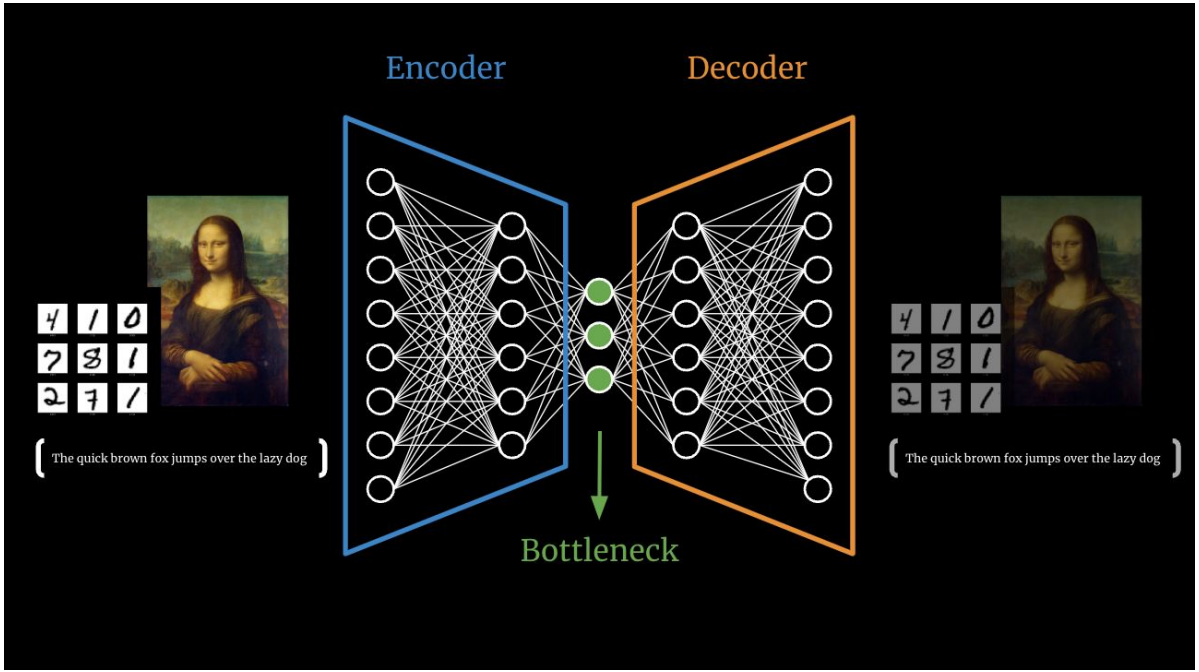
Variational Autoencoders (VAEs)

Arin Ghazarian
Chapman University

Autoencoder

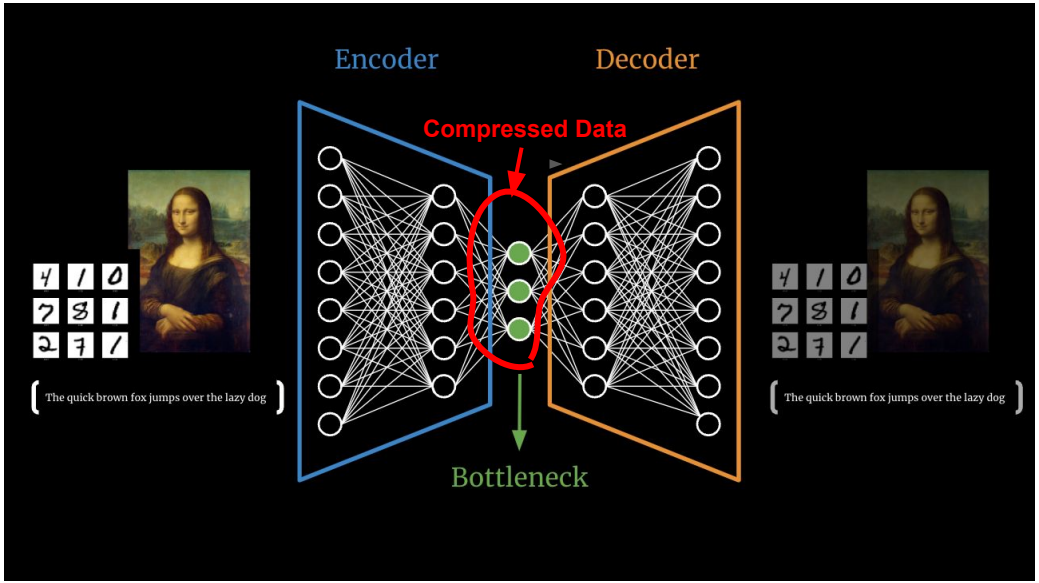
- An autoencoder is a neural network that is trained to attempt to copy its input to its output



https://www.saberhq.com/blog/autoencoders

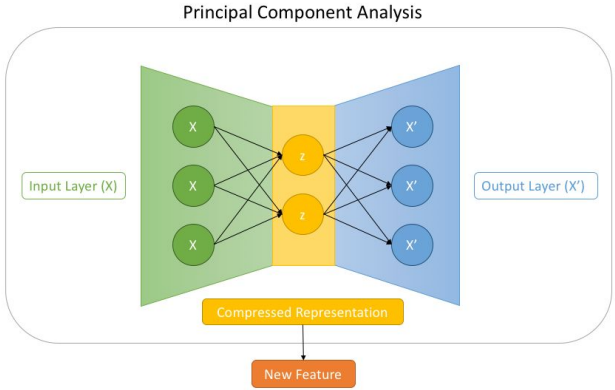
Autoencoders as A Dimensionality Reduction Technique

- VAEs behavior is similar to a dimensionality reduction technique
- For example, AEs can encode a given image using a smaller representative vector



