Learning Distributions with Variational Autoencoders: Theory, Geometry, and Applications

Willy RODRIGUEZ
TORUS AI
SWIM
2025-08-04

Motivation

- Modern machine learning relies heavily on **generative models**.
- **Variational Autoencoders (VAEs)** combine ideas from:
  - Variational inference
  - Neural networks
  - Probabilistic latent variable modeling
- Objective: Learn a distribution $p(x)$ by introducing latent variables.

Goals of This Talk

1. Introduce the **mathematical foundations** of VAEs
2. Introduce optimization function used on VAEs, the **Evidence Lower Bound (ELBO)**
3. Explore the **geometry** of latent spaces
4. Show some applications to **classification tasks**
5. Show some real data applications to **data generation**

Introduction

Before introducing VAEs, let's recall some fundamental concepts in probability and inference.

Introduction

Random Variables and Distributions

A **random variable** X on a sample space Ω is a function $X\colon \Omega \to \mathbb{R}$

A **sample space** Ω is the set of all possible outcomes that one might observe for a given experiment.

Example: Consider the experiment that consists in tossing a coin. The sample space of this experiment is $\Omega = \{H, T\}$

Given a sample space Ω, any real-valued function $X\colon \Omega \to \mathbb{R}$ is called a **random variable on** Ω, and the image of X, denoted here by $\operatorname{range}(X)$, is called the **range** of X:

$$\operatorname{range}(X) := \{X(\omega) : \omega \in \Omega\} \subseteq \mathbb{R}.$$

Introduction

In order to simplify our notation it is convenient, for each $x \in \mathbb{R}$, to let $\{X = x\}$ denote the pre-image of x by X:

$$\{X = x\} := \{\omega \in \Omega : X(\omega) = x\} \subseteq \Omega.$$

For each $x \in \mathbb{R}$, we will denote the probability of the event $\{X = x\}$ by $P(X = x)$.

We also have

$$\sum_{x \in \operatorname{range}(X)} P(X = x) = 1$$

Introduction

Example

For the experiment consisting in rolling a (standard) fair die, we can let $X\colon \{1, 2, \ldots, 6\} \to \mathbb{R}$ be the random variable representing the shown number. Note, for example, that the chances of having the value 2, denoted $P(X = 2)$, are one over six.

We have $P(X = x) = \frac{1}{6}$ for every $x \in \{1, 2, \ldots, 6\}$.

Expected Value

The **expected value** (or mean) of a random variable X is:

Discrete case:

$$\mathbb{E}[X] = \sum_{x \,\in\, \mathrm{range}(X)} x \, P(X = x).$$

Example

For the previous dice example, we can use the formula to compute the expectation of X:

$$\mathbb{E}[X] = \sum_{x \in \{1,2,\ldots,6\}} x P(X = x) = 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + \cdots + 6 \cdot \frac{1}{6} = \frac{21}{6}.$$

Introduction

Continuous Random Variables

A **continuous random variable** is a variable that can take any real value within a given range.

- Unlike discrete variables, it is **not** countable
- Defined using a **probability density function (pdf)** $f(x)$.
- The probability that it takes a specific value is **zero**:

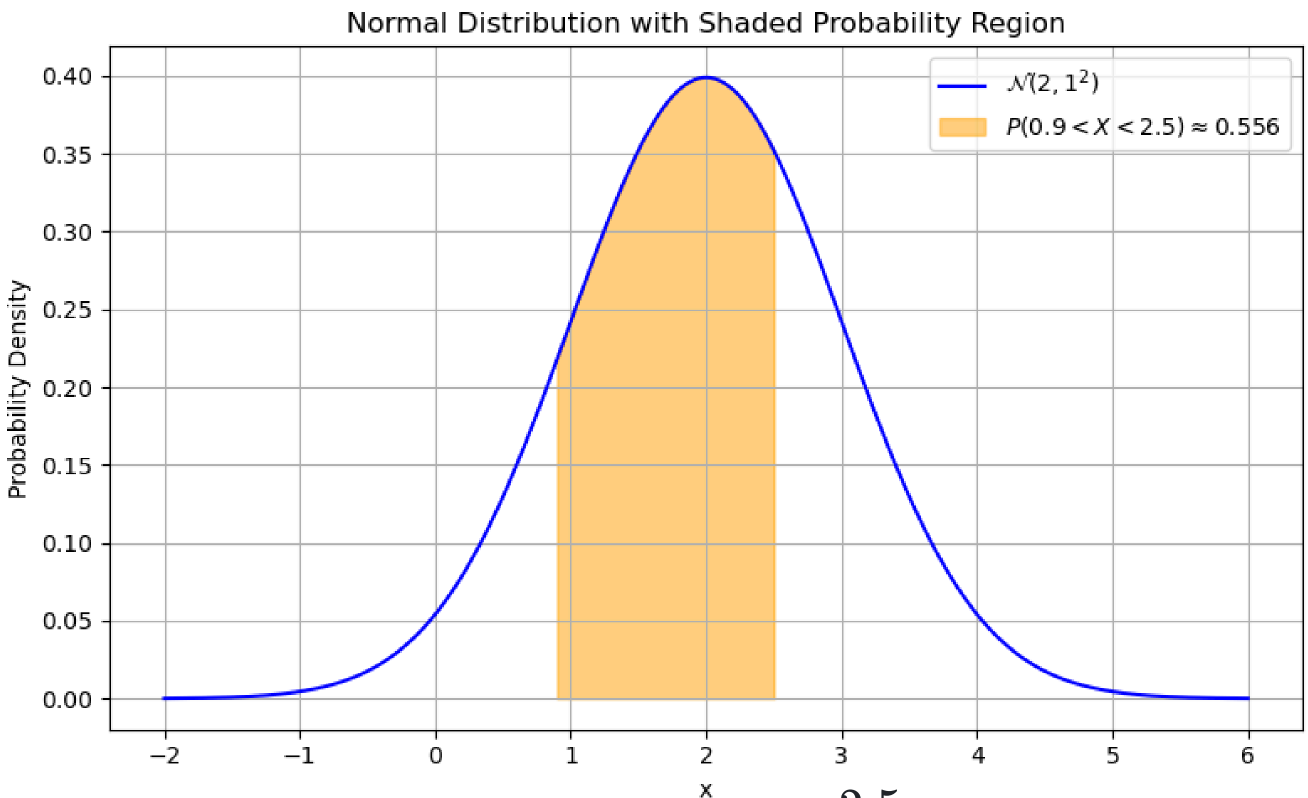
$$\mathbb{P}(X = x) = 0.$$

Instead, we compute probabilities over intervals:

$$\mathbb{P}(a \leq X \leq b) = \int_a^b f(x)\, dx.$$

Introduction

Example: The Normal Distribution



$$\mathbb{P}(0.9 < X < 2.5) = \int_{0.9}^{2.5} f(x)\, dx.$$

Probability Density Function (pdf)

The Normal distribution's pdf, denoted $\mathcal{N}(\mu, \sigma^2)$ is given by:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

- Symmetrical around μ
- μ: Expected value
- σ^2: Variance
- When $\mu = 0$ and $\sigma^2 = 1$, it is called a *standard normal*.

Expected Value

Continuous case:

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} x\, p(x)\, dx$$

Represents the "average" of the distribution.

