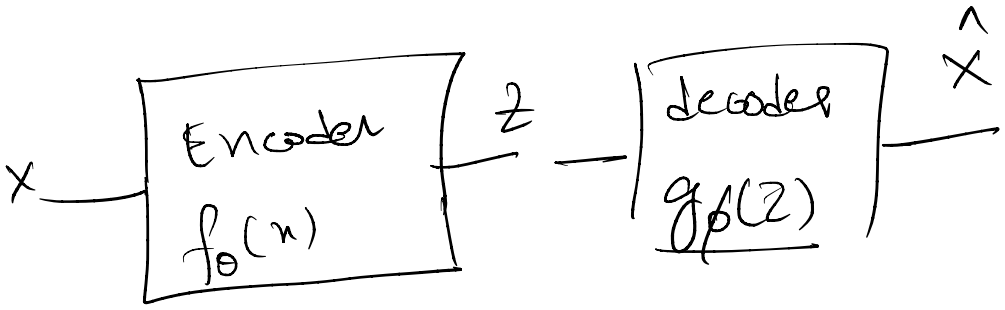
DUL: Autoencoders

Naeemullah Khan
naeemullah.khan@kaust.edu.sa

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

KAUST Academy
King Abdullah University of Science and Technology

May 27, 2025



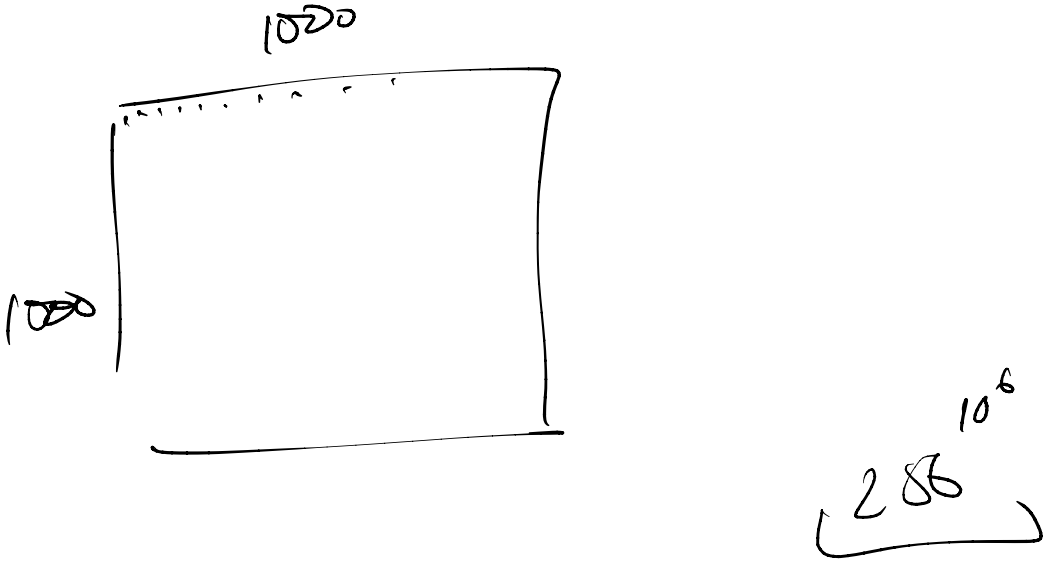
$$\min_{\theta,\phi} \|x - \hat{x}\|_2^2$$

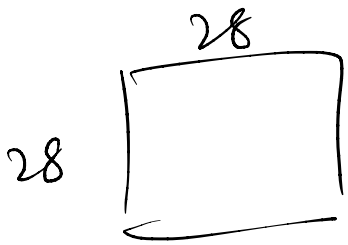
$x \in \mathbb{R}^d$

$z \in \mathbb{R}^{d'}$

latent space code

$d' < d$





$z = Wx + B$

$x \in \mathbb{R}^{784}$

$z \in \mathbb{R}^{10}$

$s(W) \rightarrow (100 \times 784)$ $s(b) = 100 \times 1$

$f_\theta \rightarrow$ 2 layer NN — 100 hidden unit

x — W^1, b^1 — W^2, b^2

$s(W^2) \rightarrow (10, 100)$ $s b^2 = (10 \times 1)$ $g_\theta \rightarrow$ 2 layer NN — 100 hidden units

