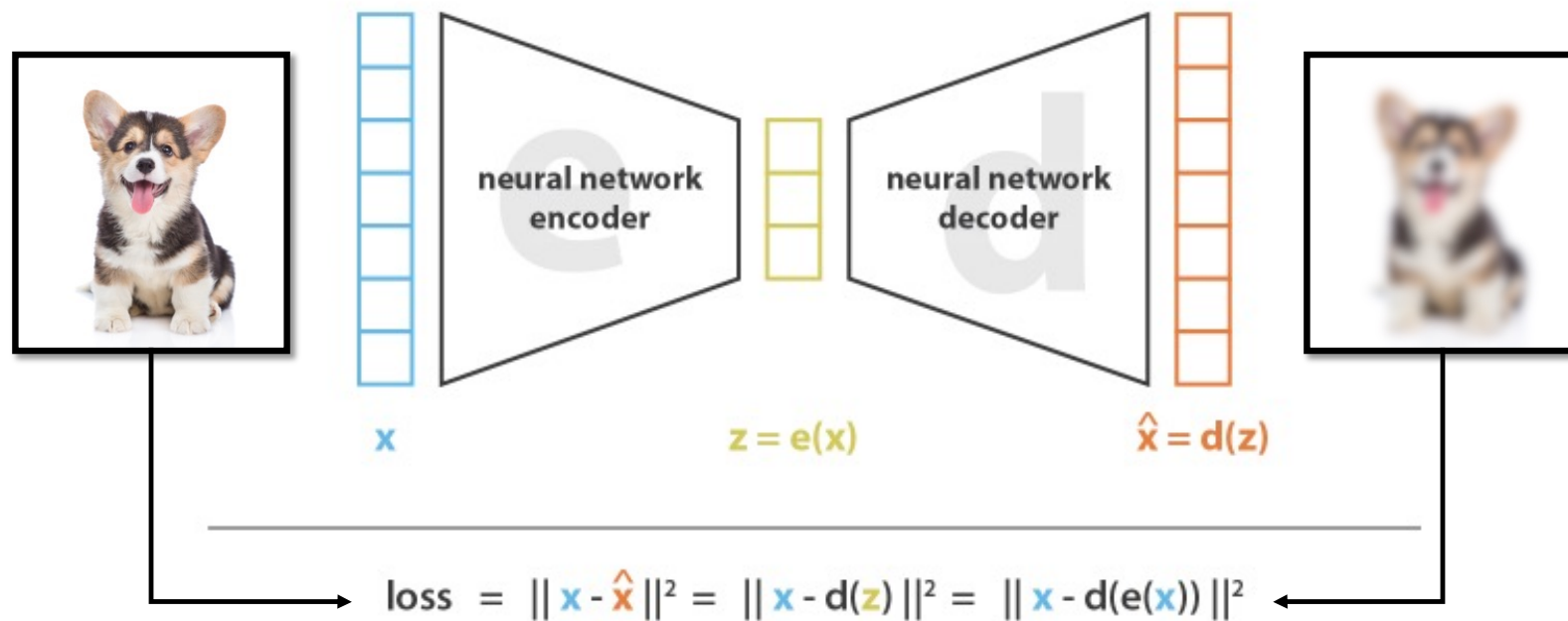
Neural Network Autoencoder – 1 Linear Layer



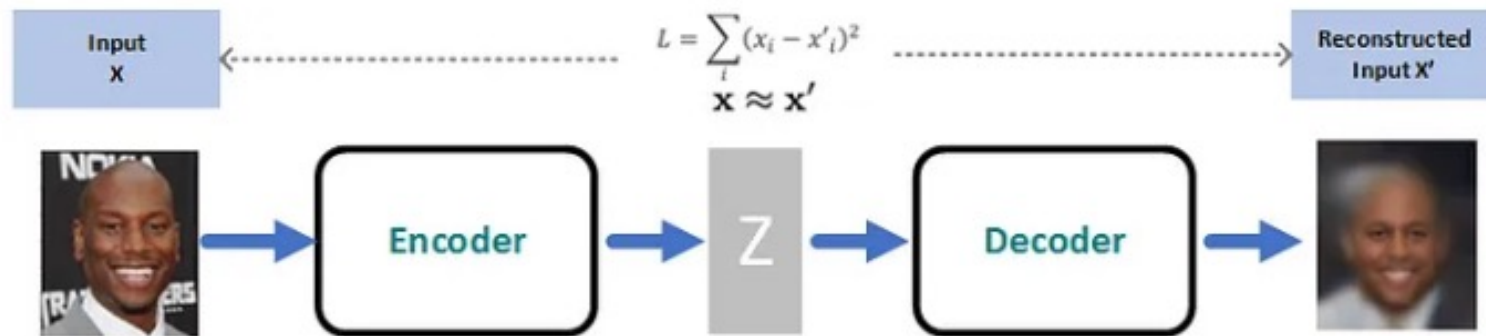
We could define encoder and decoder to each have one linear layer (no activation function), but it wouldn't necessarily converge during training to PCA solution.



Neural Network Autoencoder



Autoencoder Reconstruction



Trained on CelebA dataset.

Auto Encoder Code Examples

1. Autoencoder with MNIST

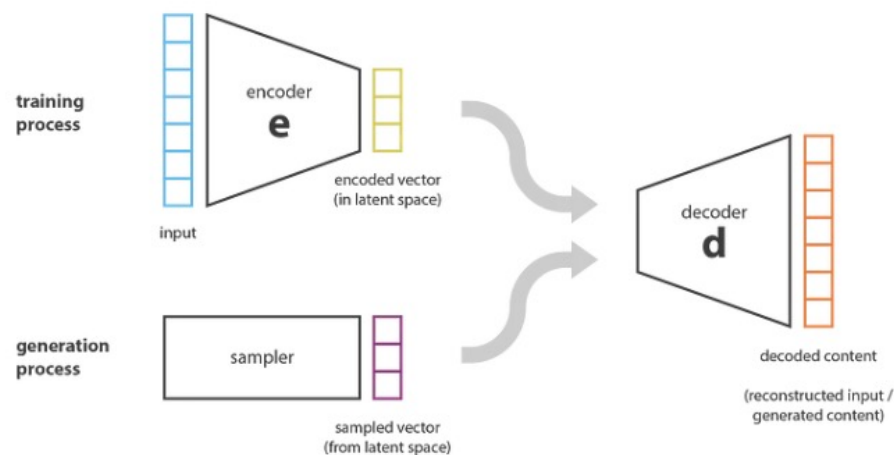


2. Autoencoder with CIFAR10



https://github.com/DL4DS/sp2025_homeworks/tree/main/lecture_collateral/vae

Can we generate new samples with autoencoder?



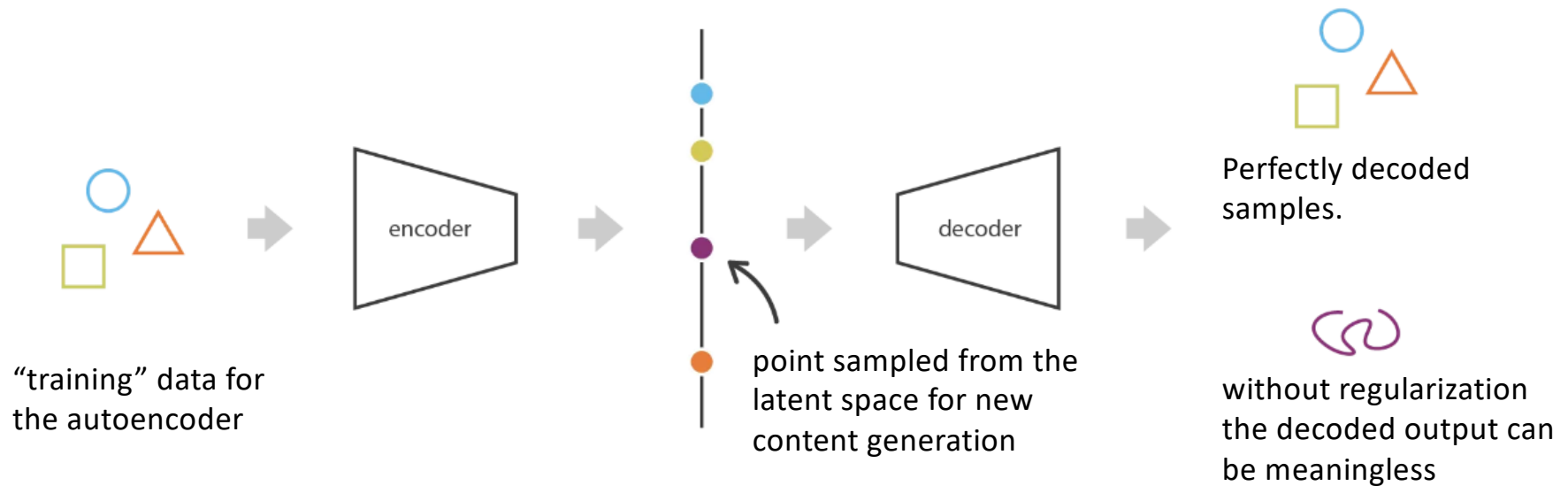
Train encoder and decoder as autoencoder.