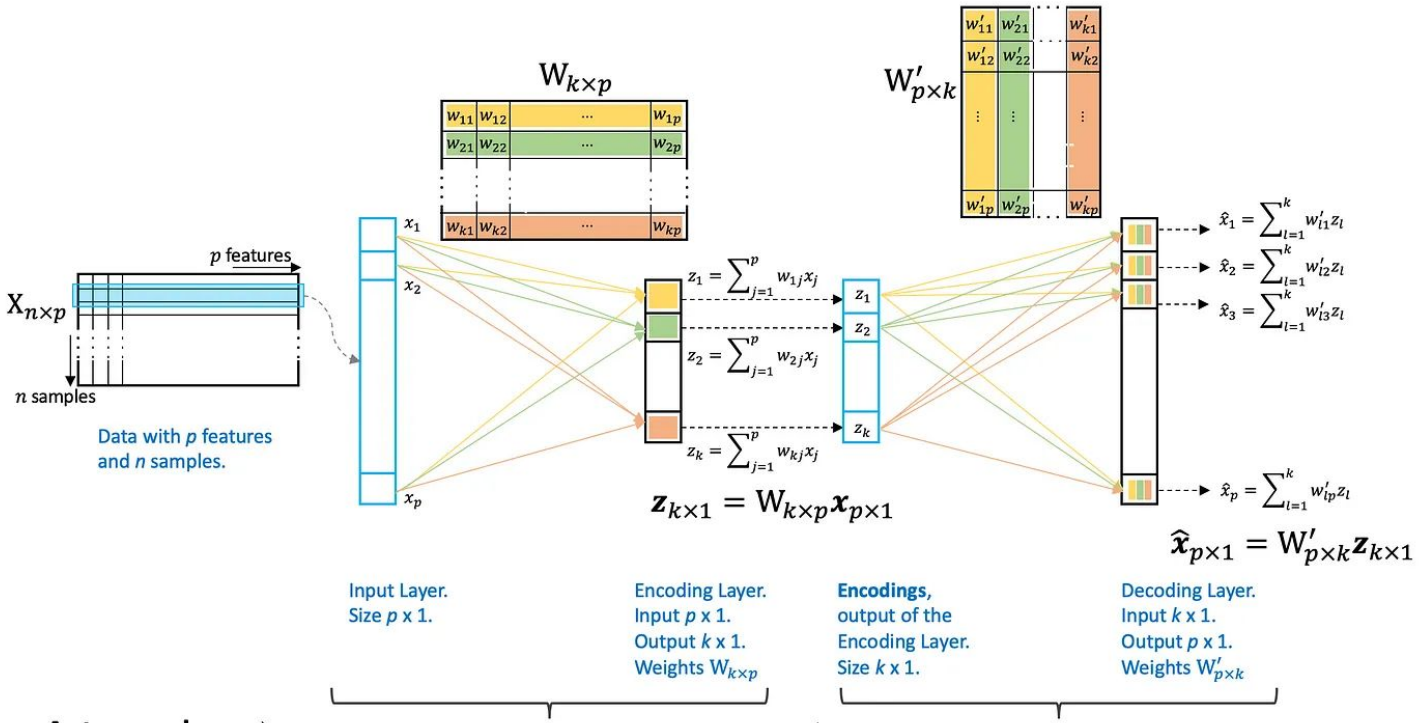
PCA using Autoencoders

- The simplest autoencoder:one hidden layer, linear activations, and squared error loss.
- Linear autoencoders with squared error loss are equivalent to Principal Component Analysis (PCA).
- Two equivalent formulations:
  - Find a subspace that minimizes the reconstruction error
  - Find a subspace that maximizes the projected variance
- Deep nonlinear autoencoders learn to project the data onto a nonlinear manifold (nonlinear dimensionality reduction)



source

PCA using Autoencoders



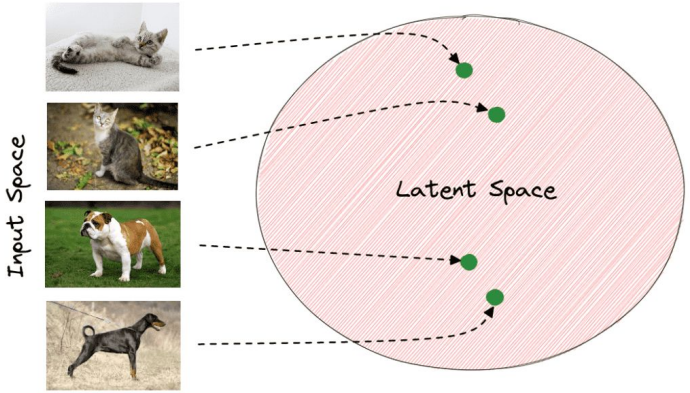
Autoencoder ➡ **Encoding**—converting data to encoded features. ➡ **Decoding**—reconstructing data from encoded features.

PCA ➡ **PC transformation**—converting data to PC scores. ➡ **Reconstruction**—reconstructing data from PC scores.

source

The Problem with AutoEncoders: Holes in the latent space

- If we keep the decoder part, just sample a random point from the encoder latent space and feed to the decoder, we expect to see a valid output
- Data points in our training dataset do not cover the whole latent space in AEs
- If we sample a point from the latent space that belongs to a hole, we won't get a valid output.
- Thus to fix this issue, we have to somehow regularize our latent space, so that the whole manifold of images is mapped to the whole latent space
- VAEs try to solve this problem by using a gaussian distribution for the latent space
    - If we gradually move from one point of latent space to its neighbouring point, then output will also gradually change in a meaningful manner



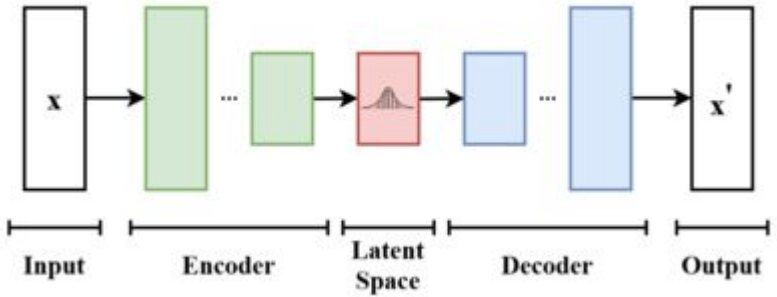
source

Variational Autoencoders (VAEs)