z — U^1, d^1 — U^2, d^2

$s(U^1) \rightarrow (100, 10)$

$s(d^1) \rightarrow (100, 1)$

$s(U^2) = (784 \times 100)$

$s(d^2) = (784 \times 1)$

$$Z = WX + b$$

↓

column vector

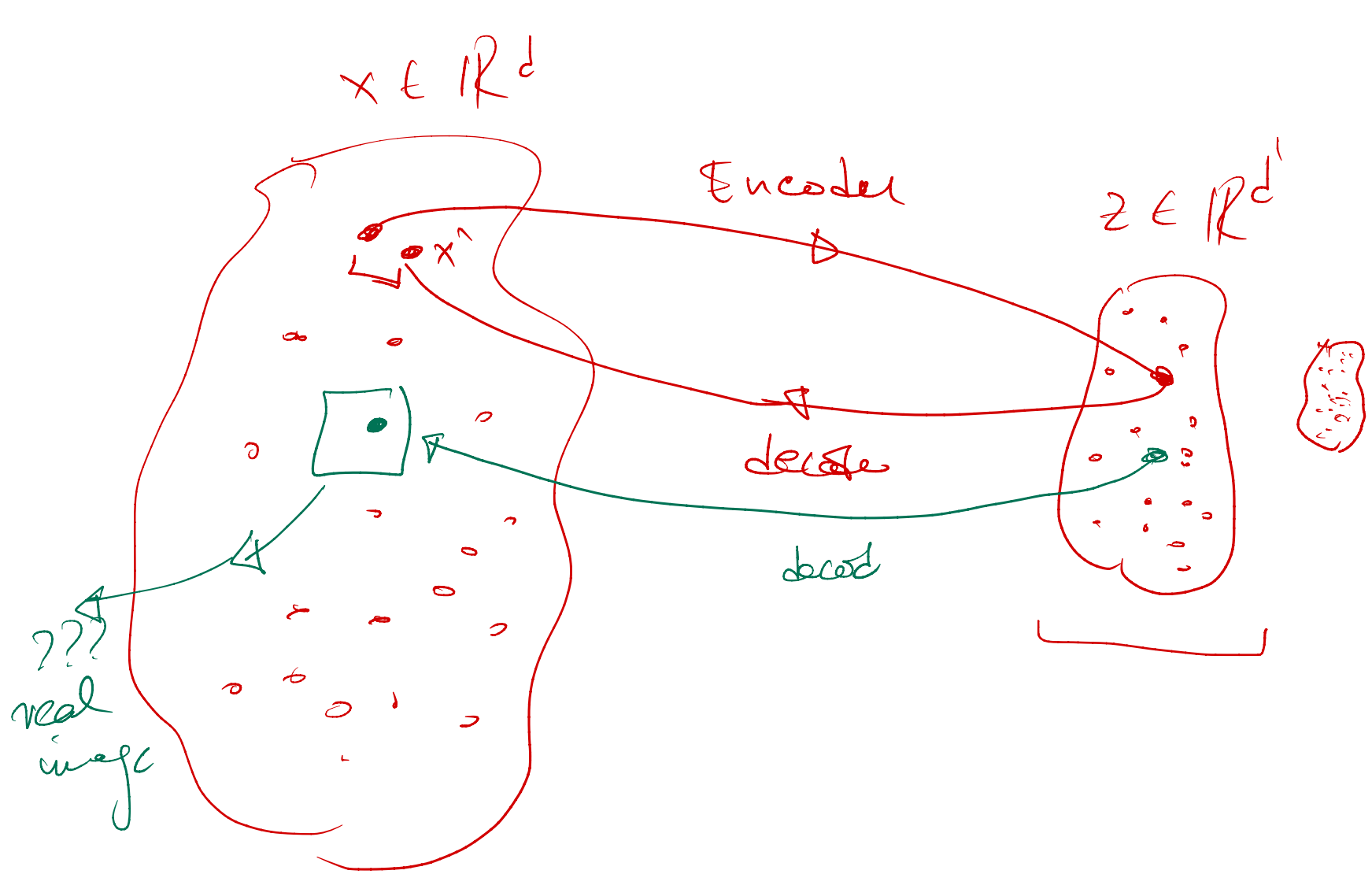
textbook / lecture

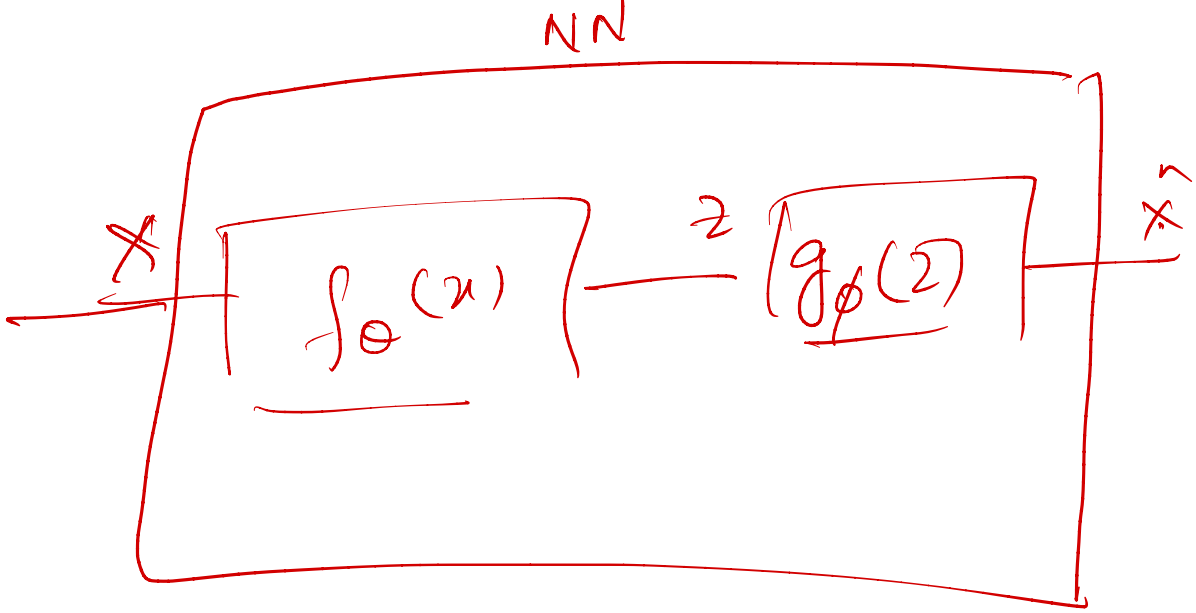
$$Z = X W^T + b$$

↓