Randomly select a different point in the latent space.

Provide as input to the decoder to generate an output.

Will this produce a good quality output?
Why?

Extreme case: Memorization



Encoder and decoder are so powerful that they can fully memorize the data.

Outline

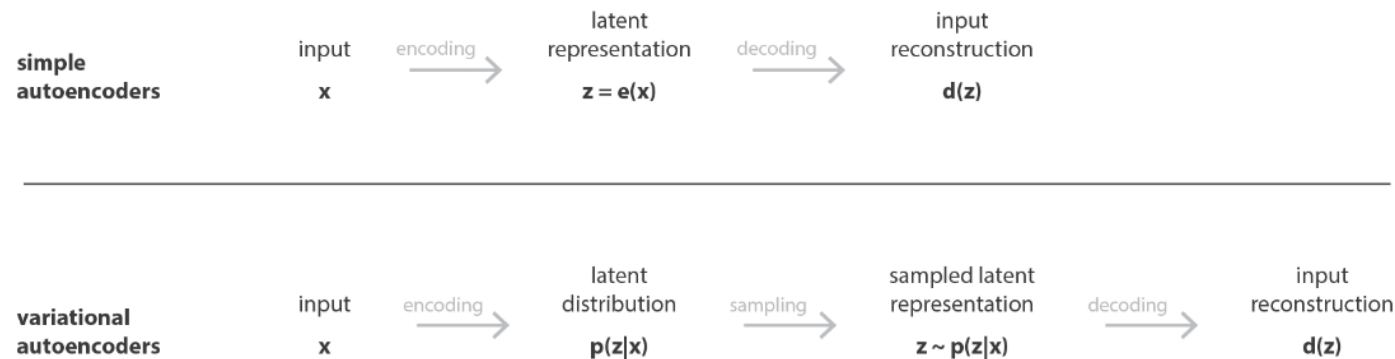
- Autoencoder and its limitations
- Intuition behind VAEs
- Derivation of VAE
- Example applications

Variational Autoencoder...

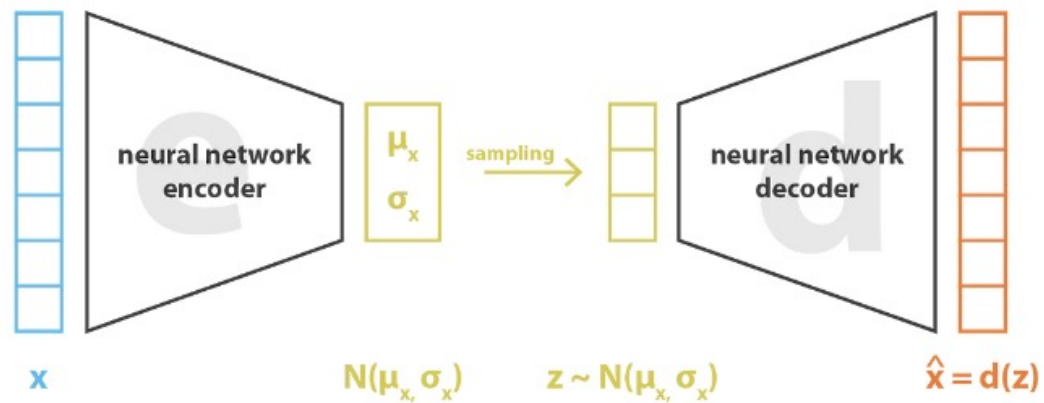
...is an autoencoder whose training is *regularized* to avoid overfitting and ensure that the *latent space has good properties* that enable generative process.

Instead of encoding as a *single point*, encode it as a *distribution* over the latent space.

Assume distributions are normal.

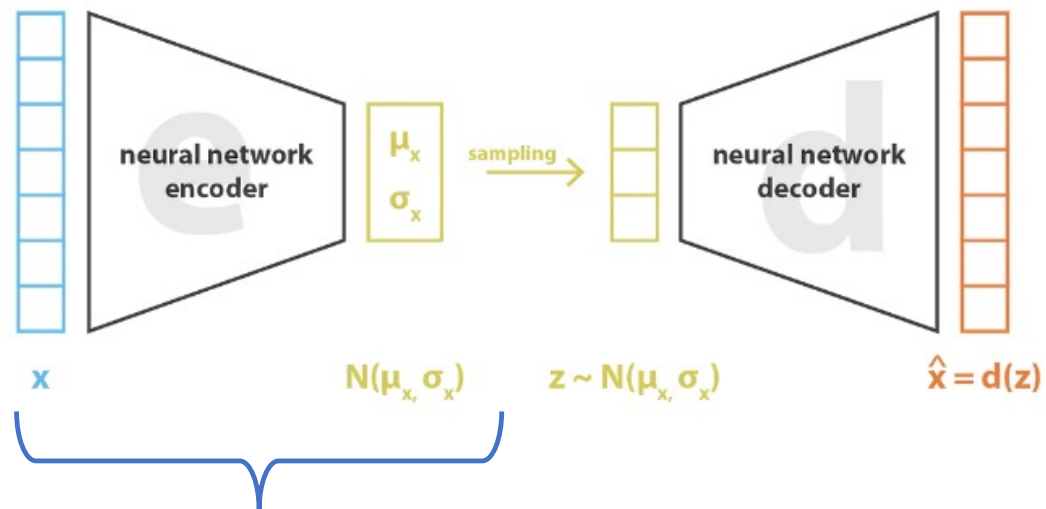


Variational Autoencoder



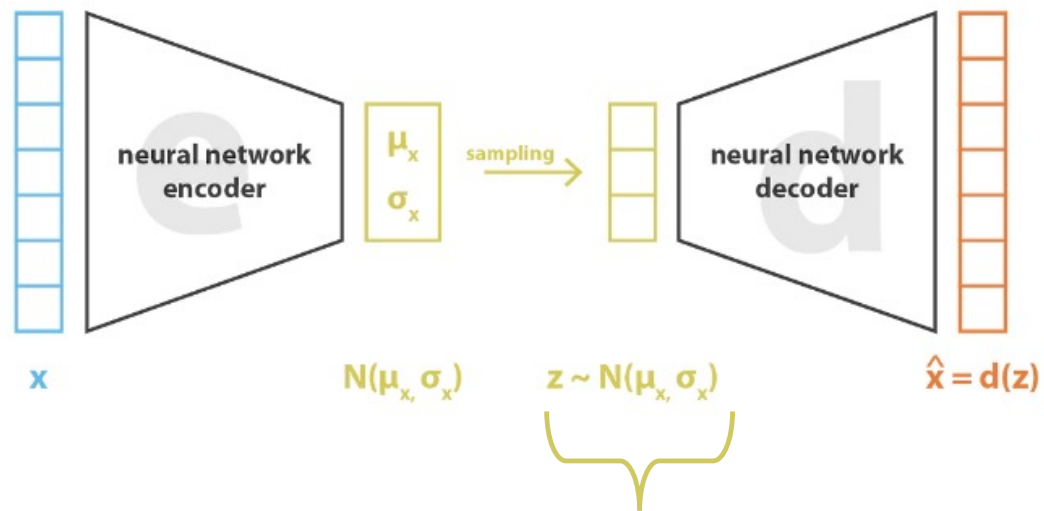
$$\text{loss} = ||x - \hat{x}||^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)] = ||x - d(z)||^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)]$$

Variational Autoencoder

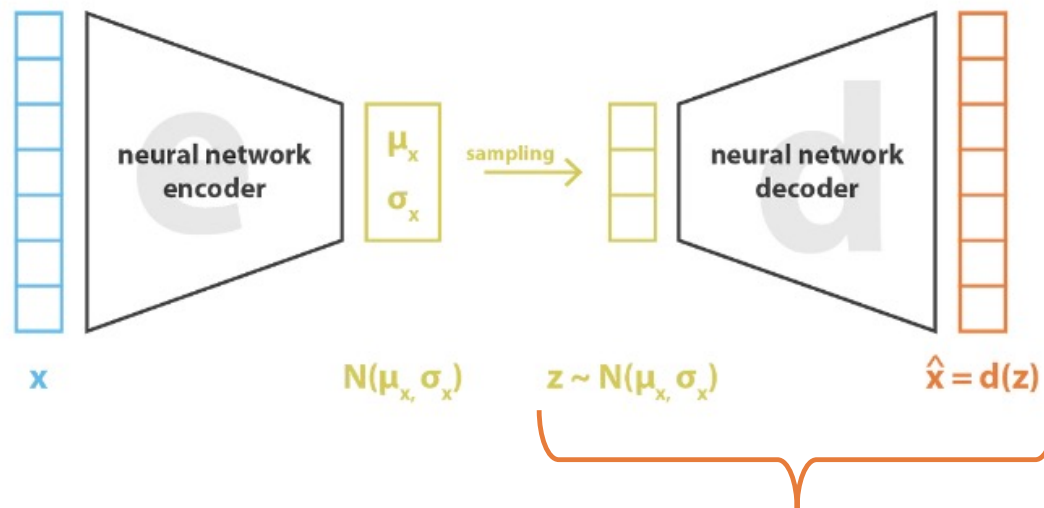


Encoder is emitting μ_x vector and σ_x diagonal vector for independent gaussian densities.