Introduction

Joint Probability Distribution

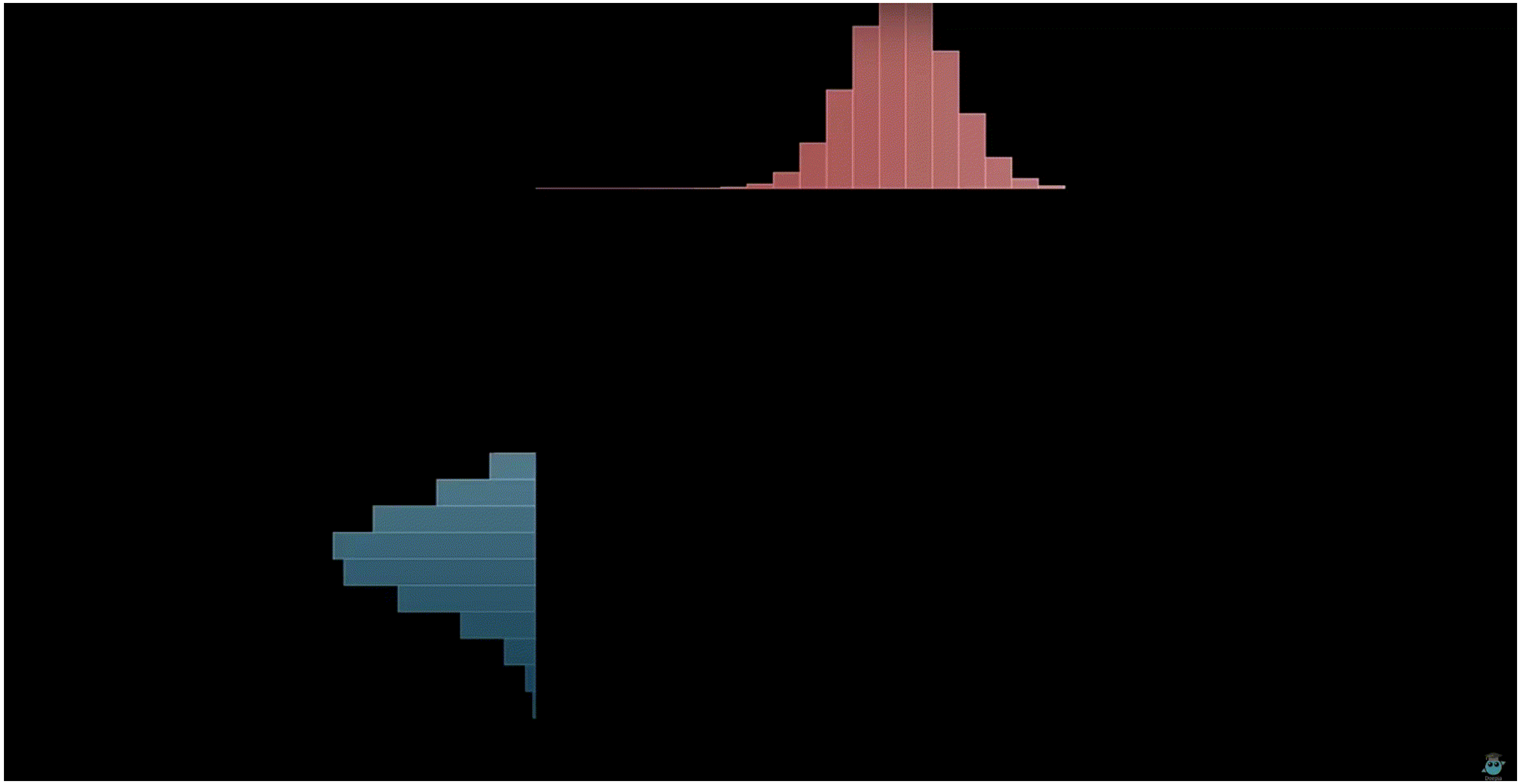


image from: https://www.youtube.com/watch?v=qJeaCHQ1k2w

Joint Probability Distribution

- A **joint distribution** models two (or more) random variables:

$$p(x, z) := P(X = x, Z = z)$$

- It defines the probability that $X = x$ and $Z = z$ simultaneously.

If X and Z are continuous:

$$\int\int p(x, z)\, dx\, dz = 1$$

Marginal Distribution

- The **marginal distribution** of X from $p(x, z)$ is:

$$p(x) = \int p(x, z)\, dz$$

- Integrate (or sum) out the other variable.

Used when we are only interested in X and not Z.

Introduction

Conditional Distribution

- The **conditional distribution** of Z given X is:

$$p(z \mid x) := \frac{p(x,z)}{p(x)} \quad (\text{if } p(x) > 0)$$

- Describes the probability of Z, assuming we know X.

Introduction. Bayes' Rule

Bayes' Rule relates the **posterior**, **likelihood**, **prior**, and **evidence**:

$$p(z \mid x) = \frac{p(x \mid z)\, p(z)}{p(x)}.$$

- $p(z \mid x)$: **Posterior** — what we want to infer
- $p(x \mid z)$: **Likelihood** — probability of the data given the latent
- $p(z)$: **Prior** — our initial belief about z
- $p(x)$: **Evidence** or marginal likelihood:

$$p(x) = \int p(x \mid z) p(z)\, dz.$$

Introduction. Bayes' Rule

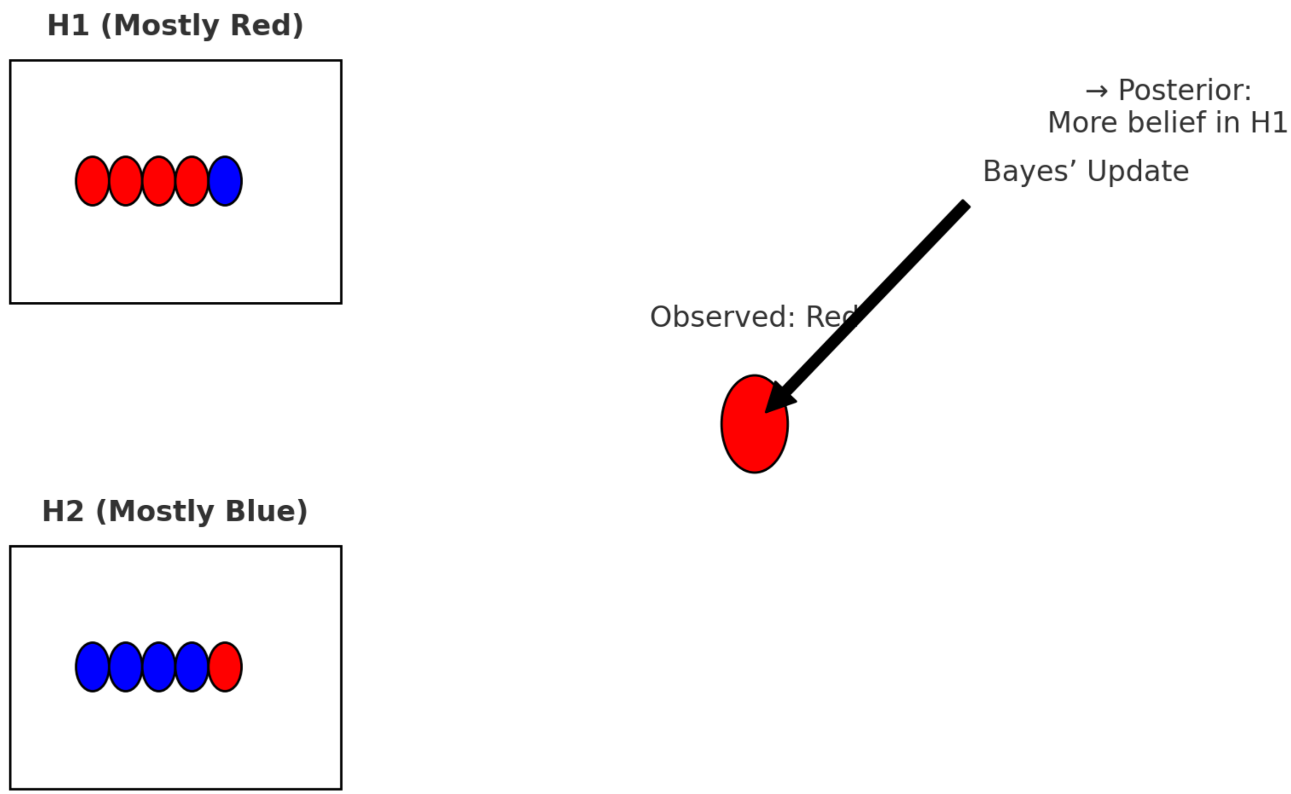
Example

Imagine you have boxes of colored balls (red balls and blue balls). You pull one ball from a box and you want to infer which box you're drawing from.

Two type of boxes:

- H_1: Mostly red balls (let's say 4 red and 1 blue)
- H_2: Mostly blue balls (let's say 1 red and 4 blue)

You draw a red ball. What is the probability it came from H_1 type?

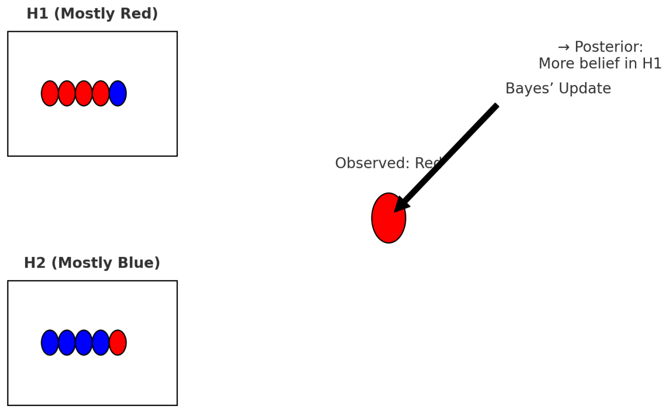

H1 (Mostly Red)
H2 (Mostly Blue)
→ Posterior:
More belief in H1
Bayes' Update
Observed: Red

Introduction. Example

Interpretation:

- **Prior**: initial belief in each box: $1/2$
- **Likelihood**: how likely you are to draw red from each box
- **Posterior**: updated belief after seeing the red ball

Bayes' rule formalizes how **data updates our beliefs**.



Introduction. Bayes' Rule Example

$$p(H_1 \mid Red) = \frac{p(Red \mid H_1)\, p(H_1)}{p(Red)}$$

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}}$$

$$p(H_1 \mid Red) = \frac{4/5 \times\ 1/2}{1/2} = 4/5$$

Introduction

Check this nice visualization: https://setosa.io/ev/conditional-probability/

- In VAEs, **Bayes' Rule** allows to compute the exact posterior $p(z \mid x)$, but it's typically **intractable** to compute.
- So we **approximate** it with $q_\phi(z \mid x)$, learned via variational inference.

Introduction. (KL) Divergence

Kullback–Leibler (KL) Divergence

$$\mathrm{KL}(q\|p) := \int q(z) \log \frac{q(z)}{p(z)}\, dz$$

- Measures the "distance" between two distributions $q(z)$ and $p(z)$:
- Always non-negative: $\mathrm{KL}(q\|p) \geq 0$
- Zero if and only if $q = p$ almost everywhere
- Asymmetric: $\mathrm{KL}(q\|p) \neq \mathrm{KL}(p\|q)$

Used in VAEs to regularize $q(z \mid x)$ towards the prior $p(z)$.