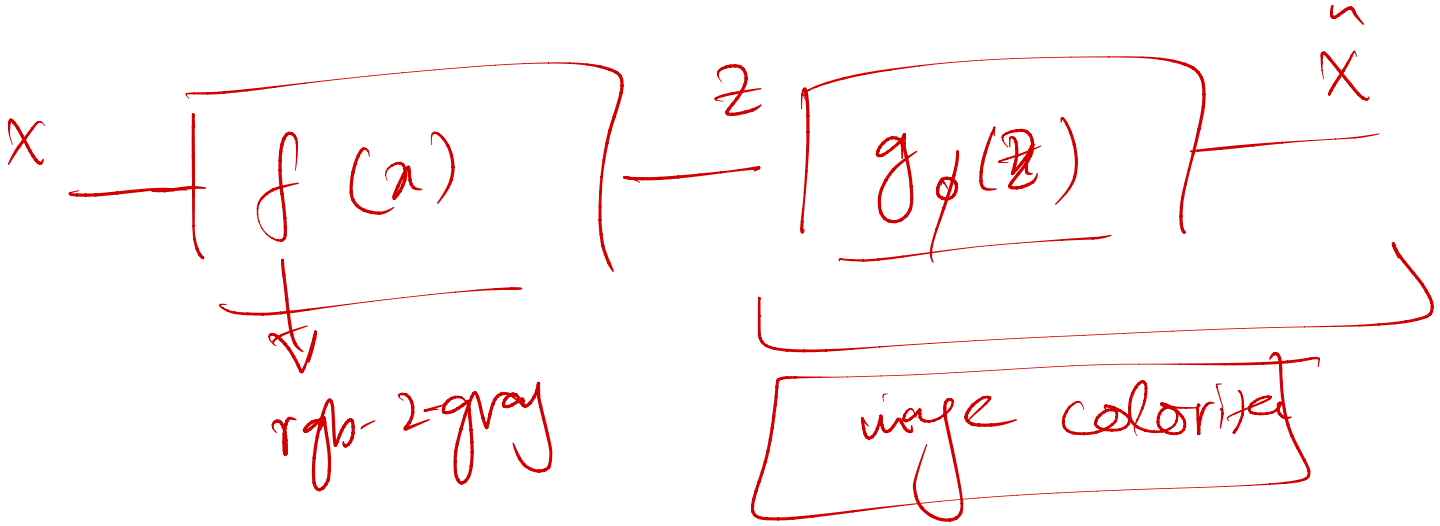
row vector

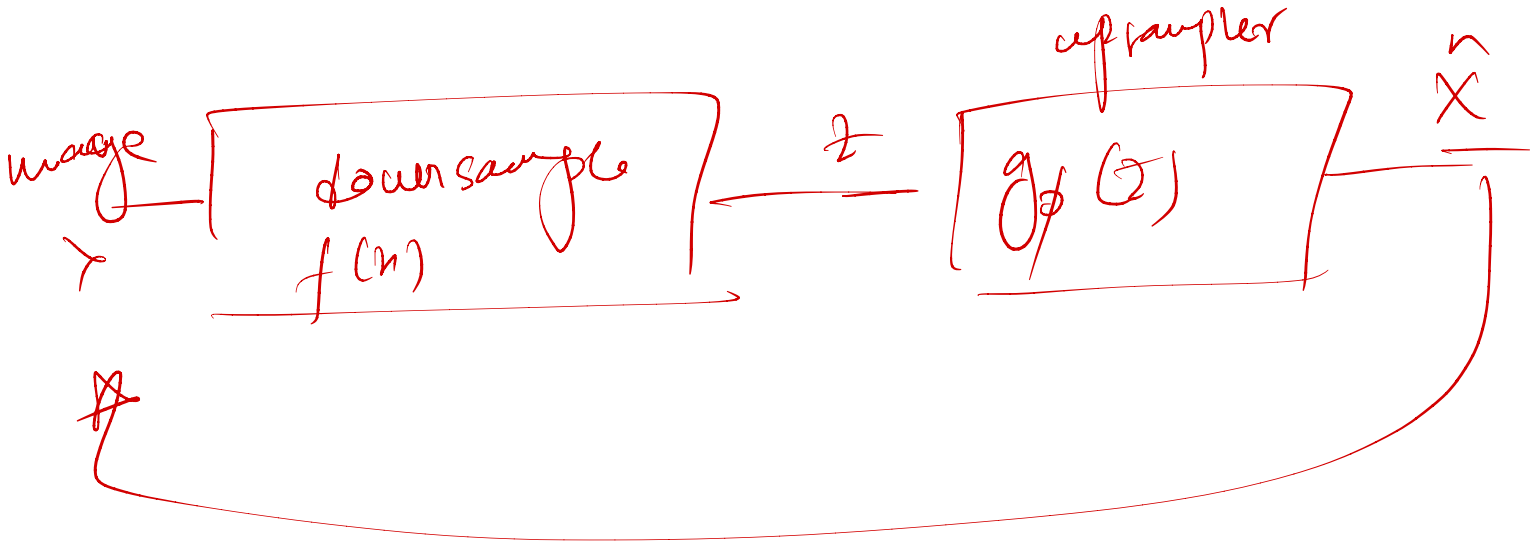
deep learning libraries


$x \in \mathbb{R}^d$
Encoder
$z \in \mathbb{R}^{d'}$
x'
decode
decod
???
real
image



$$\min_{\theta,\phi} \| x - \hat{x} \|_2^2$$




upsampler
image
downsample
$f(x)$
z
$g_\phi(z)$
$\hat{x}$
x

Variational AE

Table of Contents

Introduction

Autoencoders are a type of artificial neural network used for learning efficient codings of input data in an unsupervised manner. The goal is to learn a compressed (encoded) representation of the input and then reconstruct the original input from this compressed version.