Variational Autoencoder

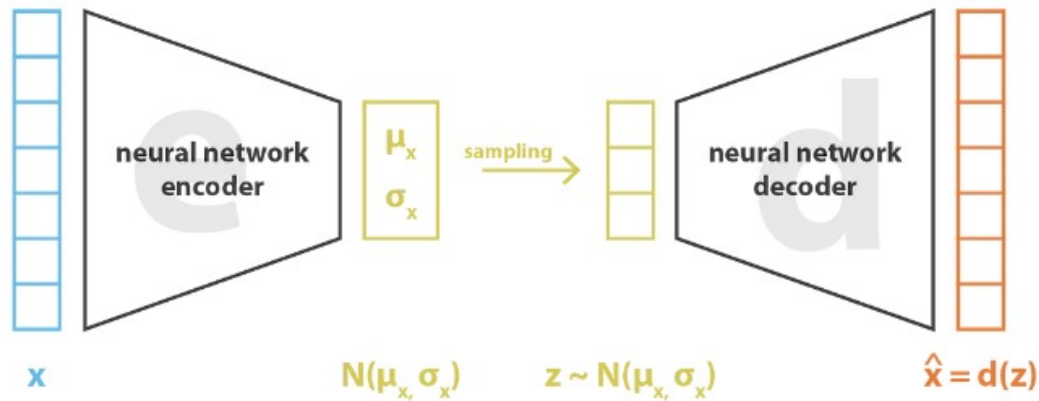


Variational Autoencoder



Then input z to the decoder network to produce output.

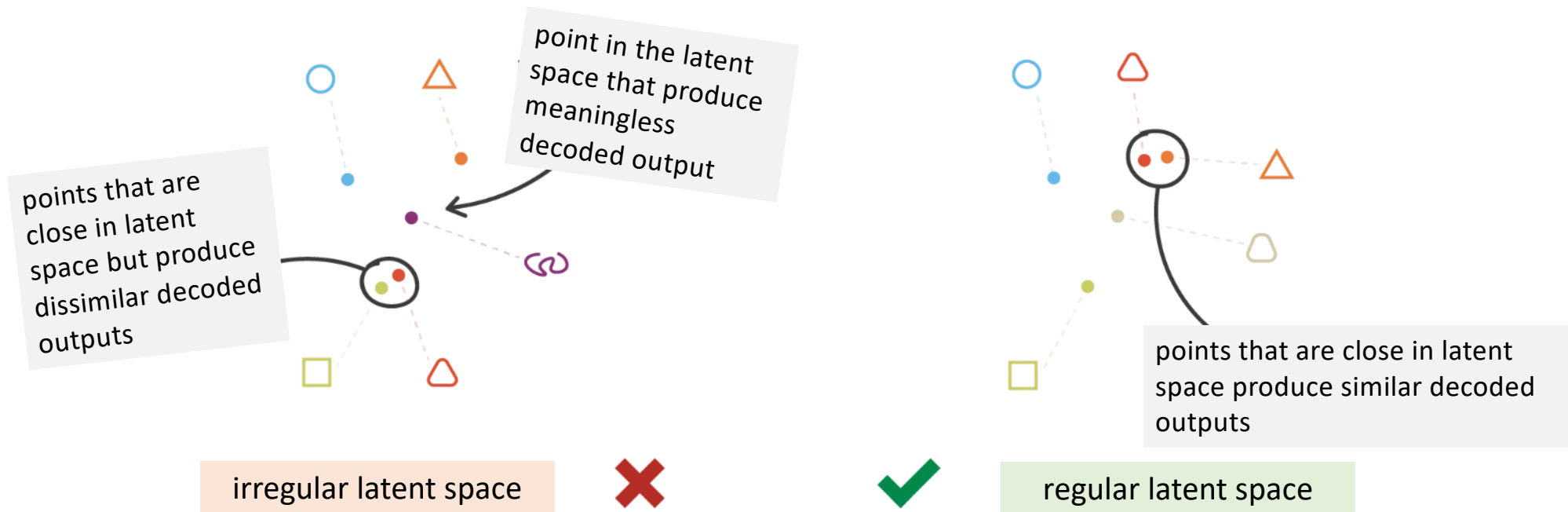
Variational Autoencoder



$$\text{loss} = \underbrace{\|x - \hat{x}\|^2}_{\text{L2 Loss}} + \underbrace{\text{KL}[N(\mu_x, \sigma_x), N(0, I)]}_{\text{Kulback-Leibler divergence}} = \|x - d(z)\|^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)]$$

The loss is now the L2 loss as with the autoencoder, but with an additional KL-divergence term as regularizer.

Intuitions about Regularization



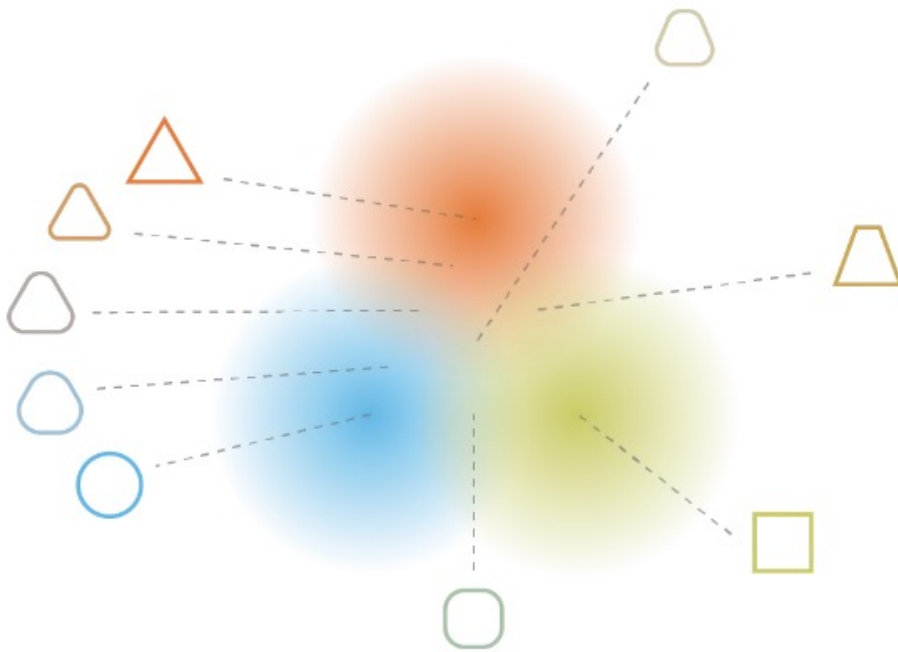
Encoding to Normal distributions is not enough



We have to regularize the means and the covariances too!
Regularize to a standard normal.

$$\text{loss} = ||\mathbf{x} - \hat{\mathbf{x}}||^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)]$$

Benefit of regularization



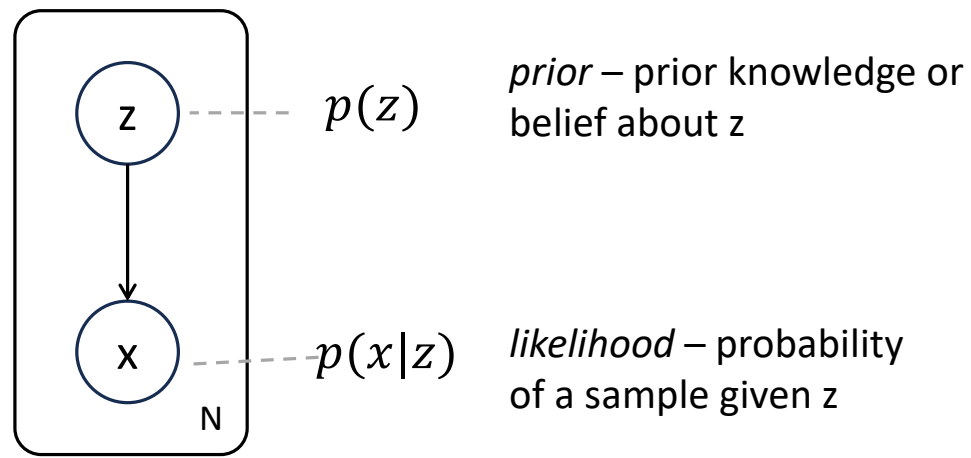
The continuity and completeness obtained from regularization tends to create a “gradient” over the information encoded in latent space.