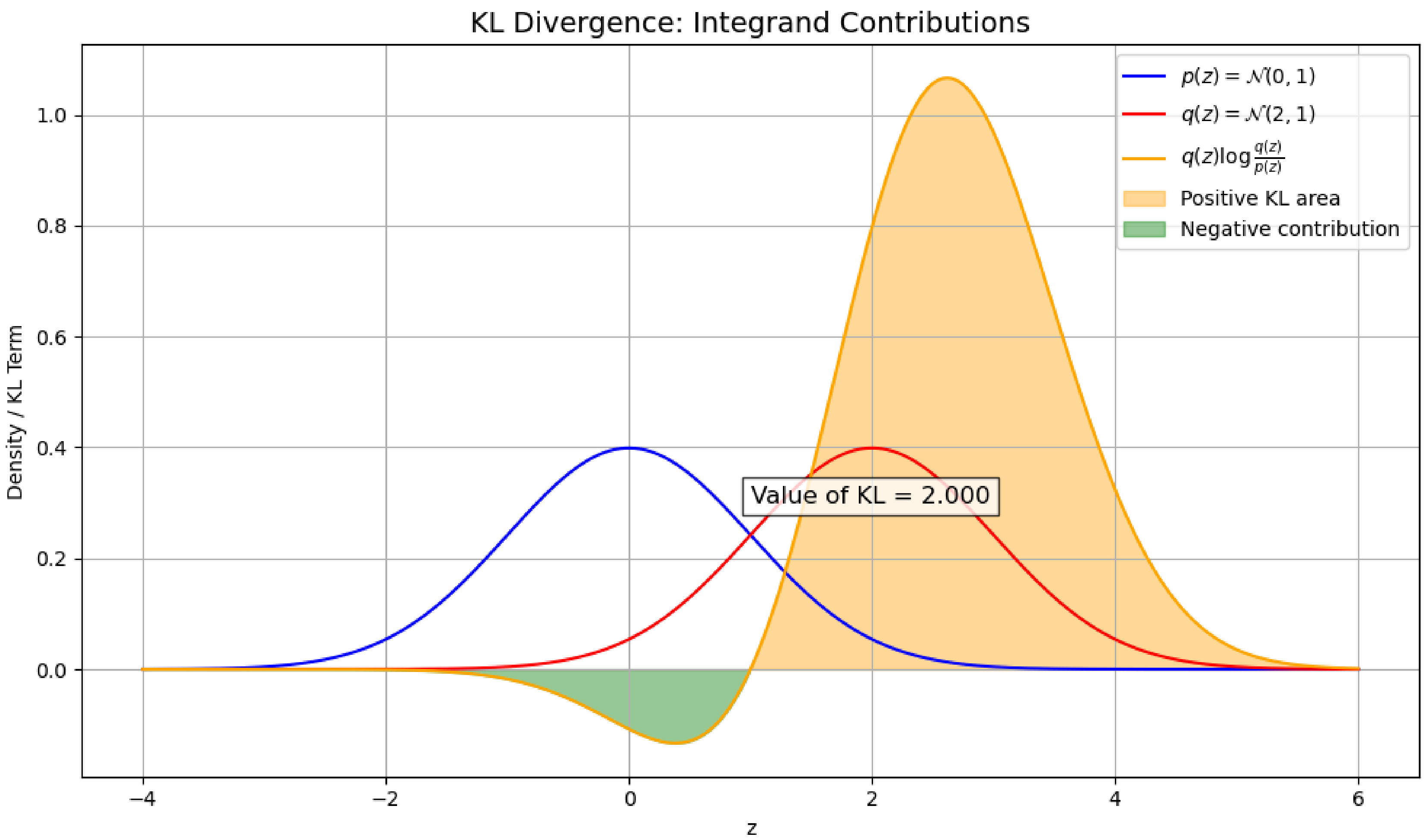
KL Divergence: Normal vs. Standard Normal

When comparing:

- $q(z) = \mathcal{N}(\mu, \sigma^2)$
- $p(z) = \mathcal{N}(0, 1)$

The **analytic KL divergence** is given by:

$$\mathrm{KL}\big(q \,\|\, p\big) \;=\; \frac{1}{2}\Big(\mu^2 + \sigma^2 - 1 - \ln \sigma^2\Big).$$


KL Divergence: Integrand Contributions
$p(z) = \mathcal{N}(0, 1)$
$q(z) = \mathcal{N}(2, 1)$
$q(z) \log \frac{q(z)}{p(z)}$
Positive KL area
Negative contribution
Value of KL = 2.000
Density / KL Term
z
1.0
0.8
0.6
0.4
0.2
0.0
−4
−2
0
2
4
6

Introduction

In VAEs

- We approximate the intractable posterior $p(z \mid x)$ with $q(z \mid x)$
- The KL term is included in the **Loss function** which ensures

$$q(z \mid x) \approx p(z) = \mathcal{N}(0, I)$$

- This **regularizes** the encoder and prevents overfitting.

Variational Autoencoders (VAE)

Variational Autoencoders (VAE)