- **Developed originally in the 1980s**, autoencoders have gained renewed popularity due to advances in deep learning and hardware.
- Often used in **dimensionality reduction, denoising, anomaly detection, and generative modeling**.

Autoencoders

- Family of neural networks for which the input is the same as the output. They work by compressing the input into a latent-space representation, and then reconstructing the output from this representation.
- The idea is to project the input into a latent space and then reconstruct the input from that latent space representation
- Consist of two parts: Encoder and decode.
  - **Encoder** projects the input to a latent space Z (compresses the input into a lower-dimensional representation).
  - **Decoder** takes the encoded embedding vector and reconstructs the input from it.
  - We also use altered versions of input as output which can be even more interesting.

Autoencoders (cont.)

- **Bottleneck**
  - The central, compressed layer that forces the network to learn the most important features.
  - Introduces a constraint to avoid learning a trivial identity function.
- **Loss Function**
  - Usually **Mean Squared Error (MSE)** between the input and the reconstructed output.
  - Other losses (e.g., binary cross-entropy, KL divergence) can be used depending on the application.

Autoencoders: Architecture

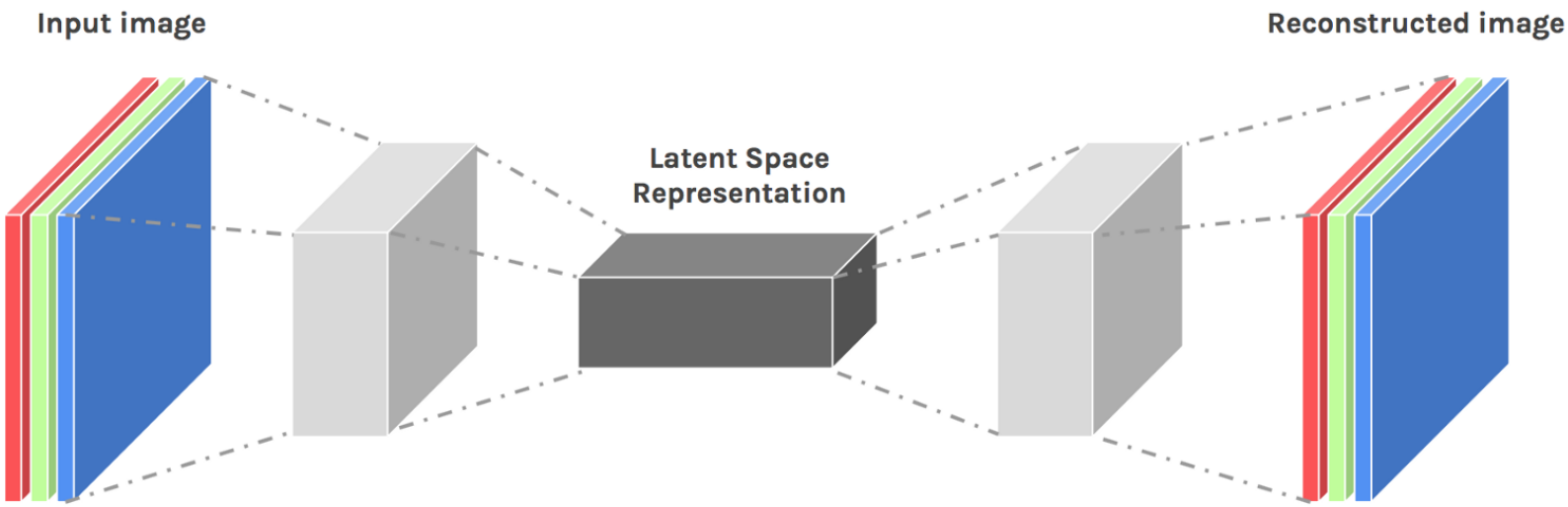


Figure 2: Autoencoder architecture

Autoencoders: Architecture (cont.)