- VAE is another deep learning based method to train generative models
- Since its invention in 2013, VAE has become one of two most popular generative models alongside GANs, specially for producing photorealistic images
- Part of the families of probabilistic graphical models and variational Bayesian methods.
- Auto-Encoding Variational Bayes DP Kingma, M Welling

VAE Architecture

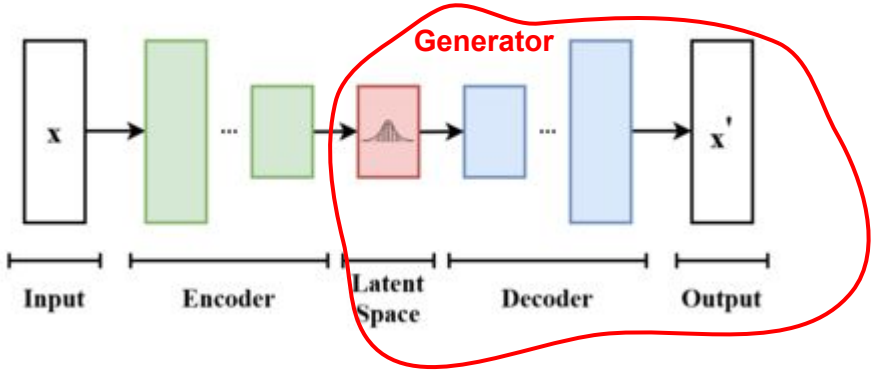
- A VAE model is consisted of an encoder network followed by a decoder network
- We feed the VAE model a sample input, and then at the end compare the encoded-decoded output with the initial input data and backpropagate the error through the architecture to update the weights of the networks.



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

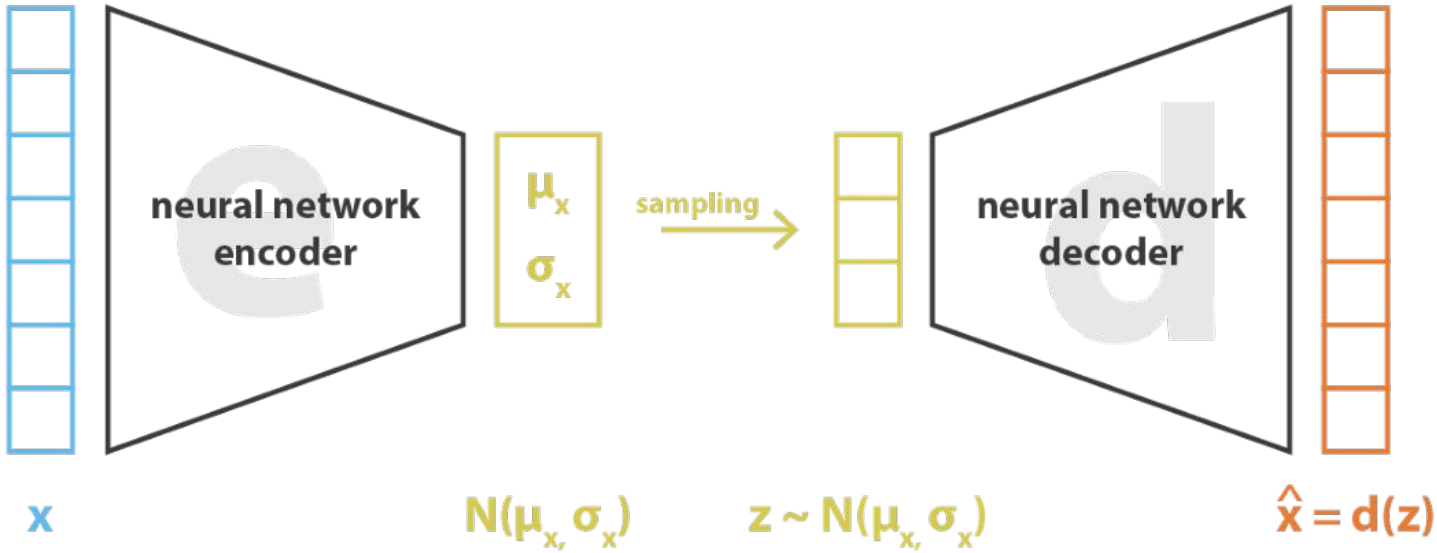
VAEs as Generative Models

- After the training is done, a VAE can be used as a generator model
  - We can take a random point from that encoded vector space and feed it to the decoder network
- The encoded vector space is called the latent space



VAE roots in Information Theory

- Gaussian Distribution in Latent Space
  - In its basic form, the VAE assumes that the latent variables follow a Gaussian distribution. This means the latent space is modeled as a multi-dimensional normal distribution, typically centered around zero with a diagonal covariance matrix (i.e., the dimensions of the latent space are uncorrelated).
- We want to train such an encoder neural network ε_ϕ with parameters ϕ that the Kullback-Liebler divergence between the input image x and its latent representation z is minimized: Min (KL(x,z))

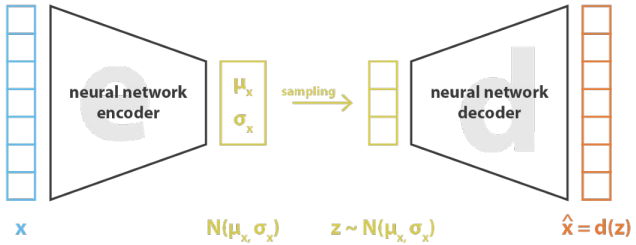


$$\text{loss} = \|x - \hat{x}\|^2 + KL[N(\mu_x, \sigma_x), N(0, I)] = \|x - d(z)\|^2 + KL[N(\mu_x, \sigma_x), N(0, I)]$$

https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

KL Divergence Loss

- To ensure that the learned latent space distribution remains close to a prior distribution (usually a standard normal distribution N(0,I)) VAEs include a KL divergence term in their loss function.
- This term regularizes the latent space and encourages the model to learn meaningful latent variables by preventing the encoder from straying too far from the prior.