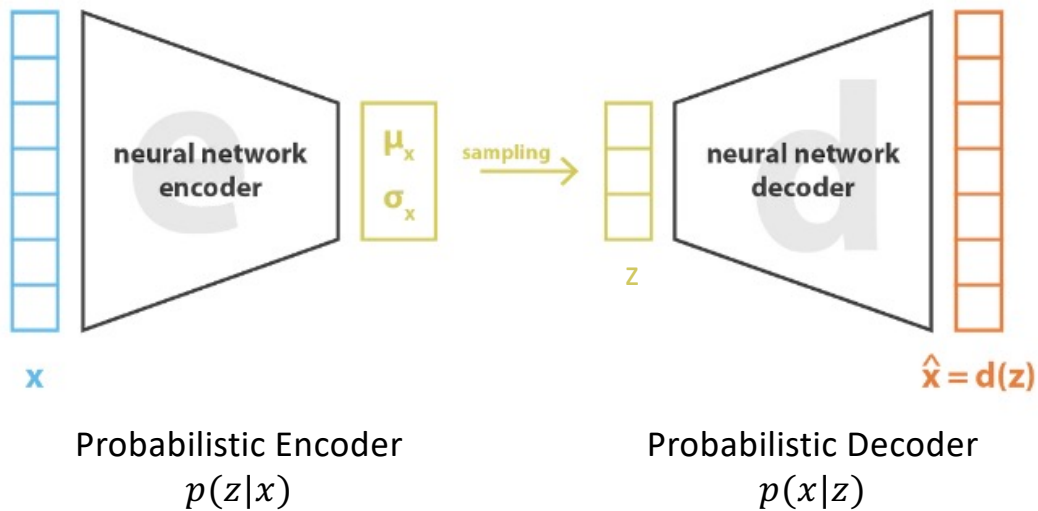
Outline

- Autoencoder and its limitations
- Intuition behind VAEs
- **Derivation of VAE**
- Example applications