Design Considerations:

- **Depth**: Deeper architectures can capture more abstract representations but may increase the risk of overfitting and require more training data.
- **Width**: Wider layers (more neurons) increase model capacity but can reduce compression effectiveness.
- **Symmetry**: Encoders and decoders are often symmetrical but do not have to be. Asymmetrical designs may be better suited for specific tasks.
- **Regularization**: Techniques such as dropout, weight decay (L1/L2), and sparsity constraints help reduce overfitting and improve generalization.
- **Latent Dimension Size**: A crucial parameter that determines how compressed the representation is. Too small may lose information; too large may not compress enough.
- **Activation Functions**: Nonlinearities like ReLU, sigmoid, or tanh allow the model to learn complex mappings. The final layer typically uses a sigmoid or linear activation depending on the output format.

Autoencoders: Architecture (cont.)

Component	Description
Encoder	Transforms the input data into a compressed latent representation using layers such as Dense, Convolutional, or Recurrent layers.
Latent Space (Bottleneck)	The central compact representation of the input. Forces the model to learn the most essential features.
Decoder	Reconstructs the input from the latent space. Typically mirrors the encoder's structure in reverse.
Loss Function	Measures the difference between input and output (e.g., Mean Squared Error, Binary Cross-Entropy). Drives learning during training.
Training	Involves minimizing reconstruction error using gradient-based optimization methods such as SGD or Adam.

Table 1: Architecture Summary Table

Autoencoders: Architecture (cont.)

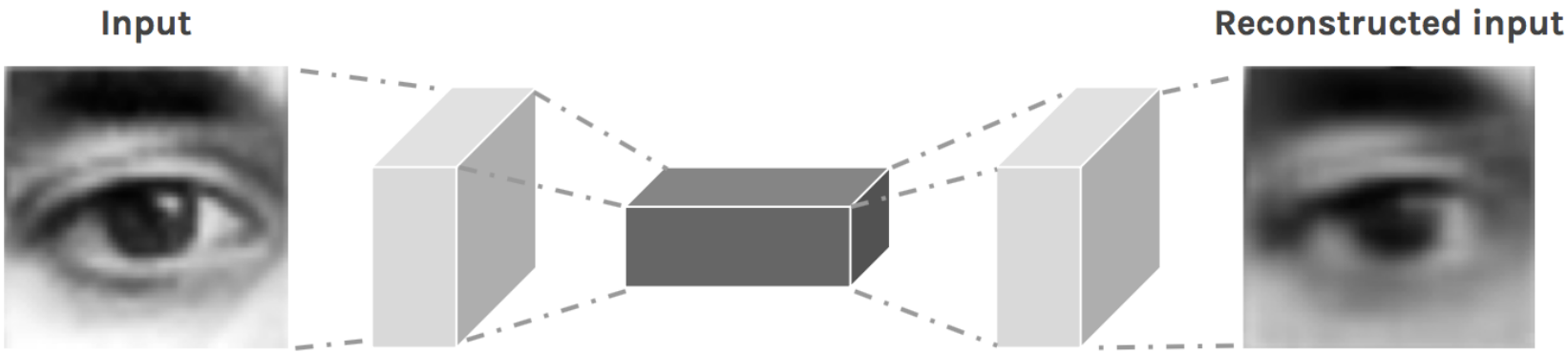


Figure 3: Sample Autoencoder

Autoencoders: Interactive Demo

https://douglasduhaime.com/posts/visualizing-latent-spaces.html

Autoencoders as Generative Models

- Autoencoders project data into a latent space Z.
- The latent space is not necessarily continuous or structured.
- *What if we sample a new embedding vector from Z and then have the decoder reconstruct the image from it?*
- **Does not work**. Autoencoders just learn a function that maps input to output. The learned latent space is too discontinous to work as a generative model.
- Sampling randomly from the latent space may result in invalid outputs.
- They do not maximize the likelihood of the data.

Autoencoders as Generative Models (cont.)