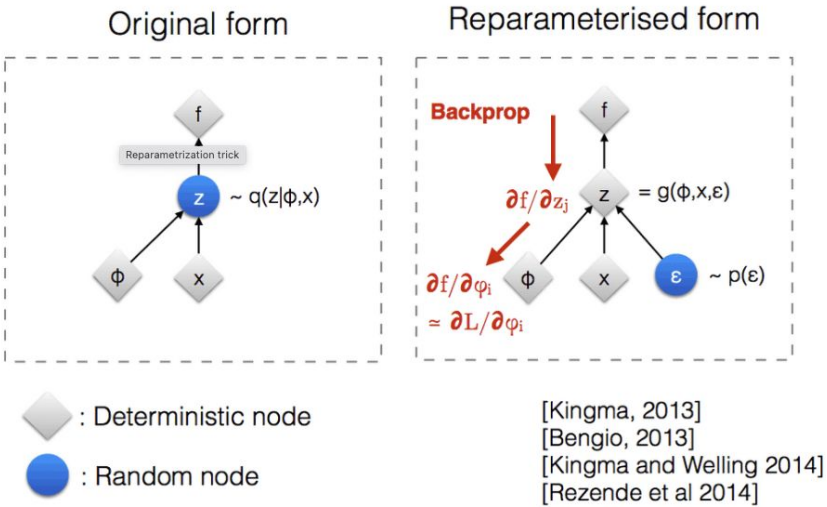
$$\text{loss} = ||x - \hat{x}||^2 + KL[N(\mu_x, \sigma_x), N(0, I)] = ||x - d(z)||^2 + KL[N(\mu_x, \sigma_x), N(0, I)]$$

Dimensionality of the Latent Space

- For small-scale datasets (e.g., MNIST), latent spaces of 2 to 10 dimensions are commonly used.
- For more complex datasets (e.g., CIFAR-10 or CelebA), latent spaces may range from 20 to 100 dimensions.
- For even more complex datasets like high-resolution images, the latent space can be hundreds of dimensions.

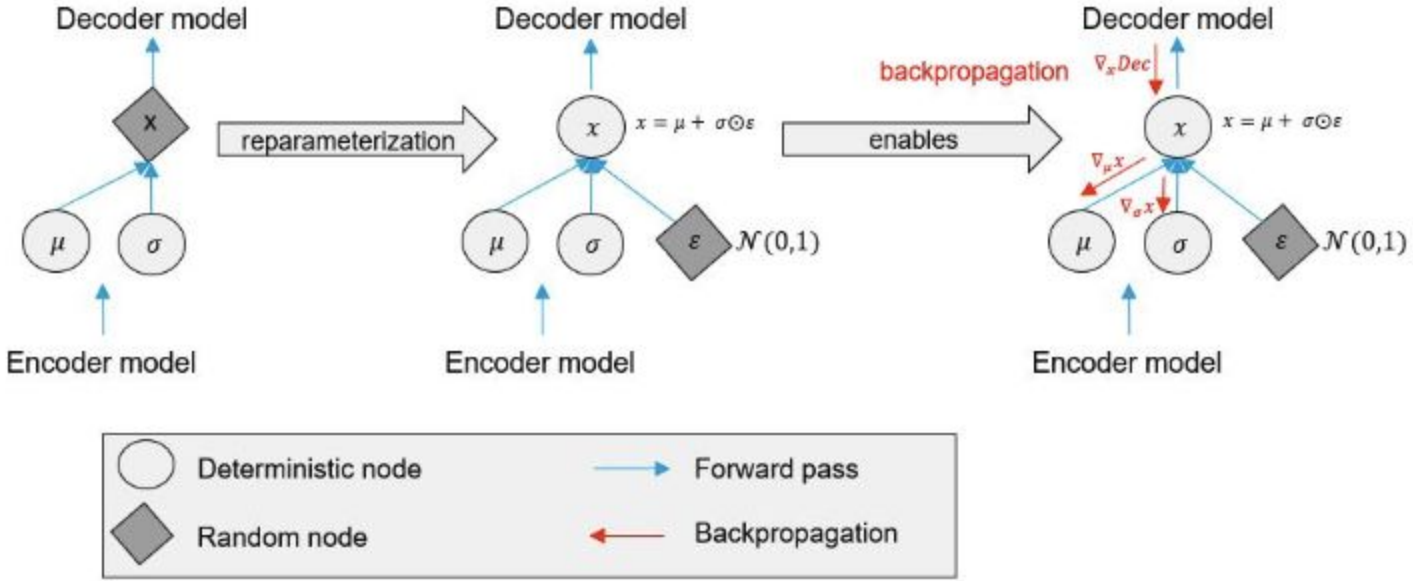
The Reparameterization Trick

- z is a random variable (multivariate gaussian), with mean μ and variance σ which the model aims to learn.
- For each data point in the training batch we sample a random point from that distribution, introducing some extra noise.
- A reparameterization trick is needed to create a differentiable inference model in order to use backpropagation
- In order to keep the network differentiable, we keep distribution mean and variance deterministic, but inject randomness through a random variable called ϵ
    - Otherwise how could we differentiate node Z which generates a random number!?