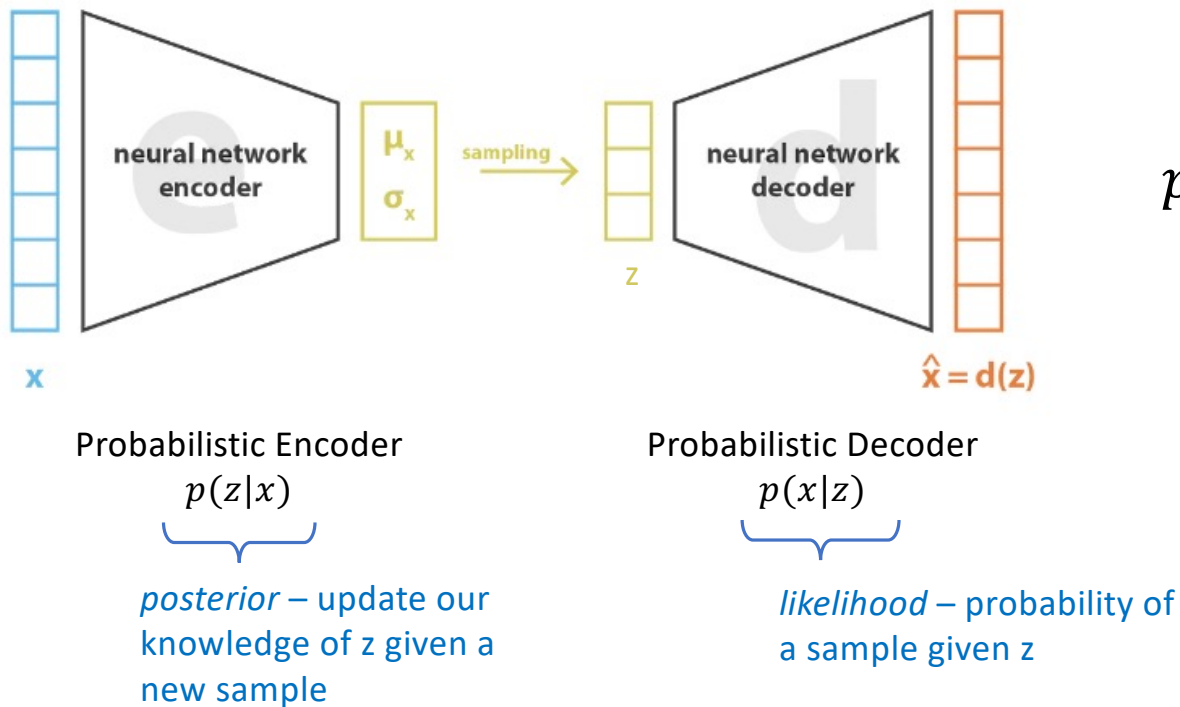
Preliminaries: Bayesian Models



Bayesian Inference



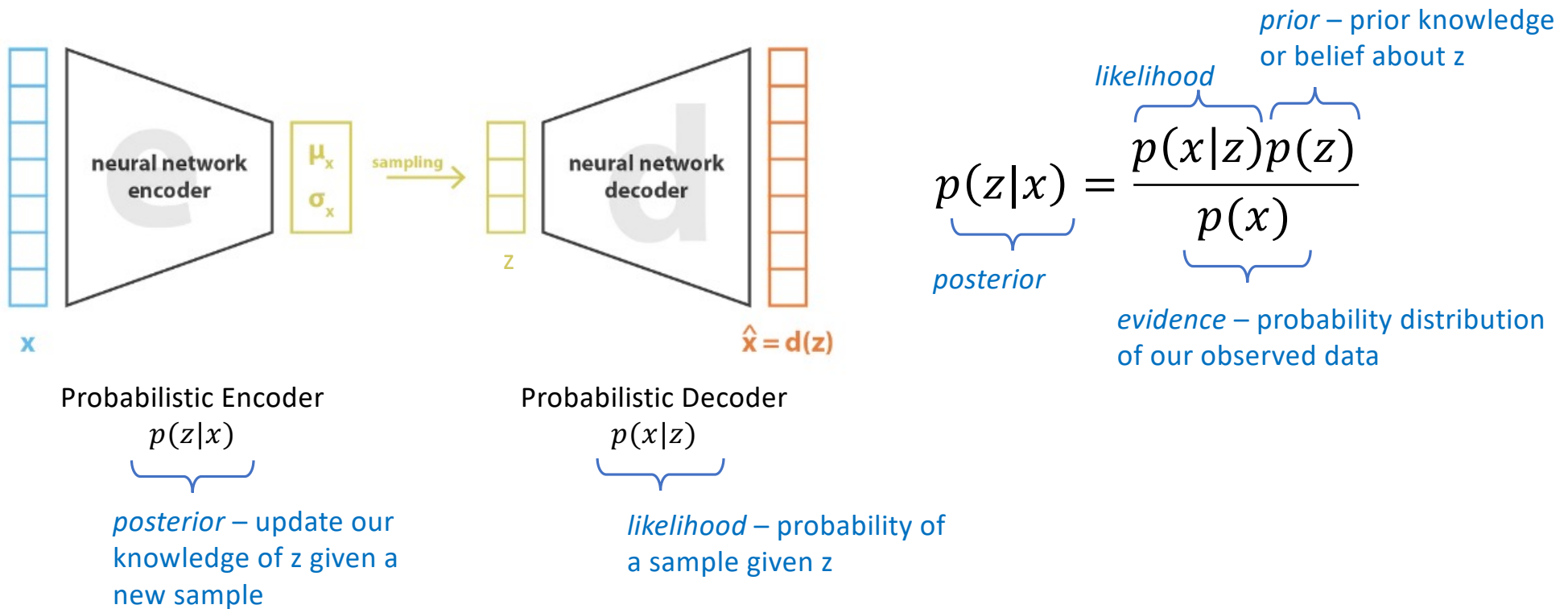
Bayesian Inference



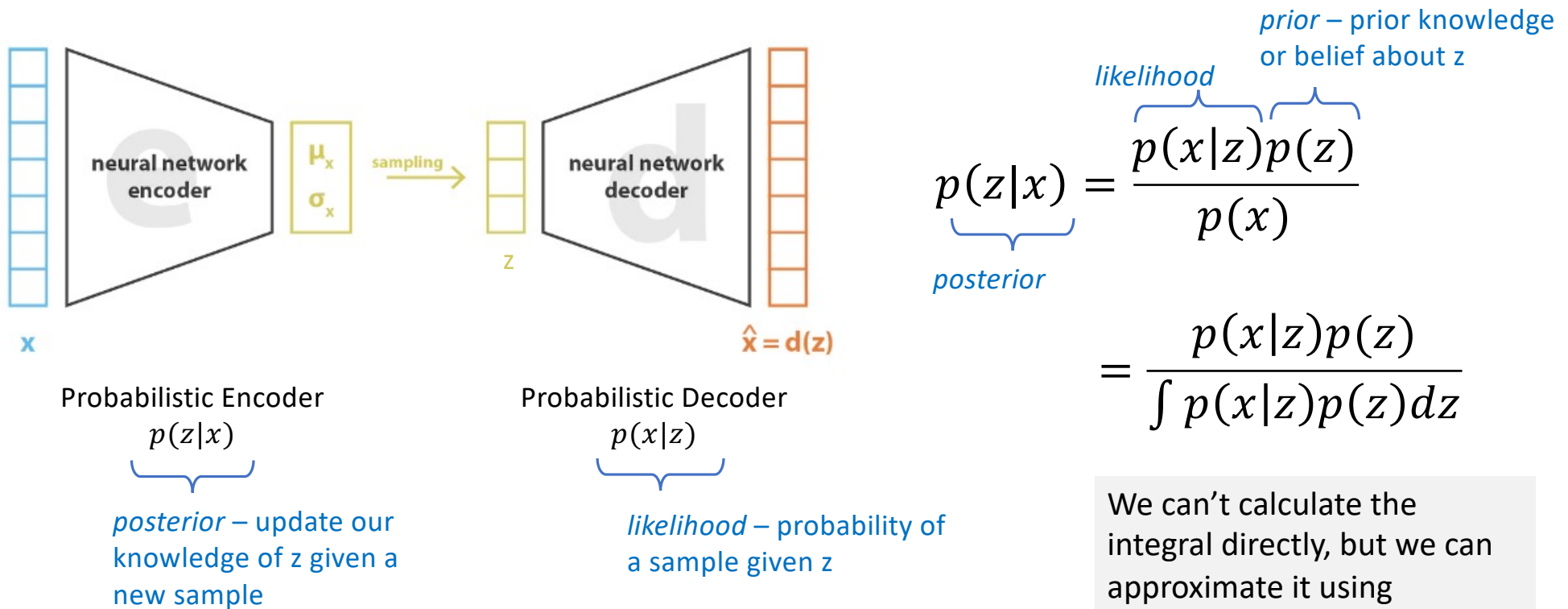
$$p(z|x) = \frac{p(x|z)p(z)}{p(x)}$$

We can relate the *posterior* to the *likelihood* via **Bayes Theorem**.

Bayesian Inference

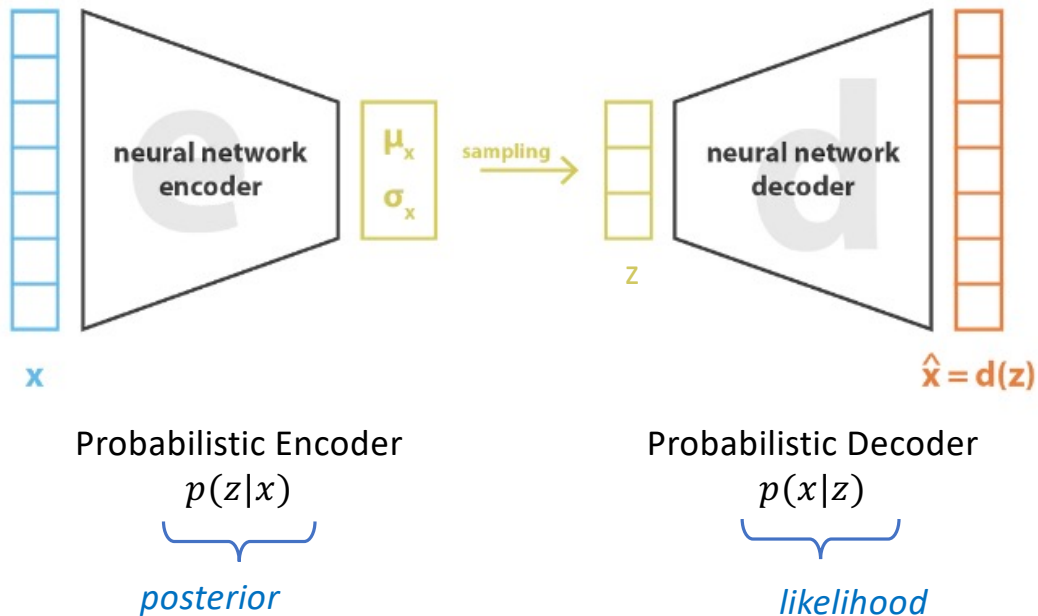


Bayesian Inference



We can't calculate the integral directly, but we can approximate it using *variational inference*

Simplifying Assumptions



Assume that the *prior* is a standard Gaussian

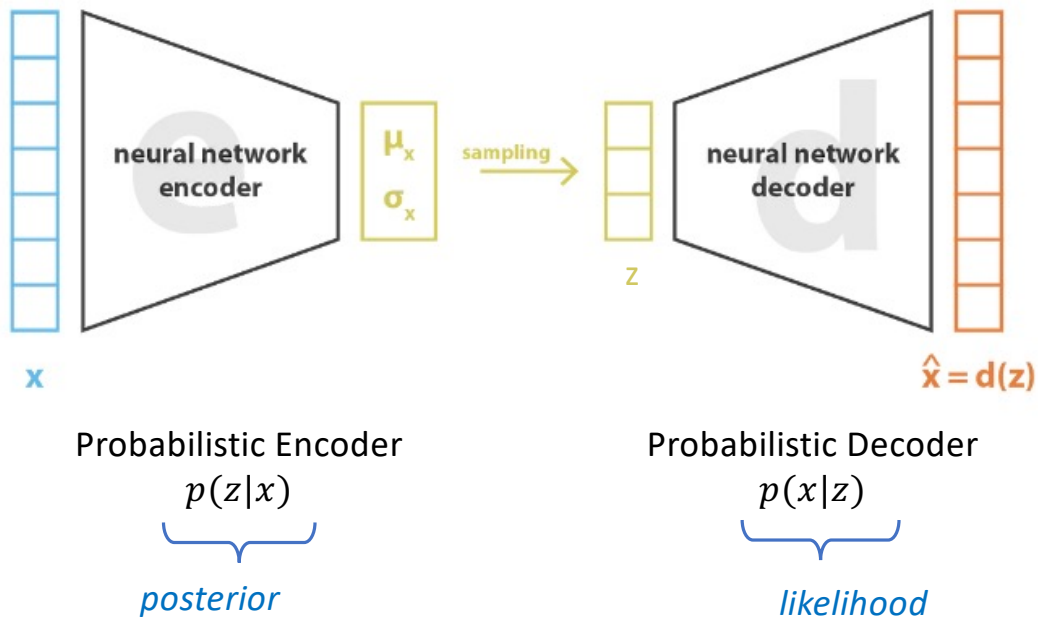
$$p(z) \equiv \mathcal{N}(0, I)$$

And *likelihood* is a Gaussian

$$p(x|z) \equiv \mathcal{N}(f(z), cI)$$

where $f \in F$ is a family of functions we will specify later and $c > 0$.

Variational Inference Formulation



We are going to approximate *posterior* to parameterized set of Gaussians.

Approximate $p(z|x)$ by a Gaussian $q_x(z)$.

$$q_x(z) \equiv \mathcal{N}(g(x), h(x))$$

where $g \in G$ and $h \in H$ are a family of functions we will define shortly.

$$q_x(z) \equiv \mathcal{N}(g(x), h(x))$$

Variational Inference

$$(g^*, h^*) = \arg \min_{(g, h) \in G \times H} KL(q_x(z), p(z|x))$$

We want to find the best functions, g and h , to minimize the KL-divergence from the posterior $p(z|x)$.