VAE Latent Space

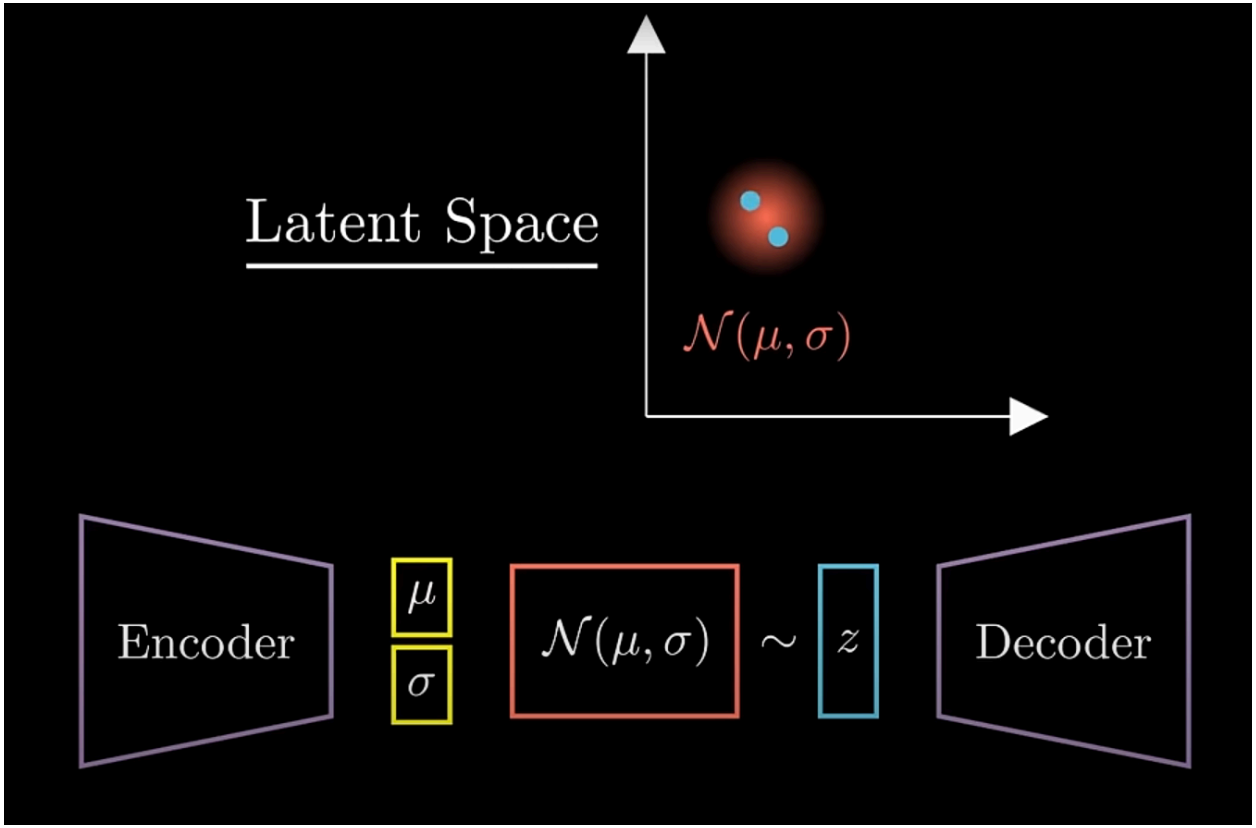


image from: https://www.youtube.com/watch?v=qJeaCHQ1k2w

Variational Autoencoders

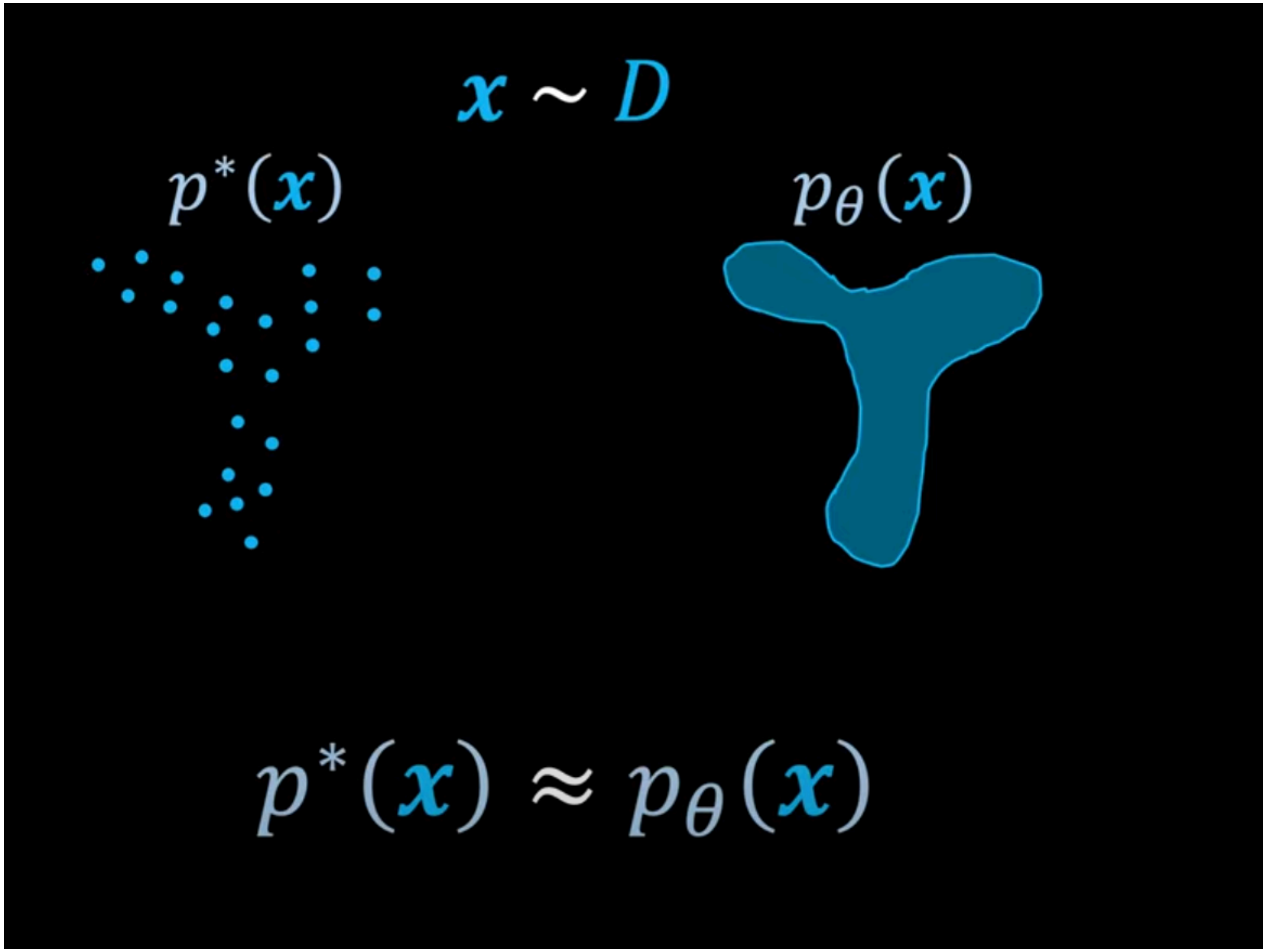


image from: https://www.youtube.com/watch?v=HBYQvKlaE0A

Latent Variable Models

Assume that data $x \in \mathbb{R}^d$ is generated from a latent variable $z \in \mathbb{R}^k$ via a probabilistic model:

$$p(x, z) = p(z)p(x \mid z)$$

- $p(z)$: prior distribution, typically $\mathcal{N}(0, I)$
- $p(x \mid z)$: likelihood (decoder)

The marginal likelihood of the data is:

$$p(x) = \int p(x \mid z)p(z)\, dz$$

The Inference Problem

We want to **maximize the marginal likelihood** $p(x)$ (train the model).

But computing $p(x)$ requires integrating over z, which is **intractable** in general.

Instead, we introduce an approximate posterior:

$$q(z \mid x) \approx p(z \mid x)$$

and optimize a **variational bound**.

Evidence Lower Bound (ELBO): Definition

Let X and Z be random variables with joint model $p_\theta(x, z)$. For any approximate posterior $q_\phi(z|x)$, the ELBO is defined:

$$\mathcal{L}(\phi, \theta; x) := \mathbb{E}_{z \sim q_\phi(\cdot|x)} \left[\ln \frac{p_\theta(x, z)}{q_\phi(z|x)} \right].$$

This can be equivalently written:

$$\mathcal{L} = \ln p_\theta(x) \; - \; \mathrm{KL}\big(q_\phi(z|x) \,\|\, p_\theta(z|x)\big).$$

Thus, maximizing $\mathcal{L}$ simultaneously **increases** the data log-likelihood $\ln p_\theta(x)$ and **reduces** the divergence between the approximate and true posterior.

ELBO: Alternate Form & Key Intuition

Alternate ELBO form:

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

First term: expected log-likelihood or reconstruction quality

Second term: KL regularization, encourages $q_\phi(z|x)$ to stay near the prior $p(z)$.

Intuition:

- Converts an **intractable inference problem** into a **tractable optimization**
- Acts as a **trade-off**: accurate reconstructions vs. structured latent space

Variational Autoencoders

VAE jointly learns:

- An **encoder** $q_\phi(z \mid x)$ — maps input x to latent z
- A **decoder** $p_\theta(x \mid z)$ — reconstructs x from latent z (reconstruction likelihood)
- Typically, $p(z) = \mathcal{N}(0, I)$

Both components are parameterized by neural networks. The model is trained by maximizing the **ELBO** over the dataset.

VAE Loss Function

For a single observation x, we have:

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

Variational Autoencoders

Notion of encoding

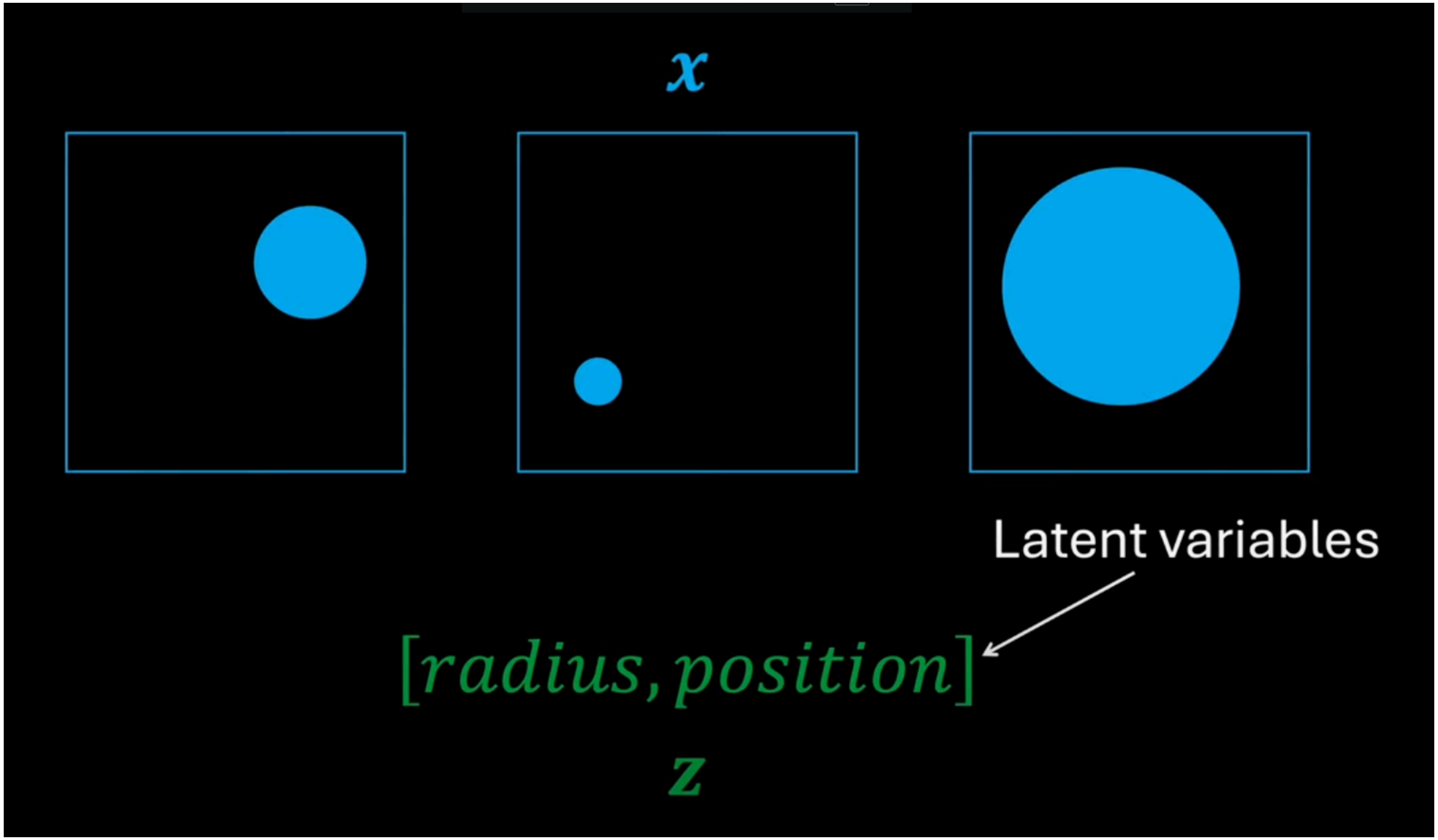


image from: https://www.youtube.com/watch?v=HBYQvKlaE0A

Encoding - Decoding functions

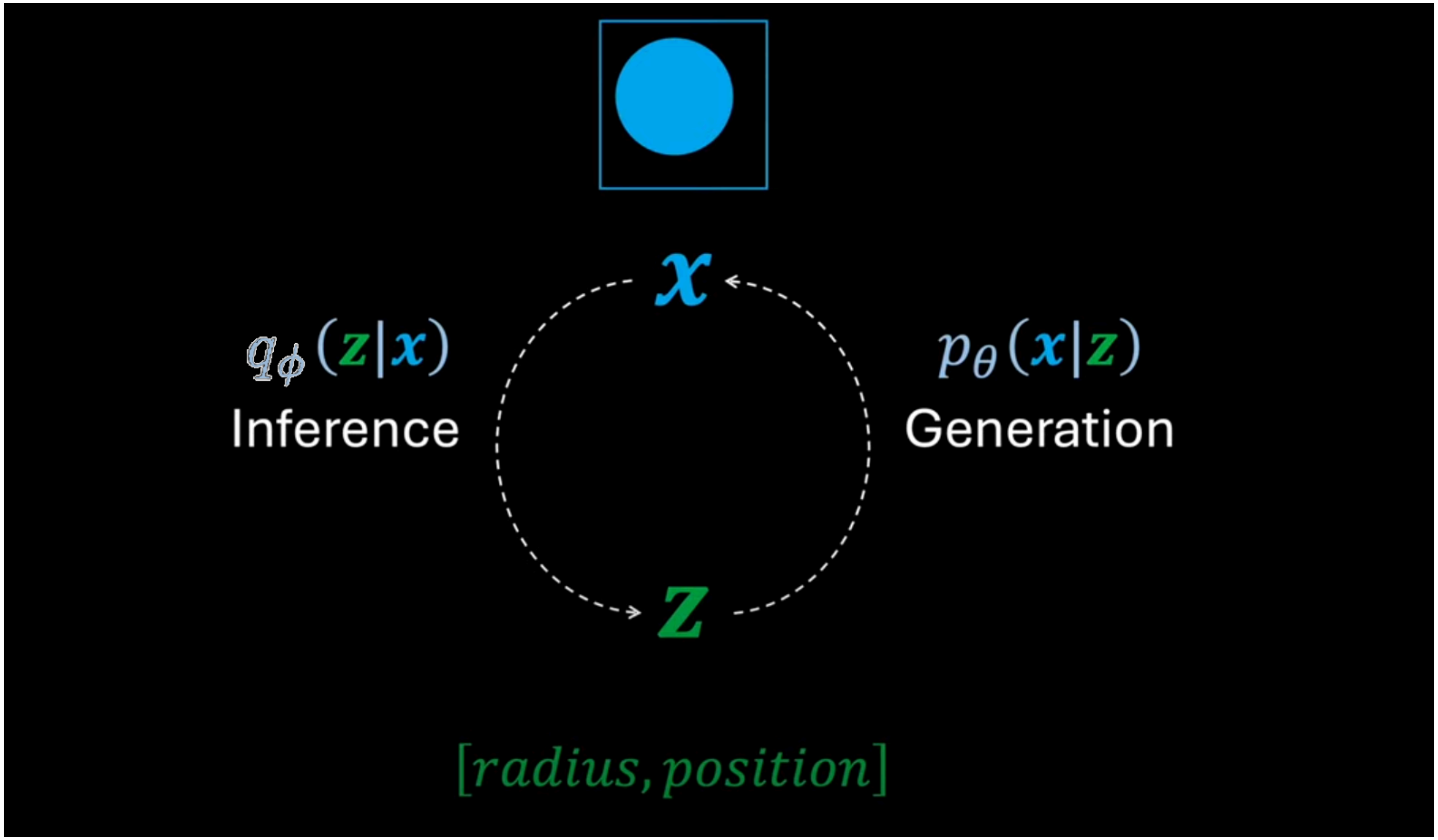


image from: https://www.youtube.com/watch?v=HBYQvKlaE0A

Simple Autoencoders

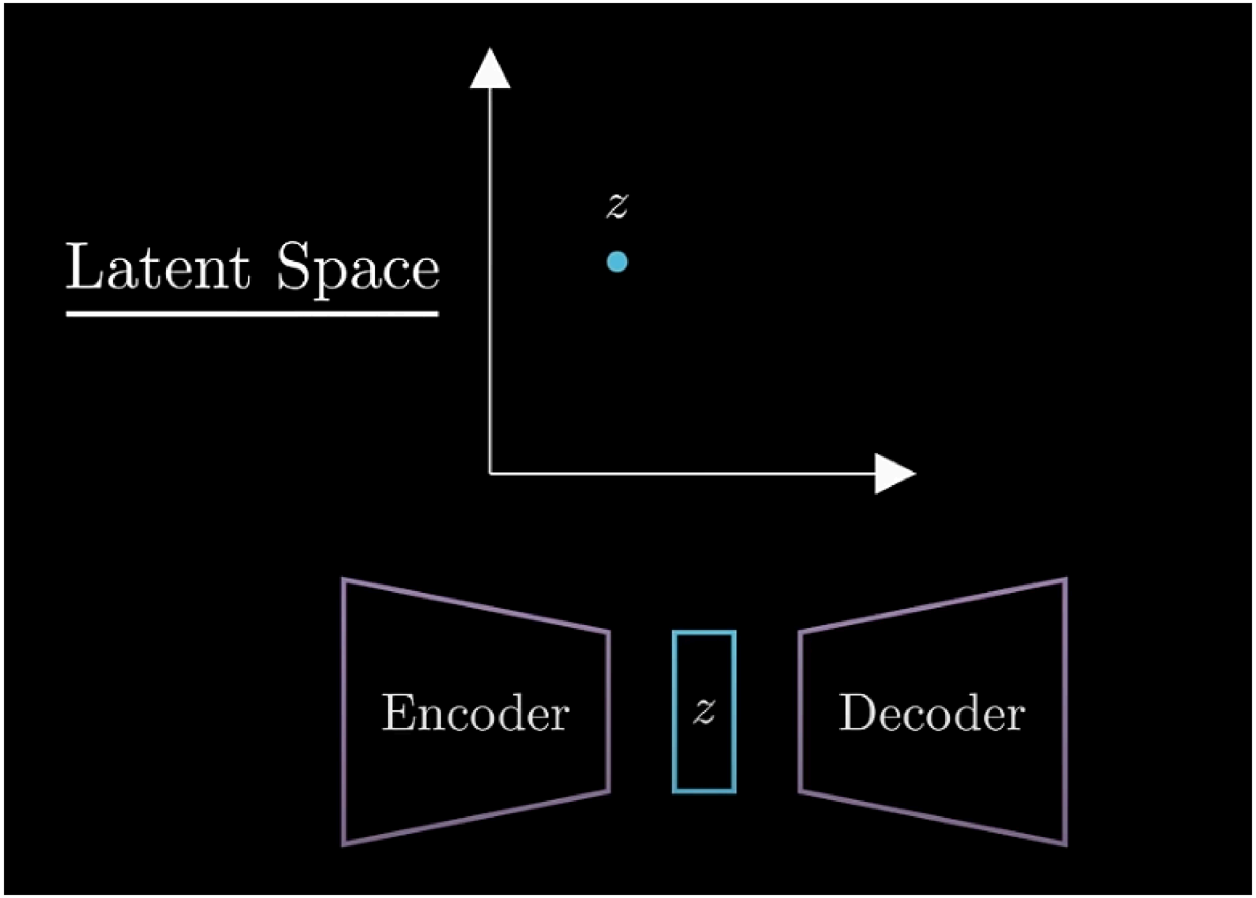


image from: https://www.youtube.com/watch?v=qJeaCHQ1k2w

Classical Autoencoders

An **Autoencoder** is a neural network trained to reconstruct its input:

- **Encoder**: $x \mapsto z = f_\phi(x)$
- **Decoder**: $z \mapsto \hat{x} = g_\theta(z)$
- **Objective**: minimize the difference between x and $\hat{x}$

Common Loss Functions

Mean Squared Error (MSE):

$$\mathcal{L}(x, \hat{x}) = \|x - \hat{x}\|^2 = \sum_i (x_i - \hat{x}_i)^2.$$

Popular for image data.

Goal

Learn **compressed latent representations** z that retain meaningful information to reconstruct x.