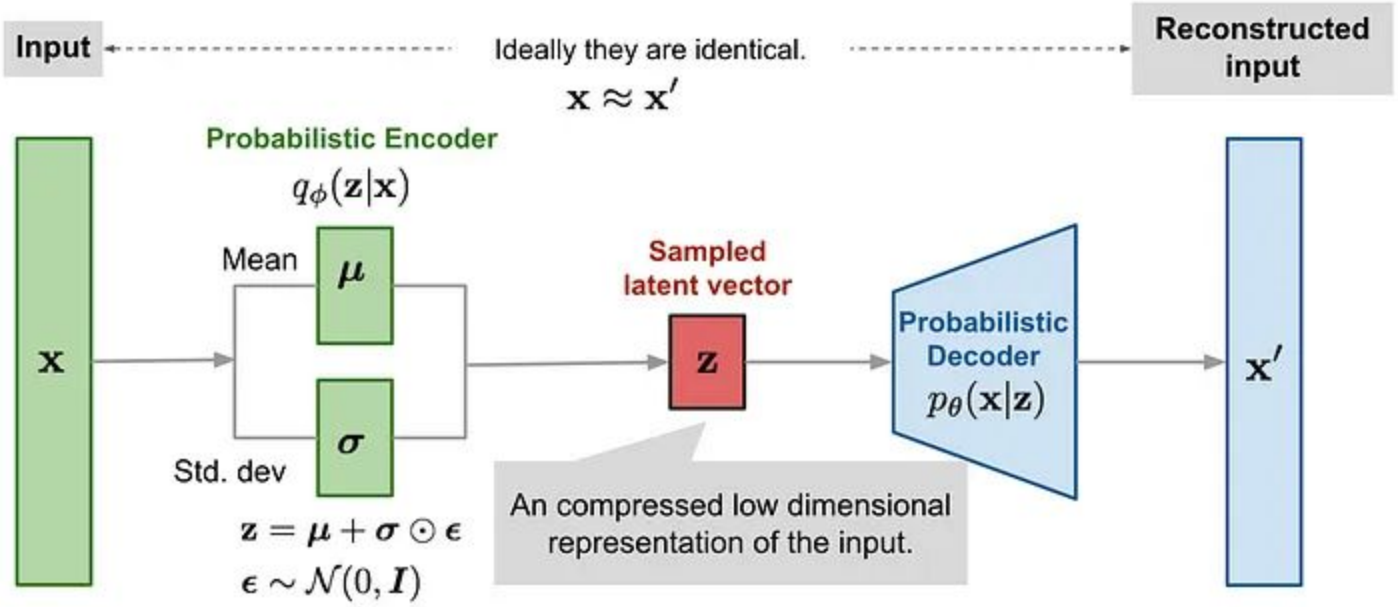
$z = \mu + \sigma * \epsilon$

The Reparameterization Trick



Parameters of the Variational Distribution

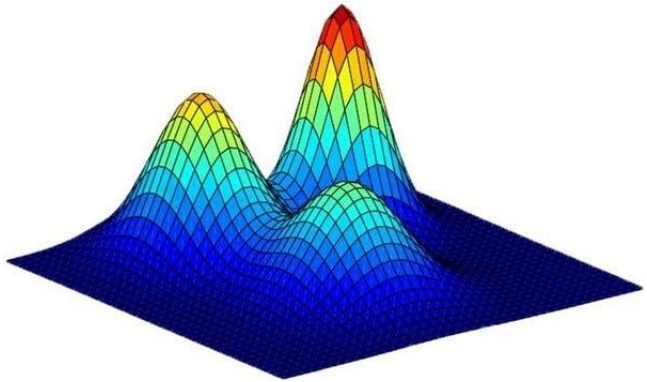
- Instead of directly trying to generate the encoded data with size n, a VAE creates the sample data using probability values with the same size.
- A VAE learns two vectors in the encoder: mean and standard deviation (parameters of the variational distribution)
- Using these two vectors, the decoder can generate a sample point which is then used to decode and transform the data back into the original format.


Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
Mean
μ
σ
Std. dev
$\mathbf{z} = \mu + \sigma \odot \epsilon$
$\epsilon \sim \mathcal{N}(0, I)$
$\mathbf{x}$
Sampled latent vector
$\mathbf{z}$
An compressed low dimensional representation of the input.
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$