C.5.1 Kullback-Leibler divergence

The most common measure of distance between probability distributions $p(x)$ and $q(x)$ is the *Kullback-Leibler* or KL divergence and is defined as:

$$D_{KL}[p(x)||q(x)] = \int p(x) \log \left[\frac{p(x)}{q(x)} \right] dx. \quad (\text{C.28})$$

$$q_x(z) \equiv \mathcal{N}(g(x), h(x))$$

Variational Inference

$$\begin{aligned}(g^*, h^*) &= \arg \min_{(g, h) \in G \times H} KL(q_x(z), p(z|x)) \\ &= \arg \min_{(g, h) \in G \times H} \left(\mathbb{E}_{z \sim q_x} (\log q_x(z)) - \mathbb{E}_{z \sim q_x} \left(\log \frac{p(x|z)p(z)}{p(x)} \right) \right)\end{aligned}$$

- Rewriting KL divergence as Expectation,
- log of division is difference of the logs
- substituting for the posterior using Bayes Theorem

$$q_x(z) \equiv \mathcal{N}(g(x), h(x))$$

Variational Inference

$$\begin{aligned}(g^*, h^*) &= \arg \min_{(g, h) \in G \times H} KL(q_x(z), p(z|x)) \\&= \arg \min_{(g, h) \in G \times H} \left(\mathbb{E}_{z \sim q_x} (\log q_x(z)) - \mathbb{E}_{z \sim q_x} \left(\log \frac{p(x|z)p(z)}{p(x)} \right) \right) \\&= \arg \min_{(g, h) \in G \times H} (\mathbb{E}_{z \sim q_x} (\log q_x(z)) - \mathbb{E}_{z \sim q_x} (\log p(z)) - \mathbb{E}_{z \sim q_x} (\log p(x|z)) + \mathbb{E}_{z \sim q_x} (\log p(x)))\end{aligned}$$

- log of product becomes sum of logs
- log of division becomes difference of logs

$$q_x(z) \equiv \mathcal{N}(g(x), h(x))$$

Variational Inference

$$\begin{aligned}(g^*, h^*) &= \arg \min_{(g, h) \in G \times H} KL(q_x(z), p(z|x)) \\&= \arg \min_{(g, h) \in G \times H} \left(\mathbb{E}_{z \sim q_x} (\log q_x(z)) - \mathbb{E}_{z \sim q_x} \left(\log \frac{p(x|z)p(z)}{p(x)} \right) \right) \\&= \arg \min_{(g, h) \in G \times H} (\mathbb{E}_{z \sim q_x} (\log q_x(z)) - \mathbb{E}_{z \sim q_x} (\log p(z)) - \mathbb{E}_{z \sim q_x} (\log p(x|z)) + \mathbb{E}_{z \sim q_x} (\log p(x))) \\&= \arg \max_{(g, h) \in G \times H} (\mathbb{E}_{z \sim q_x} (\log p(x|z)) - KL(q_x(z), p(z)))\end{aligned}$$

- negating and converting from argmin to argmax
- collecting terms to form KL divergence

$$q_x(z) \equiv \mathcal{N}(g(x), h(x))$$

Variational Inference

$$\begin{aligned}
 (g^*, h^*) &= \arg \min_{(g, h) \in G \times H} KL(q_x(z), p(z|x)) \\
 &= \arg \min_{(g, h) \in G \times H} \left(\mathbb{E}_{z \sim q_x} (\log q_x(z)) - \mathbb{E}_{z \sim q_x} \left(\log \frac{p(x|z)p(z)}{p(x)} \right) \right) \\
 &= \arg \min_{(g, h) \in G \times H} (\mathbb{E}_{z \sim q_x} (\log q_x(z)) - \mathbb{E}_{z \sim q_x} (\log p(z)) - \mathbb{E}_{z \sim q_x} (\log p(x|z)) + \mathbb{E}_{z \sim q_x} (\log p(x))) \\
 &= \arg \max_{(g, h) \in G \times H} \underbrace{(\mathbb{E}_{z \sim q_x} (\log p(x|z)))}_{\text{Maximize the expected log likelihood.}} - \underbrace{KL(q_x(z), p(z))}_{\text{Minimize the difference between the approximate posterior and the prior.}}
 \end{aligned}$$

Maximize the expected log likelihood.

Minimize the difference between the approximate posterior and the prior.

$$q_x(z) \equiv \mathcal{N}(g(x), h(x))$$

Variational Inference

$$\begin{aligned}
 (g^*, h^*) &= \arg \min_{(g, h) \in G \times H} KL(q_x(z), p(z|x)) \\
 &= \arg \min_{(g, h) \in G \times H} \left(\mathbb{E}_{z \sim q_x} (\log q_x(z)) - \mathbb{E}_{z \sim q_x} \left(\log \frac{p(x|z)p(z)}{p(x)} \right) \right) \\
 &= \arg \min_{(g, h) \in G \times H} (\mathbb{E}_{z \sim q_x} (\log q_x(z)) - \mathbb{E}_{z \sim q_x} (\log p(z)) - \mathbb{E}_{z \sim q_x} (\log p(x|z)) + \mathbb{E}_{z \sim q_x} (\log p(x))) \\
 &= \arg \max_{(g, h) \in G \times H} (\mathbb{E}_{z \sim q_x} (\log p(x|z)) - KL(q_x(z), p(z))) \\
 &= \arg \max_{(g, h) \in G \times H} \left(\underbrace{\mathbb{E}_{z \sim q_x} \left(-\frac{\|x - f(z)\|^2}{2c} \right)}_{\text{Log of the Gaussian likelihood } p(x|z) \equiv \mathcal{N}(f(z), cI)} - KL(q_x(z), p(z)) \right)
 \end{aligned}$$

Log of the Gaussian likelihood $p(x|z) \equiv \mathcal{N}(f(z), cI)$.

This brings our function, f , into the equation, so...

$$q_x(z) \equiv \mathcal{N}(g(x), h(x))$$

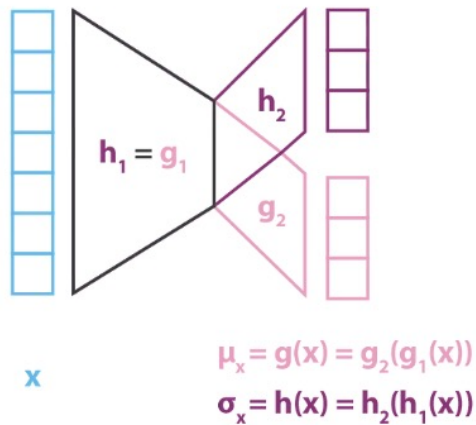
Variational Inference

We are looking for optimal f^* , g^* and h^* such that

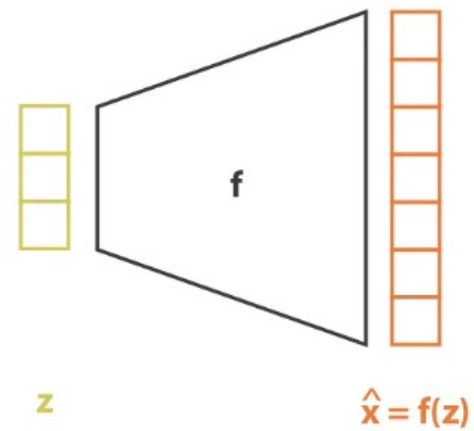
$$(f^*, g^*, h^*) = \arg \max_{(f, g, h) \in F \times G \times H} \left(\mathbb{E}_{z \sim q_x} \left(-\frac{\|x - f(z)\|^2}{2c} \right) - KL(q_x(z), p(z)) \right)$$

Note that the constant, c , determines the balance between reconstruction error and the regularization term given by KL divergence.

Enter the Neural Networks

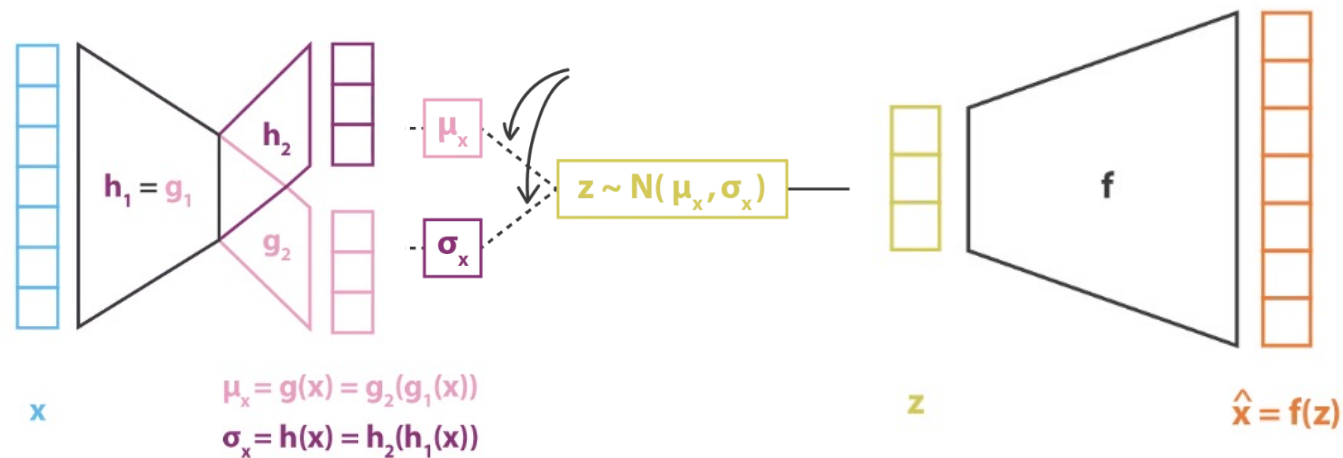


Encoder produces the mean and variance.