VAE Latent Space

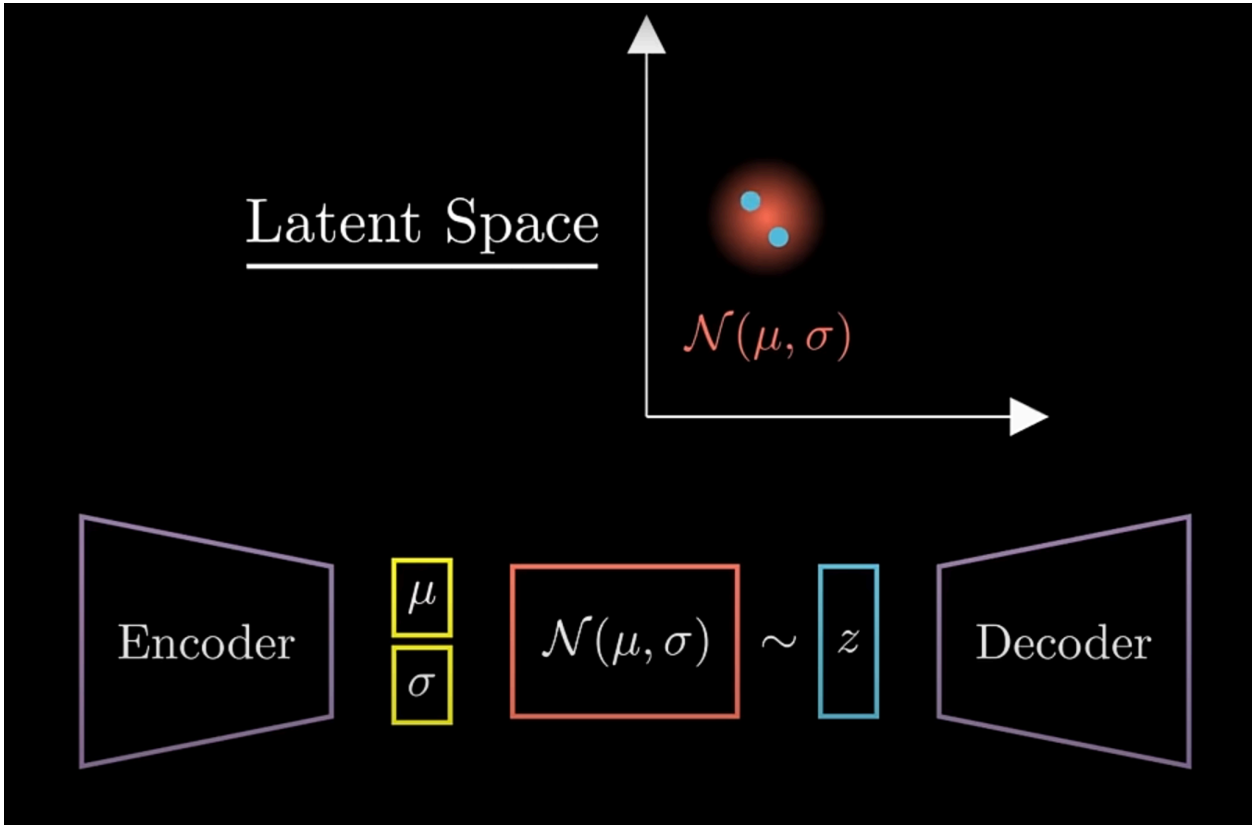


image from: https://www.youtube.com/watch?v=qJeaCHQ1k2w

The Reparameterization Trick: Definition

In VAEs, we want to compute gradients of the expected Loss with respect to the encoder parameters ϕ. We need to compute:

$$\nabla_\phi \mathbb{E}_{z \sim q_\phi(z|x)}[f(z)].$$

However, the operation of **sampling from** $q_\phi(z|x)$ is stochastic, which blocks gradients.

Variational Autoencoders

Key idea:

Reparametrize $\sim \mathcal{N}(\mu, \sigma^2 I)$:

$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I).$$

This changes the expression of the expected value:

$$\mathbb{E}_{z \sim q_\phi(z)}[f(z)] = \mathbb{E}_{\epsilon \sim p(\epsilon)}[f(g_\phi(\epsilon))].$$

So gradients **flow through** g_ϕ deterministically. This enables **gradient-based training** via backpropagation.

Intuition

- Move randomness into a fixed distribution $\epsilon \sim \mathcal{N}(0, I)$
- Compute gradients of a deterministic function $g_\phi(\epsilon)$
- Applicable to **continuous distributions** like Gaussians

Reparametrization Trick

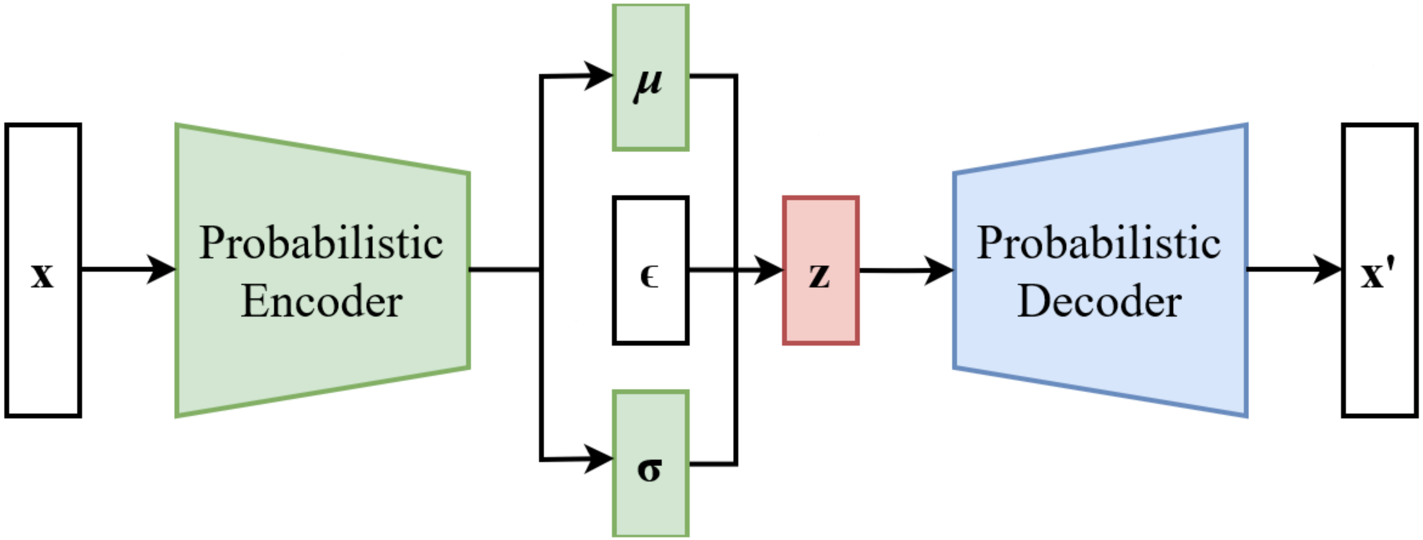


image from:
https://en.wikipedia.org/wiki/Reparameterization_trick#/media/File:Reparameterized_Variational_Autoencoder.png