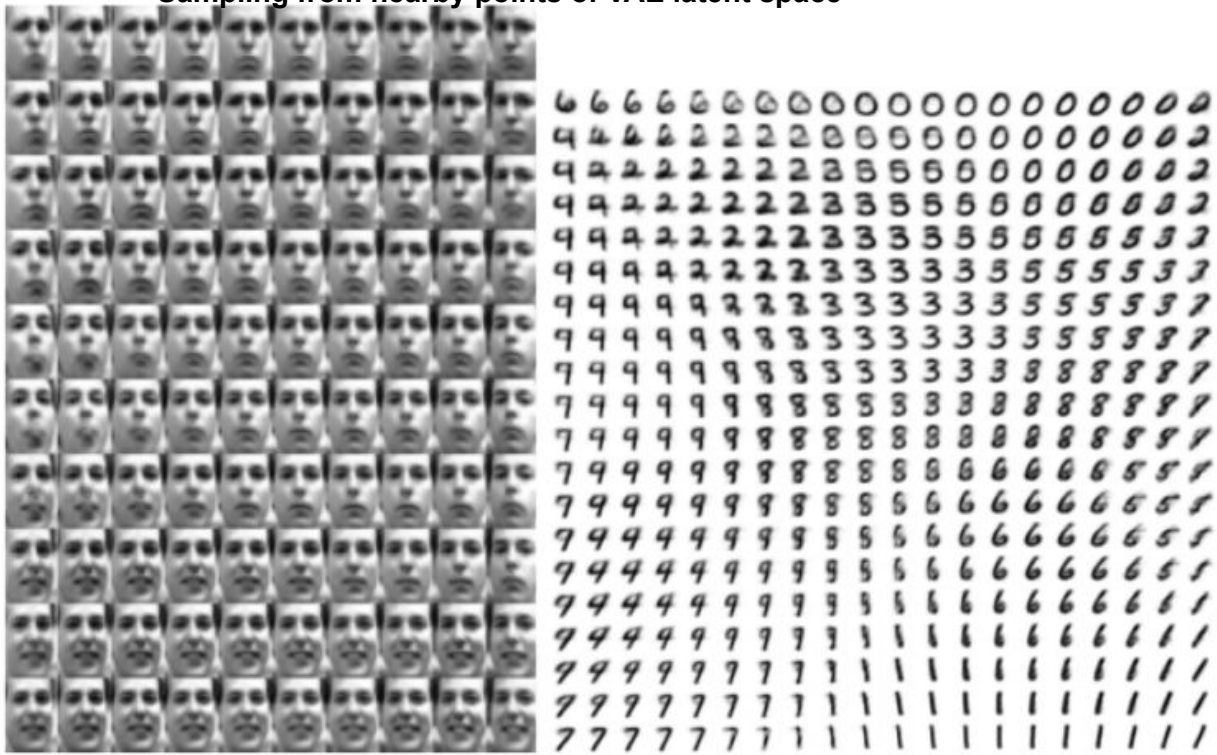
source

Gaussian Mixture VAEs

- Some advanced models, such as Gaussian Mixture VAEs (GMVAEs), incorporate GMMs into the latent space to capture more complex data distributions, but this is not part of the standard VAE framework.
- In fact instead of guessing the distribution of Z directly, we use d normal distributions and then use a combination of those (mixture) to estimate any arbitrary shape z distribution, as the z vector goes through a linear projection in the next layer in the decoder.



Sampling from nearby points of VAE latent space



Examples of 2-D coordinate systems for high-dimensional manifolds, learned by a variational autoencoder (Kingma and Welling, 2014a). Two dimensions may be plotted directly on the page for visualization, so we can gain an understanding of how the model works by training a model with a 2-D latent code, even if we believe the intrinsic dimensionality of the data manifold is much higher. The images shown are not examples from the training set but images x actually generated by the modelp(x | z), simply by changing the 2-D “code”z(each image corresponds to a different choice of “code”zon a2-D uniform grid). (Left) The 2-D map of the Frey faces manifold. One dimension that has been discovered (horizontal) mostly corresponds to a rotation of the face, while the other (vertical) corresponds to the emotional expression. (Right) The 2-D map of the MNIST manifold.

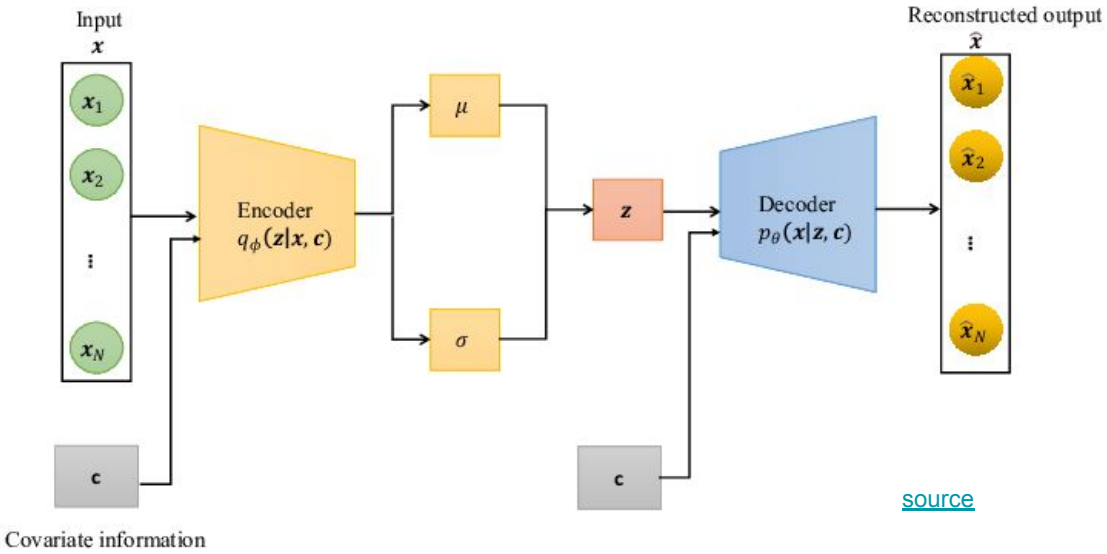
VAE: Pros and Cons

- The variational autoencoder approach is elegant, theoretically pleasing, and simple to implement. It also obtains excellent results and is among the state-of-the-art approaches to generative modeling.
- Its main drawback is that samples from variational autoencoders trained on images tend to be somewhat blurry. The causes of this phenomenon are not yet known.

source

The conditional VAE (CVAE)

- Injects the label information in the latent space to force a deterministic constrained representation of the learned data.



source

https://www.youtube.com/watch?v=9zKuYvjFFS8

https://www.youtube.com/watch?v=fcvYpzHmhvA

Sample Notebooks

- MusicVAE: Creating a palette for musical scores with machine learning
- Variational autoencoder on MNIST
- VAE: Theory and Example
- LATENT SPACE REPRESENTATION: A HANDS-ON TUTORIAL ON AUTOENCODERS USING TENSORFLOW.

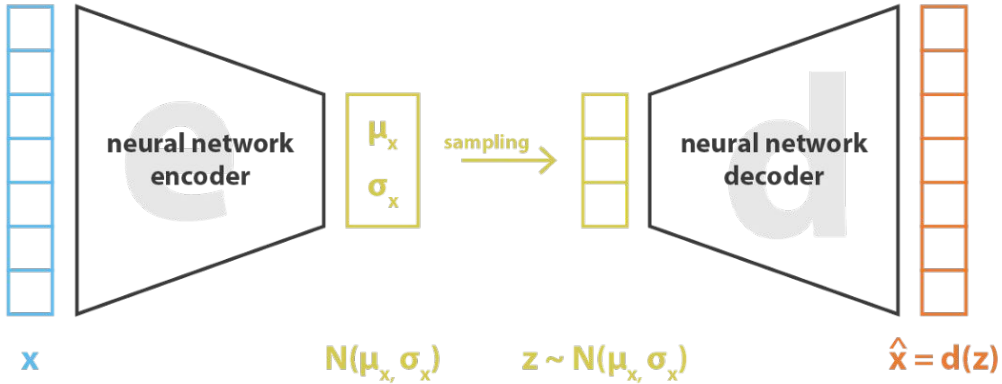
The Mathematics behind Variational AutoEncoders