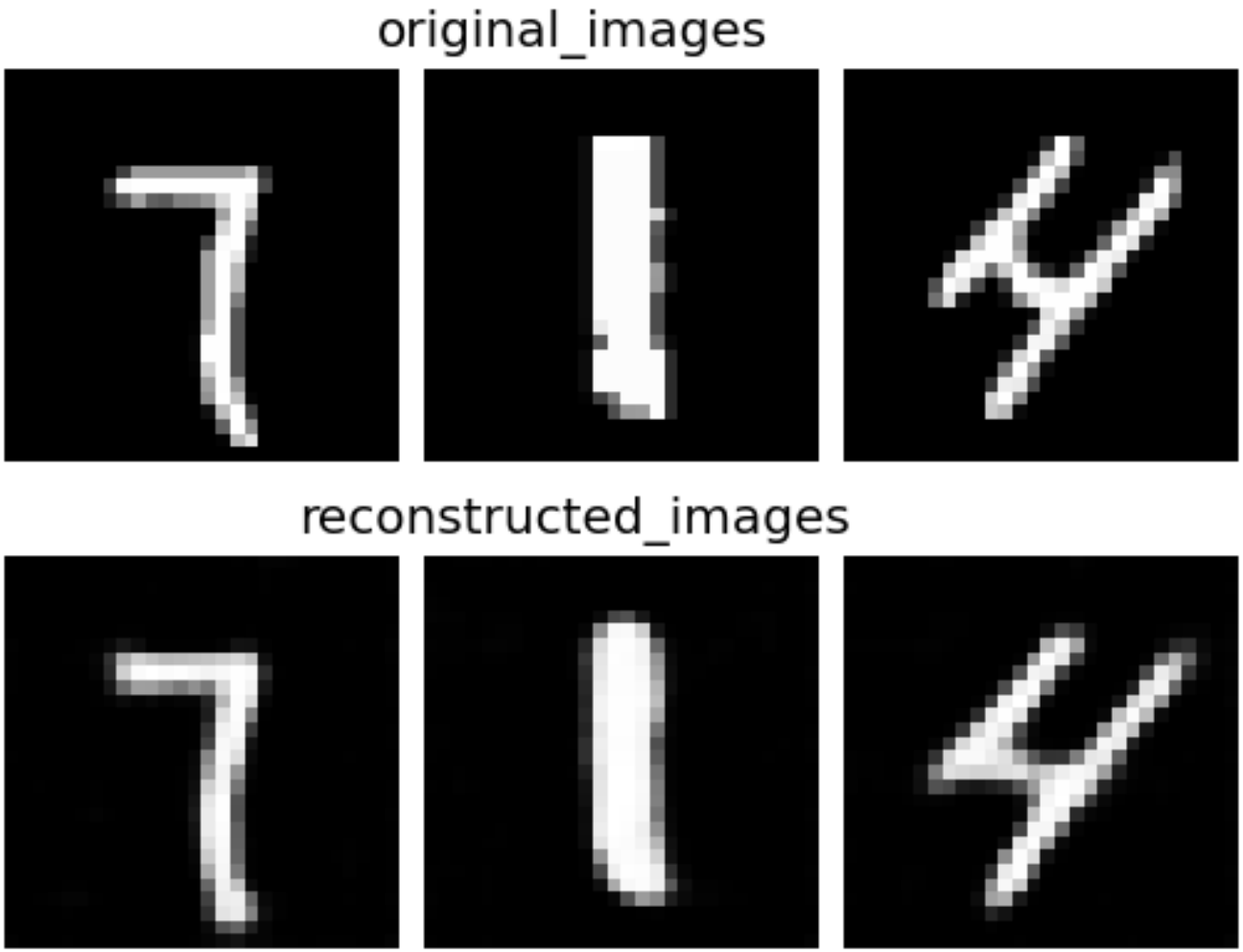


Figure 4: Image reconstruction with autoencoder trained on MNIST digits

Autoencoders as Generative Models (cont.)