VAE evaluation and Loss computation

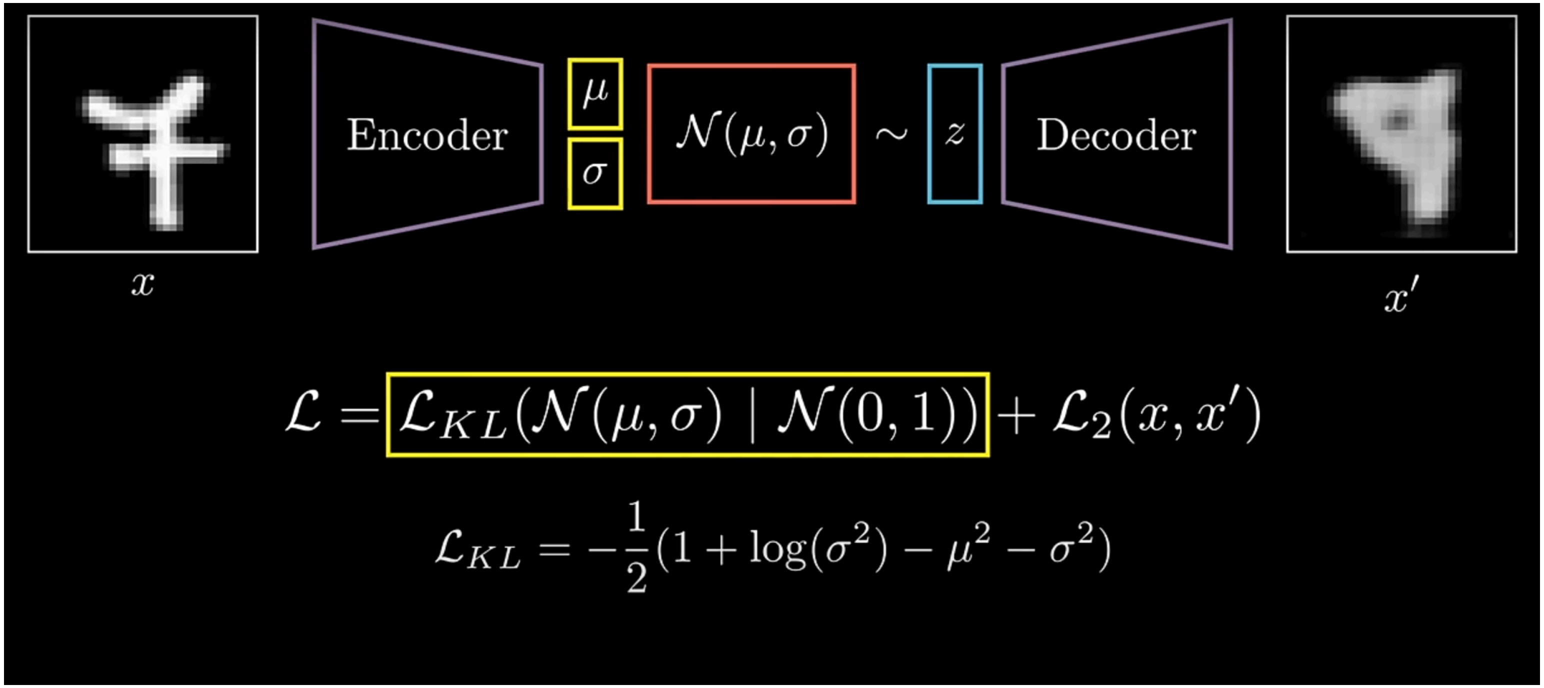


image from: https://www.youtube.com/watch?v=qJeaCHQ1k2w

VAE training process

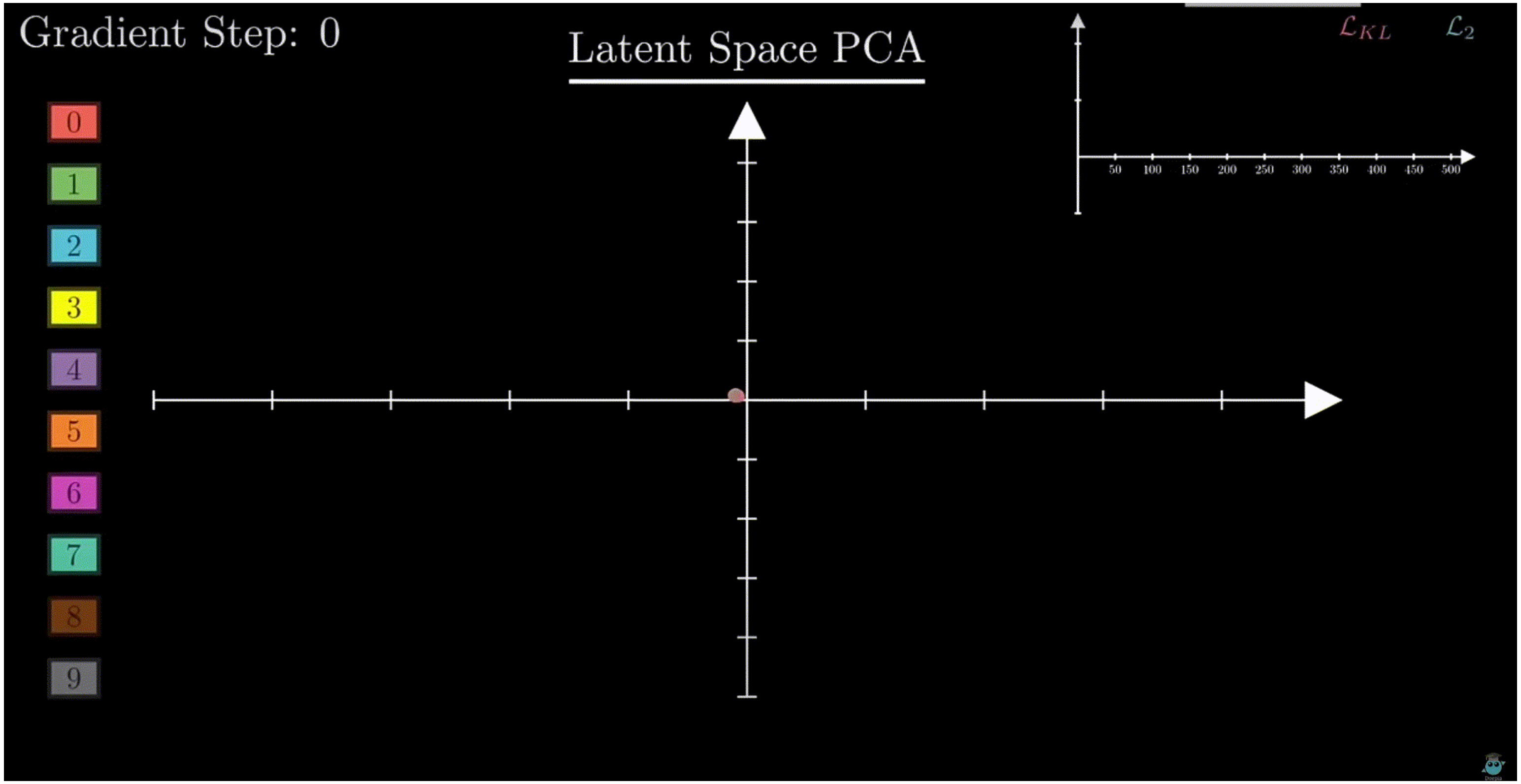


image from: https://www.youtube.com/watch?v=qJeaCHQ1k2w

Summarizing

- The basic VAE model imposes a **Gaussian prior** on the latent space:
  Encourages **smooth, continuous** embeddings
- Similar inputs map to **nearby** points in latent space
- Enables **interpolation** and **generation** of new samples
- The latent space can be visualized using 2D or 3D projections

Applications

VAEs for Classification

Idea:

Use the encoder to extract latent representations:

$$x \mapsto z = \mu_\phi(x).$$

Then use z as features for a standard classifier (e.g., Kmeans, KNN, logistic regression, etc.).

Benefits:

- Dimensionality reduction
- Denoising
- Incorporates structure from unlabeled data

Case Study: MNIST with VAE Embeddings

- Train VAE on MNIST (unlabeled)
- Encode each image into latent vector z
- Train a simple classifier on top of z

Observations:

- Comparable or better accuracy than raw pixels
- Latent space clusters according to digit classes
- Works well with few labeled samples (semi-supervised learning)

The MNIST dataset

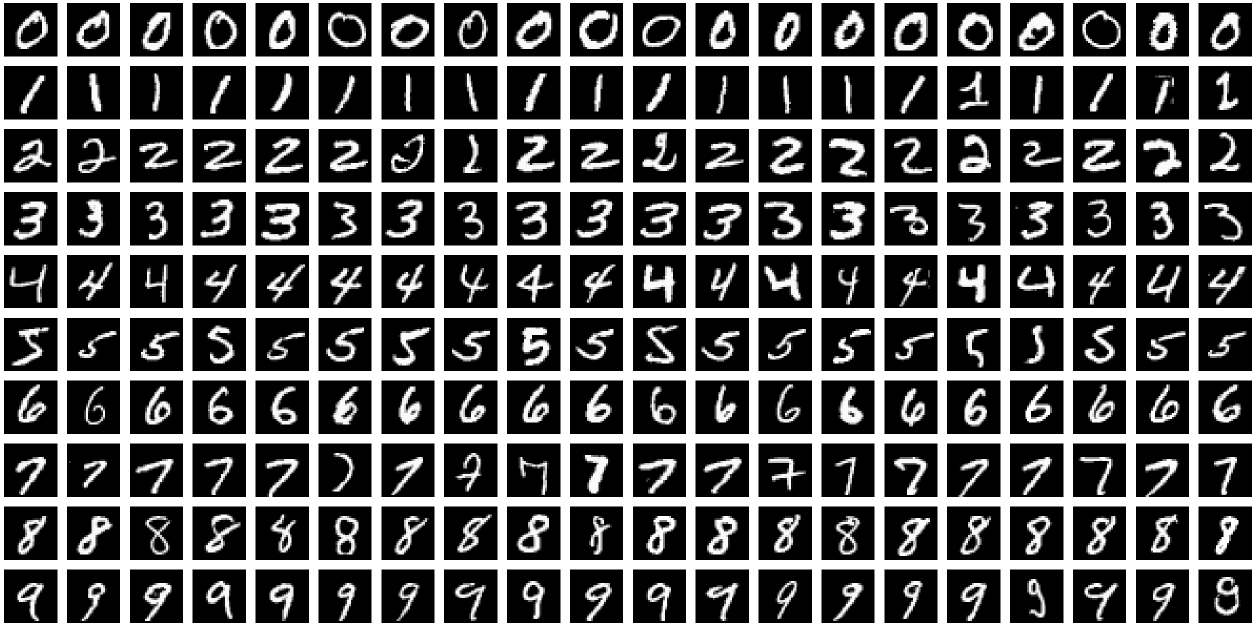


image from: https://en.wikipedia.org/wiki/MNIST_database#/media/File:MNIST_dataset_example.png