The mathematics behind Variational AutoEncoders

- http://borisburkov.net/2022-12-31-1/#:~:text=Since%20its%20inception%20in%20the,the%20scope%20of%20this%20post
- https://towardsdatascience.com/an-introduction-to-variational-auto-encoders-vaes-803ddfb623df
- https://en.wikipedia.org/wiki/Variational_autoencoder
- https://www.kaggle.com/code/charel/learn-by-example-variational-autoencoder/notebook
- https://gregorygundersen.com/blog/2021/04/16/variational-inference/

Encoder-Decoder Architecture

- A Variational Autoencoder (VAE) is a generative model that learns to represent data in a latent space and generate new data samples. The VAE consists of two main components:
  a. Encoder: Maps input data x to a latent variable z.
  b. Decoder: Maps the latent variable z back to the data space, generating x^.



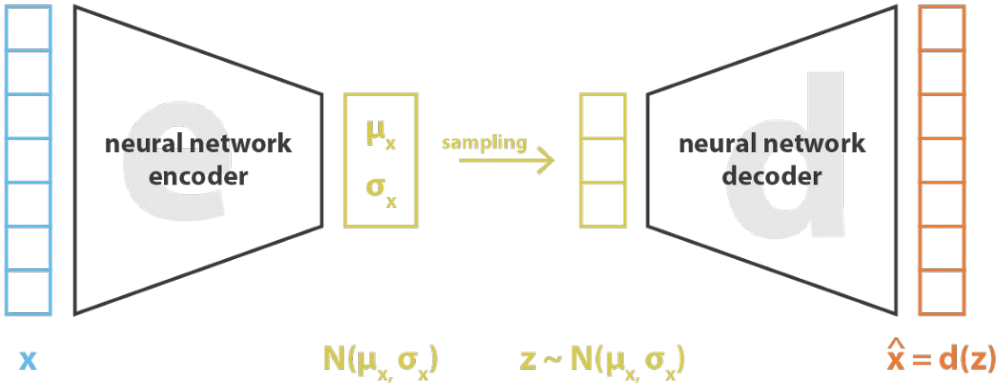
Encoder-Decoder Architecture

- The encoder is defined as $q(z|x)$ and typically parameterized as a Gaussian distribution with mean $\mu(x)$ and variance $\sigma^2(x)$:

$$q(z|x)=N(z;\mu(x),\sigma^2(x)I)$$

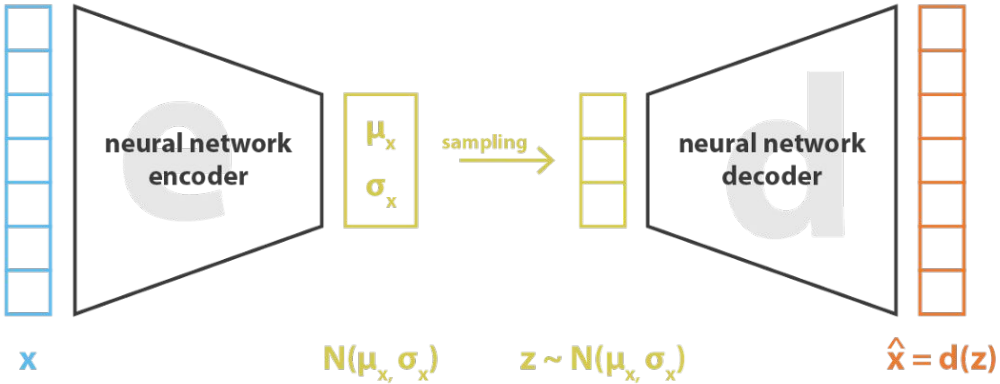
- The decoder defines the likelihood $p(x|z)$, often modeled as a Gaussian distribution:

$$p(x|z)=N(x;\mu\hat{}(z),\sigma\hat{}^2(z)I)$$



Latent Variable Model

- VAEs assume that data x is generated from some latent variable z. The generative process can be described as:
  a. Sample z from a prior distribution p(z), typically a standard normal distribution N(0,I)
  b. Generate x from the conditional distribution p(x | z)