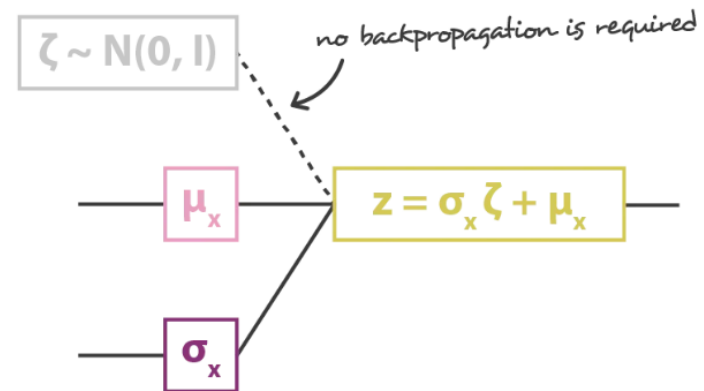
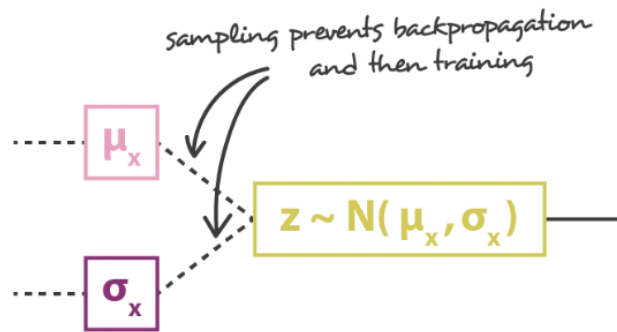
Decoder reconstructs the input (during training)

But one more problem to solve

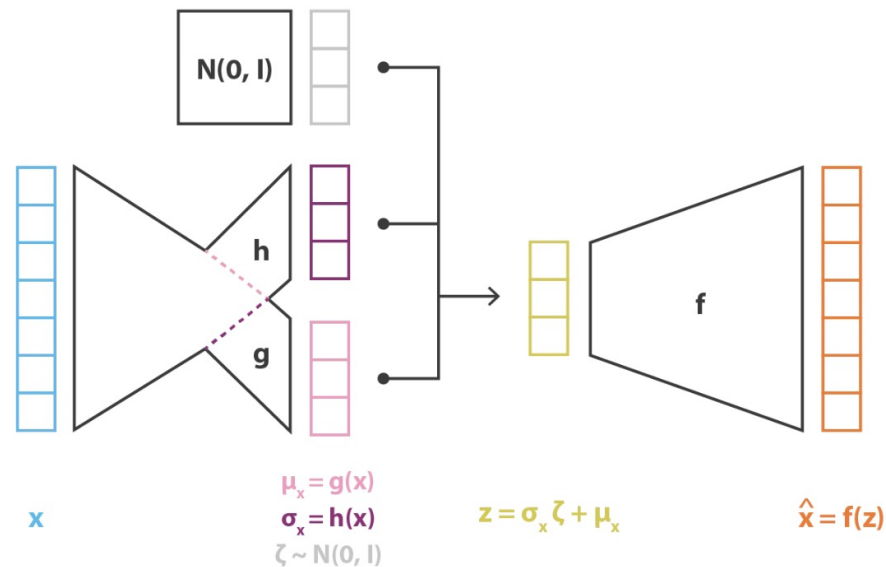


We can't backpropagate through the sampling step.

Use the reparameterization trick



Putting it all together



We use a Monte-Carlo approximation to the expectation of reconstruction loss

Convert $C = 1/(2c)$.

$$\text{loss} = C \|x - \hat{x}\|^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)] = C \|x - f(z)\|^2 + \text{KL}[N(g(x), h(x)), N(0, I)]$$

We have as trainable neural network!

Probability Distribution Divergence Measures

C.5.1 Kullback-Leibler divergence

The most common measure of distance between probability distributions $p(x)$ and $q(x)$ is the *Kullback-Leibler* or KL divergence and is defined as:

$$D_{KL}[p(x)||q(x)] = \int p(x) \log \left[\frac{p(x)}{q(x)} \right] dx. \quad (\text{C.28})$$

C.5.2 Jensen-Shannon divergence

The KL divergence is not symmetric (i.e., $D_{KL}[p(x)||q(x)] \neq D_{KL}[q(x)||p(x)]$). The Jensen-Shannon divergence is a measure of distance that is symmetric by construction:

$$D_{JS}[p(x)||q(x)] = \frac{1}{2} D_{KL} \left[p(x) \middle| \middle| \frac{p(x) + q(x)}{2} \right] + \frac{1}{2} D_{KL} \left[q(x) \middle| \middle| \frac{p(x) + q(x)}{2} \right]. \quad (\text{C.30})$$

It is the mean divergence of $p(x)$ and $q(x)$ to the average of the two distributions.



VAE Code Examples

1. VAE



2. VAE with CIFAR10



https://github.com/DL4DS/sp2025_homeworks/tree/main/lecture_collateral/vae

Outline

- Autoencoder and its limitations
- Intuition behind VAEs
- Derivation of VAE
- Example applications

Generating high quality images



Vahdat & Kautz (2020) “NVAE: A deep hierarchical variational autoencoder”

Resynthesizing real data with changes

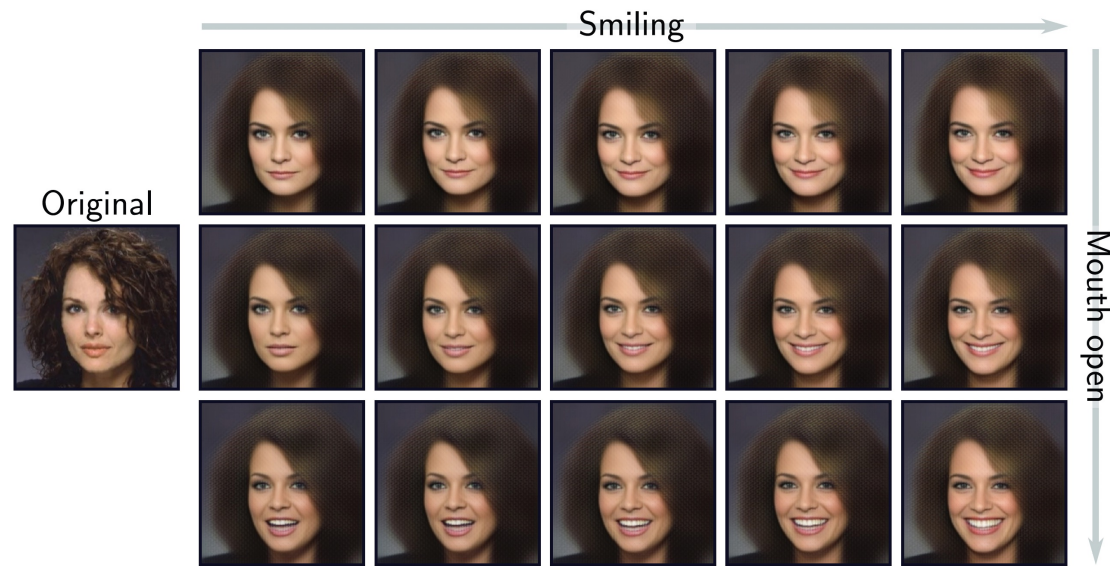
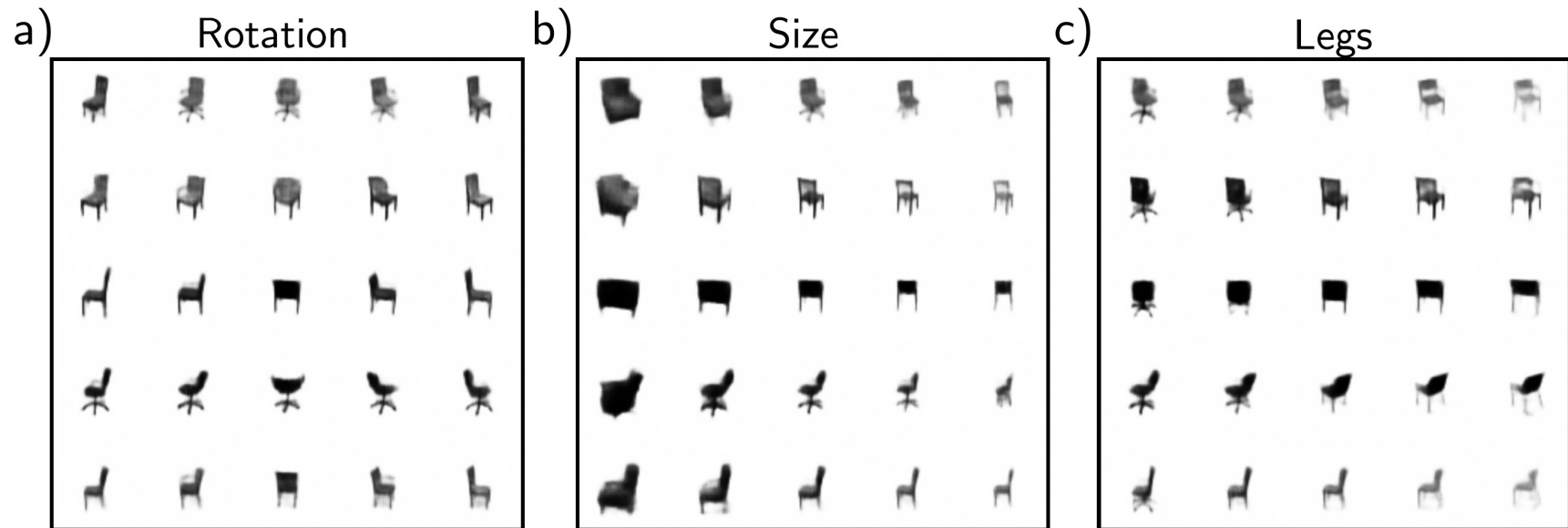


Figure 17.13 Resynthesis. The original image on the left is projected into the latent space using the encoder, and the mean of the predicted Gaussian is chosen to represent the image. The center-left image in the grid is the reconstruction of the input. The other images are reconstructions after manipulating the latent space in directions representing smiling/neutral (horizontal) and mouth open/closed (vertical). Adapted from White (2016).