Applications

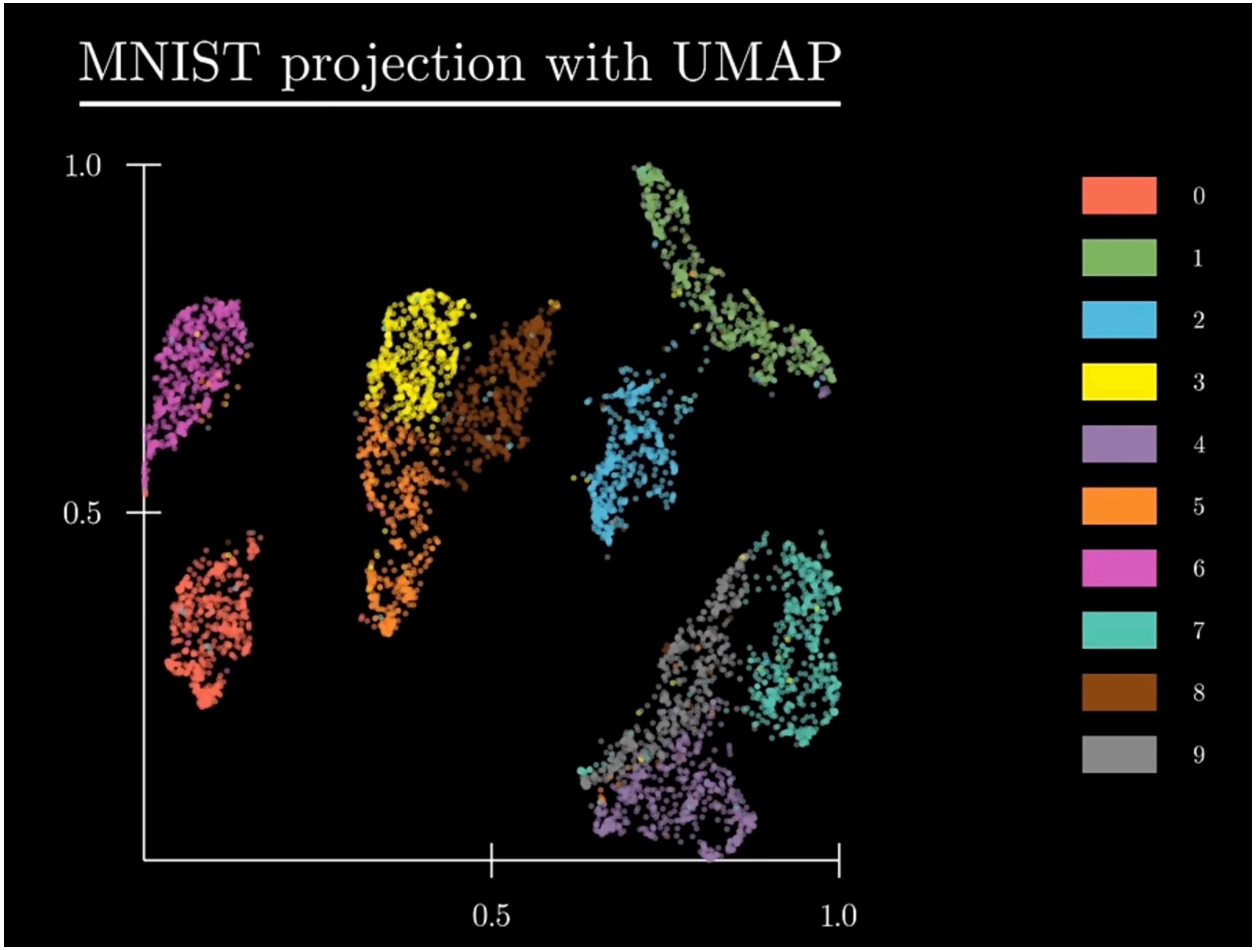
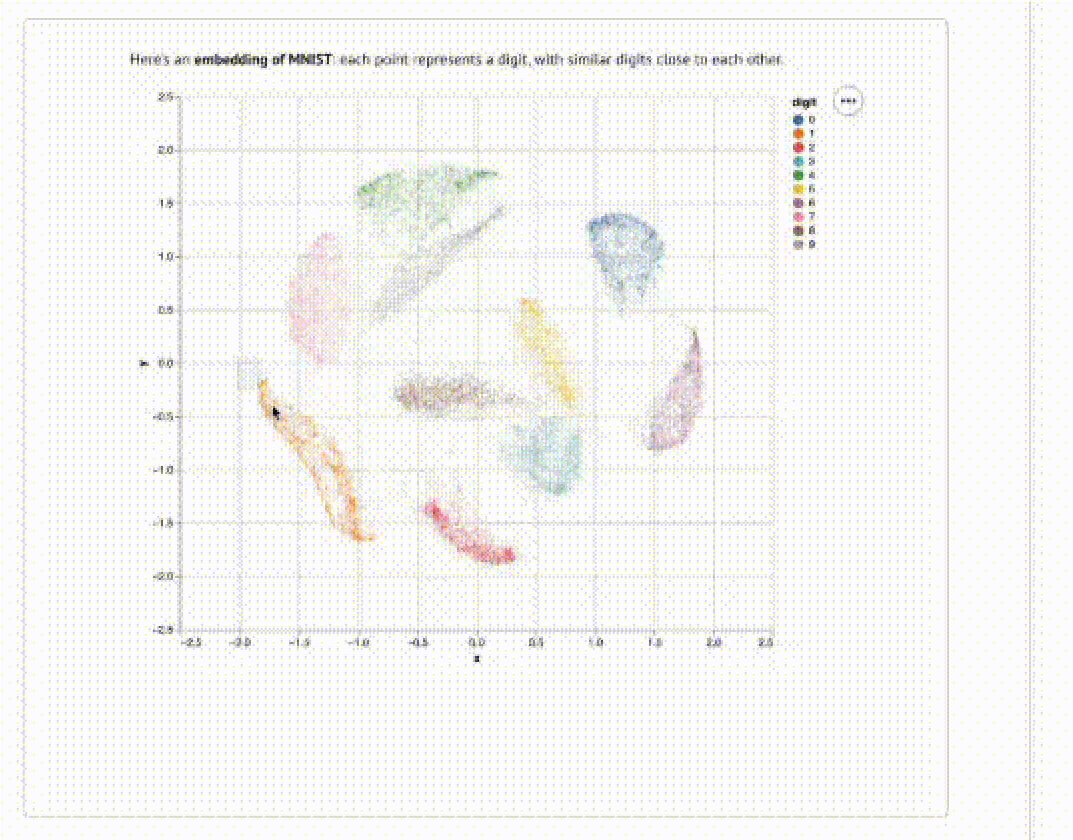


image from: https://www.youtube.com/watch?v=o_cAOa5fMhE

Applications

Two-dimensional projection of the latent space



Applications

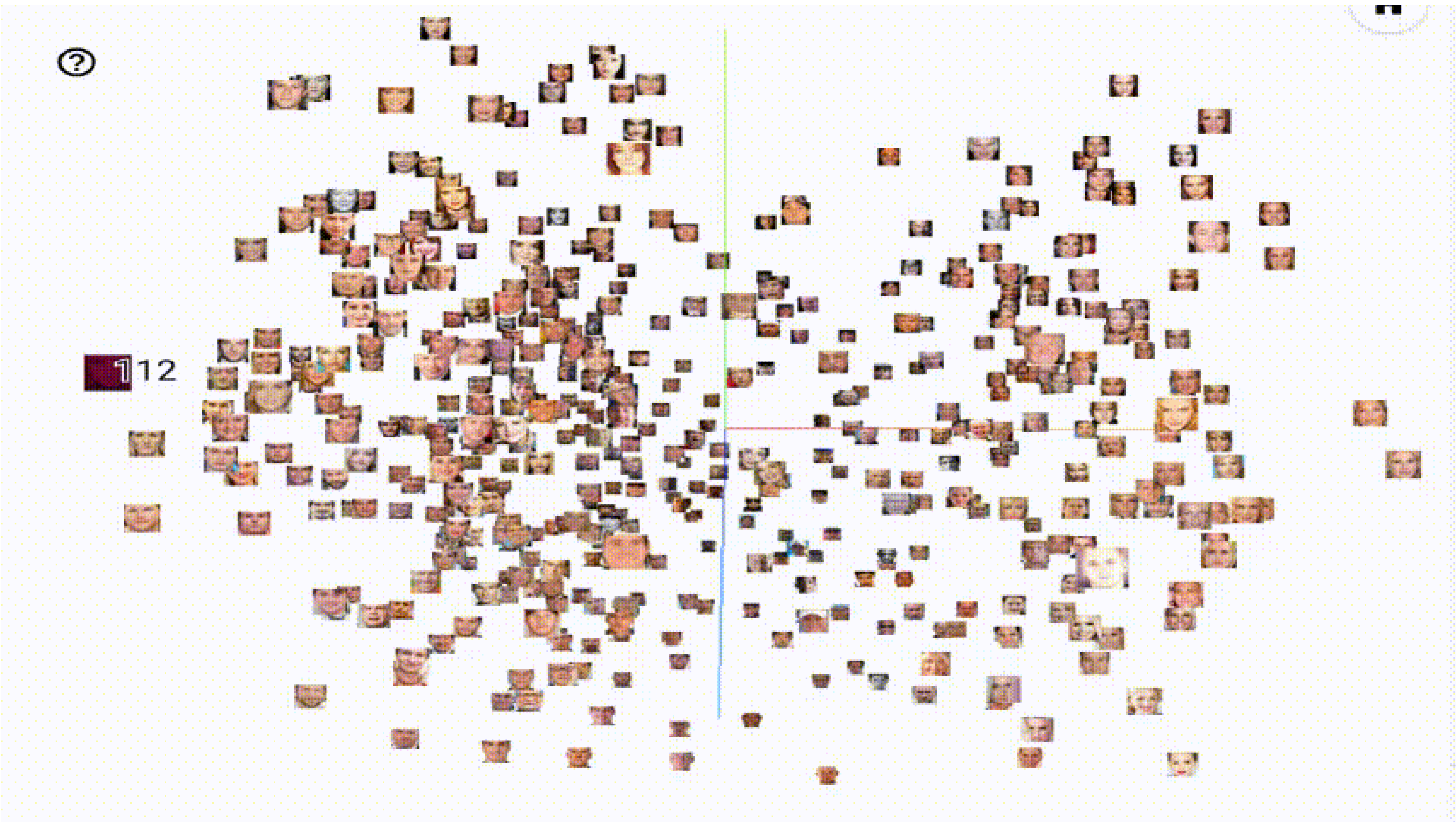
Example: Sampling from the latent space

Applications

Example: Sampling from the latent space

Applications

Example: Sampling from the latent space. 3D visualization



Conclusion

- VAEs unify **variational inference**, **neural networks**, and **generative modeling**
- Provide a principled framework for:
  - **Unsupervised learning**
  - **Representation learning**
  - **Classification and data synthesis**
- Rich area for **theoretical research** and **practical applications**

References

[1]- Kingma, D. P., & Welling, M. (2014).
Auto-Encoding Variational Bayes.
Proceedings of ICLR.
arXiv:1312.6114

[2]- Rezende, D. J., Mohamed, S., & Wierstra, D. (2014).
Stochastic Backpropagation and Approximate Inference in Deep Generative Models.
Proceedings of ICML.
arXiv:1401.4082

[3]- Doersch, C. (2016).
Tutorial on Variational Autoencoders.
arXiv:1606.05908

[4]- Higgins, I. et al. (2017).
β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework.
ICLR.

[5]- Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. (2017).
Variational Inference: A Review for Statisticians.
Journal of the American Statistical Association, 112(518), 859–877.

Useful links

https://www.youtube.com/watch?v=qJeaCHQ1k2w

https://arxiv.org/pdf/1906.02691

https://github.com/ytdeepia/Variational-Autoencoders

https://github.com/ytdeepia/Variational-Autoencoders

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

https://www.youtube.com/watch?v=HBYQvKlaE0A

Thank You!

Feel free to reach out: **willyrv@gmail.com**