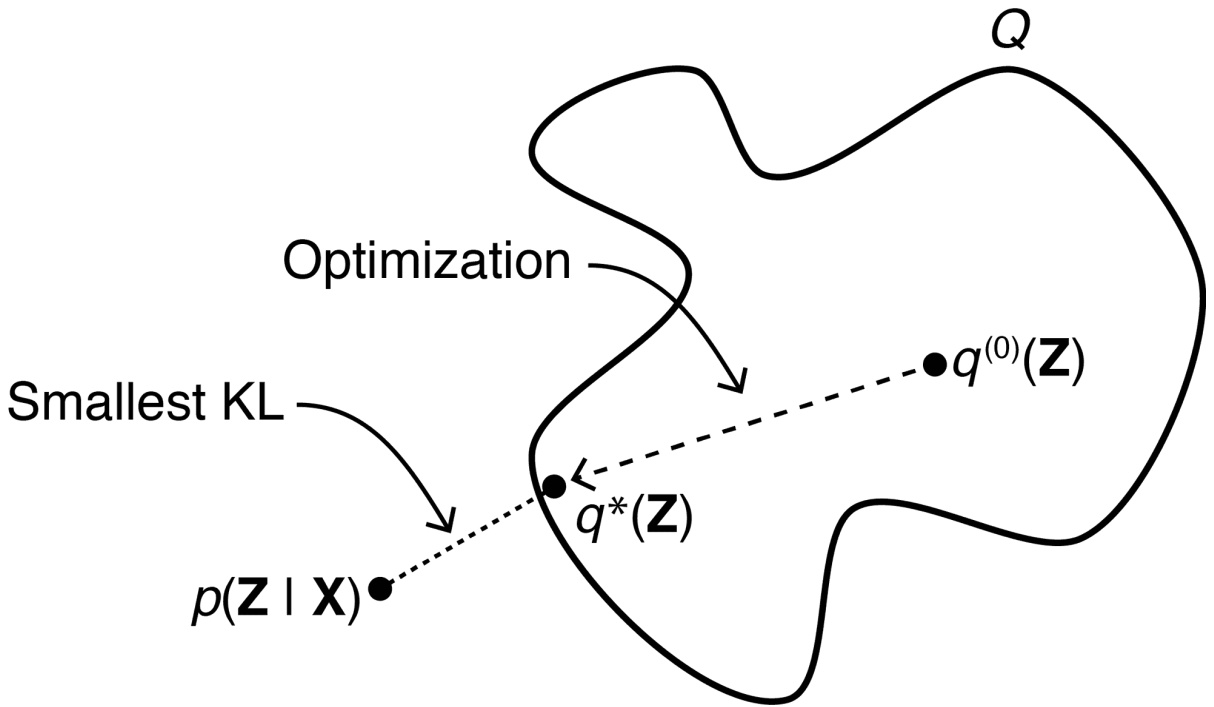


Variational Inference

- $p(x)$, for which the integral $p(x)=\int p(x|z)p(z)dz$ is intractable (i.e. it is not possible to calculate it analytically or computationally in practice).
  - one solution is Markov Chain Monte Carlo methods. In this particular case MCMC estimator is time-consuming and gradient, calculated with it, has a high variance and does not converge.
  - An alternative approach is Variational Inference approach
    - The goal is to learn the posterior distribution $p(z|x)$, but this is often intractable. Instead, VAEs use variational inference to approximate $p(z|x)$ with a simpler distribution $q(z|x)$. This approximation introduces the variational lower bound (Evidence Lower Bound, ELBO).

Variational Inference

- Variational inference aims to approximate the true variational posterior $p(z|x)$ with the best approximation $q^*(z)$ from a certain class of functions
- This optimization process minimizes the Kullback-Liebler divergence between the approximation $q(z)$ and true posterior $p(z|x)$



source

ELBO and Objective Function

The ELBO is derived from the Kullback-Leibler (KL) divergence between the approximate posterior $q(z|x)$ and the true posterior $p(z|x)$:

$$\log p(x) \geq \mathbb{E}_{q(z|x)}[\log p(x|z)] - \mathrm{KL}(q(z|x)||p(z))$$

This can be split into two parts:

1. **Reconstruction Term**: $\mathbb{E}_{q(z|x)}[\log p(x|z)]$
2. **Regularization Term**: $-\mathrm{KL}(q(z|x)||p(z))$

The objective is to maximize the ELBO with respect to the parameters of the encoder and decoder.

Training Objective Function

The final objective function to maximize is:

$$\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \mathrm{KL}(q_\phi(z|x)||p(z))$$

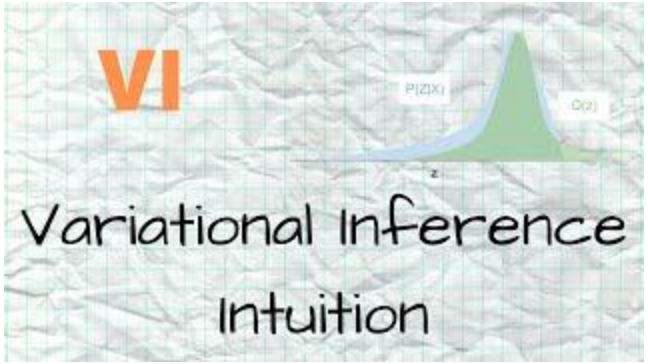
where θ and ϕ are the parameters of the decoder and encoder, respectively.

Practical implementation of loss Function for VAEs

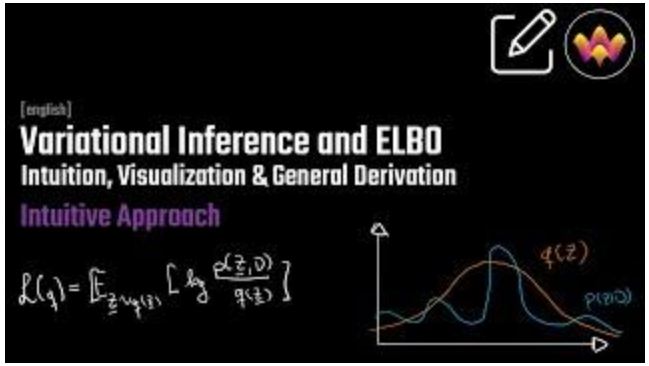
- Reconstruction quality term is either L2 norm of difference between input and reconstructed image in case of Gaussian posterior $p_\theta(x|z)$, or cross-entropy in case of multivariate Bernoulli posterior $p_\theta(x|z)$.

$$\mathcal{L}oss(y_{observation};\theta,\phi) = \underbrace{D_{KL}(q_\phi(x_{hypotheses} \mid y_{observation}) \parallel p_\theta(x_{hypotheses}))}_{KL-loss,\ divergence\ from\ prior} + \underbrace{binary\ crossentropy\ (input\ image, reconstructed\ image)}_{reconstruction\ loss}$$

$$\mathcal{L}oss(y_{observation};\theta,\phi) = \underbrace{-\frac{1}{2}\sum_j (1 + log(\sigma_j^2) - \mu_j^2 - \sigma_j^2)}_{KL-loss,\ divergence\ from\ prior} + \underbrace{binary\ crossentropy\ (input\ image, reconstructed\ image)}_{reconstruction\ loss}$$



https://www.youtube.com/watch?v=A9WmgK9qpm0



https://www.youtube.com/watch?v=HxQ94L8n0vU

https://www.youtube.com/watch?v=rK6bchqeaN8&t=3140s