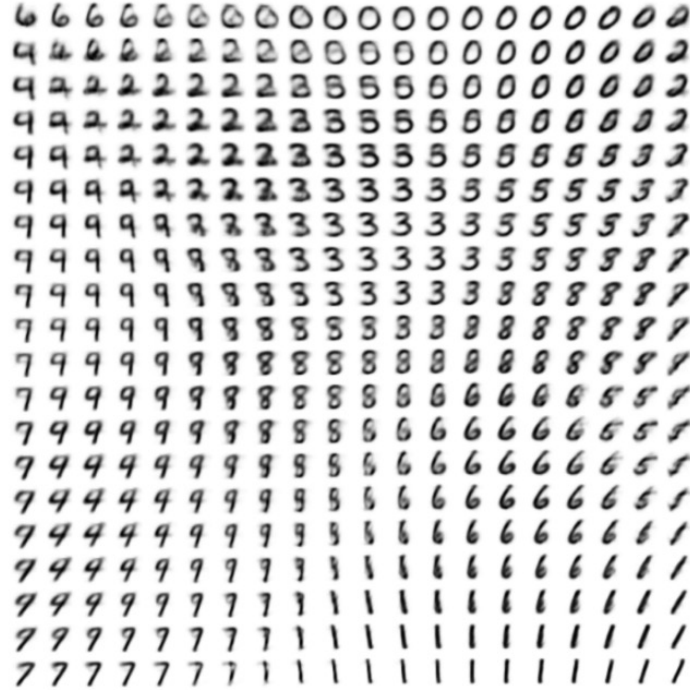
Disentanglement of the latent space



Chen et al (2021) "Cross-layer distillation with semantic calibration."



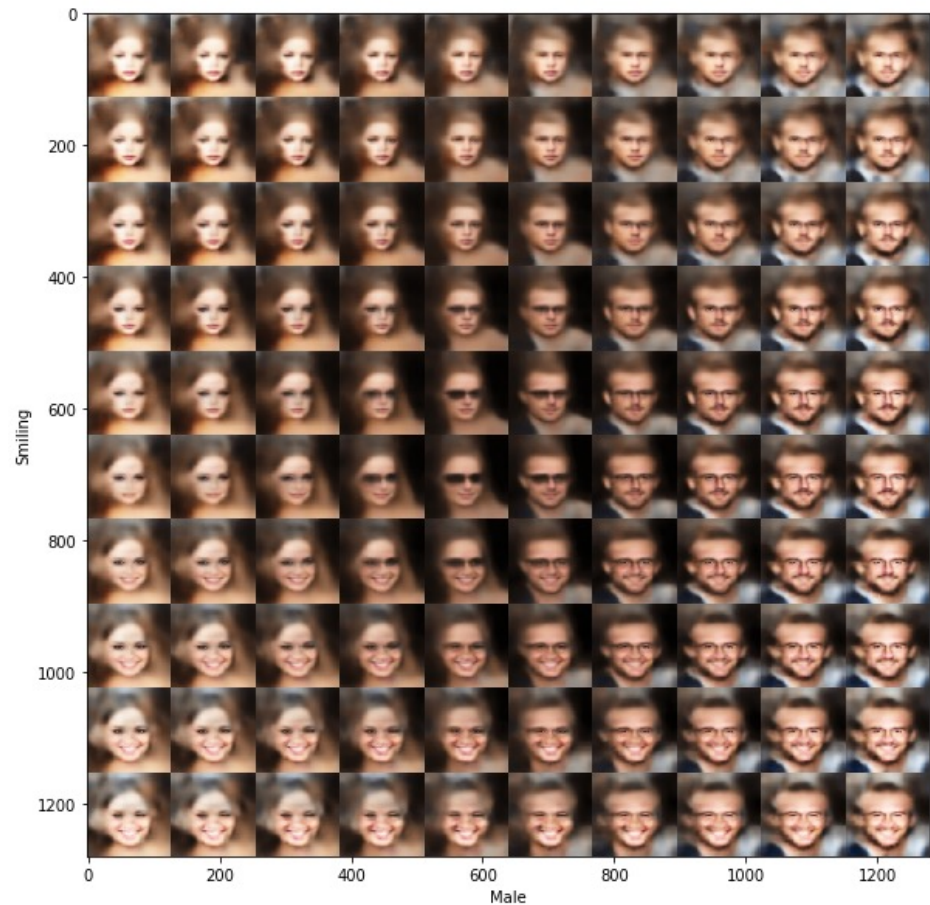
(a) Learned Frey Face manifold



(b) Learned MNIST manifold

Figure 4: Visualisations of learned data manifold for generative models with two-dimensional latent space, learned with AEVB. Since the prior of the latent space is Gaussian, linearly spaced coordinates on the unit square were transformed through the inverse CDF of the Gaussian to produce values of the latent variables $\mathbf{z}$. For each of these values $\mathbf{z}$, we plotted the corresponding generative $p_{\theta}(\mathbf{x}|\mathbf{z})$ with the learned parameters θ .

Conditional VAEs



Example from <https://towardsdatascience.com/variational-autoencoders-vaes-for-dummies-step-by-step-tutorial-69e6d1c9d8e9>

Debiasing

Capable of uncovering **underlying features** in a dataset



Homogeneous skin color, pose

VS



Diverse skin color, pose, illumination

How can we use this information to create fair and representative datasets?

Amini et al, "Uncovering and Mitigating Algorithmic Bias through Learned Latent Structure," 2019

© Alexander Amini and Ava Amini, MIT 6.S191: Introduction to Deep Learning, IntroToDeepLearning.com

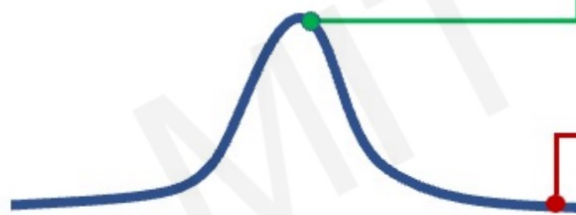
Outlier Detection

- **Problem:** How can we detect when we encounter something new or rare?
- **Strategy:** Leverage generative models, detect outliers in the distribution
- Use outliers during training to improve even more!

95% of Driving Data:
(1) sunny, (2) highway, (3) straight road



Detect outliers to avoid unpredictable behavior when training



Edge Cases



Harsh Weather

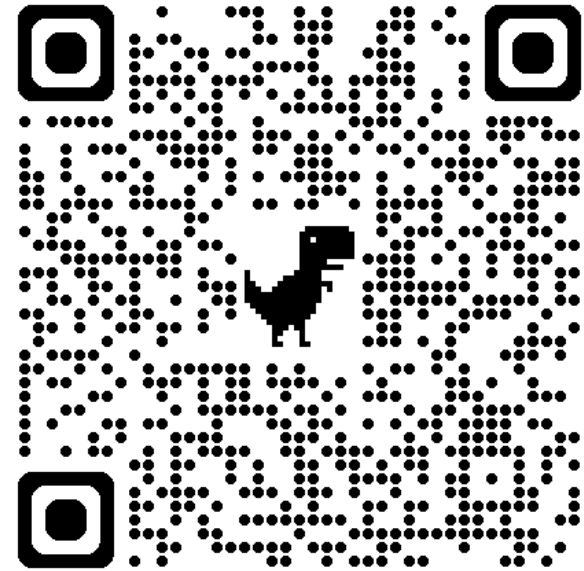


Pedestrians

Upcoming Topics

- Diffusion Models
- Graph Neural Networks

Feedback



[Link](#)