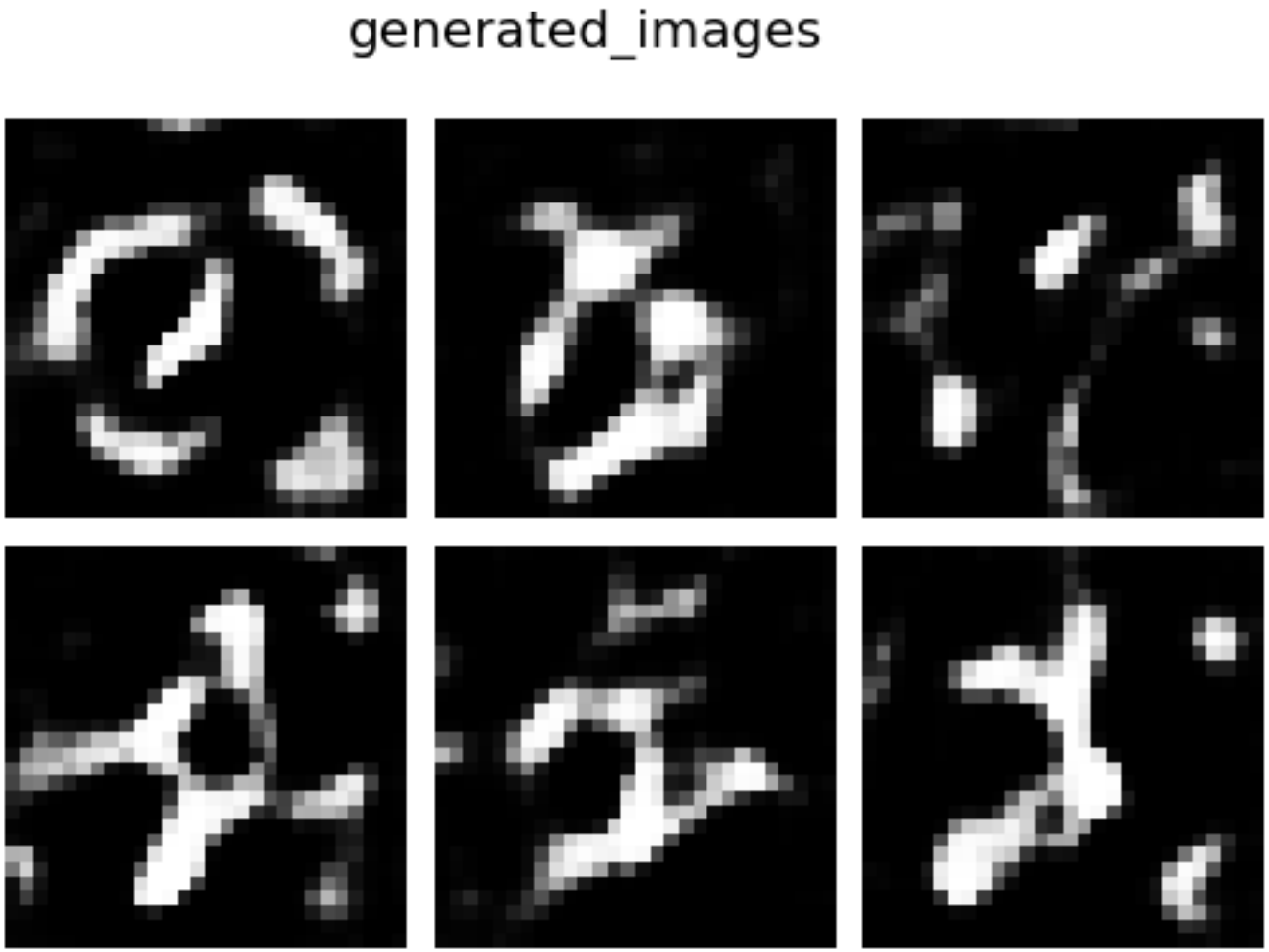


Figure 5: Image generation with autoencoder trained on MNIST digits. Encoding vector sampled from latent space Z and the passed to decoder.

Dimensionality Reduction

Autoencoders are widely used as nonlinear alternatives to traditional dimensionality reduction techniques like PCA (Principal Component Analysis).

- The encoder compresses high-dimensional data into a low-dimensional latent space.
- This latent representation preserves meaningful features and patterns.
- Especially effective for visualizing data or preprocessing inputs for downstream tasks.

Use Case: Visualizing high-dimensional datasets (e.g., MNIST) in 2D or 3D.

Autoencoders: Applications (cont.)

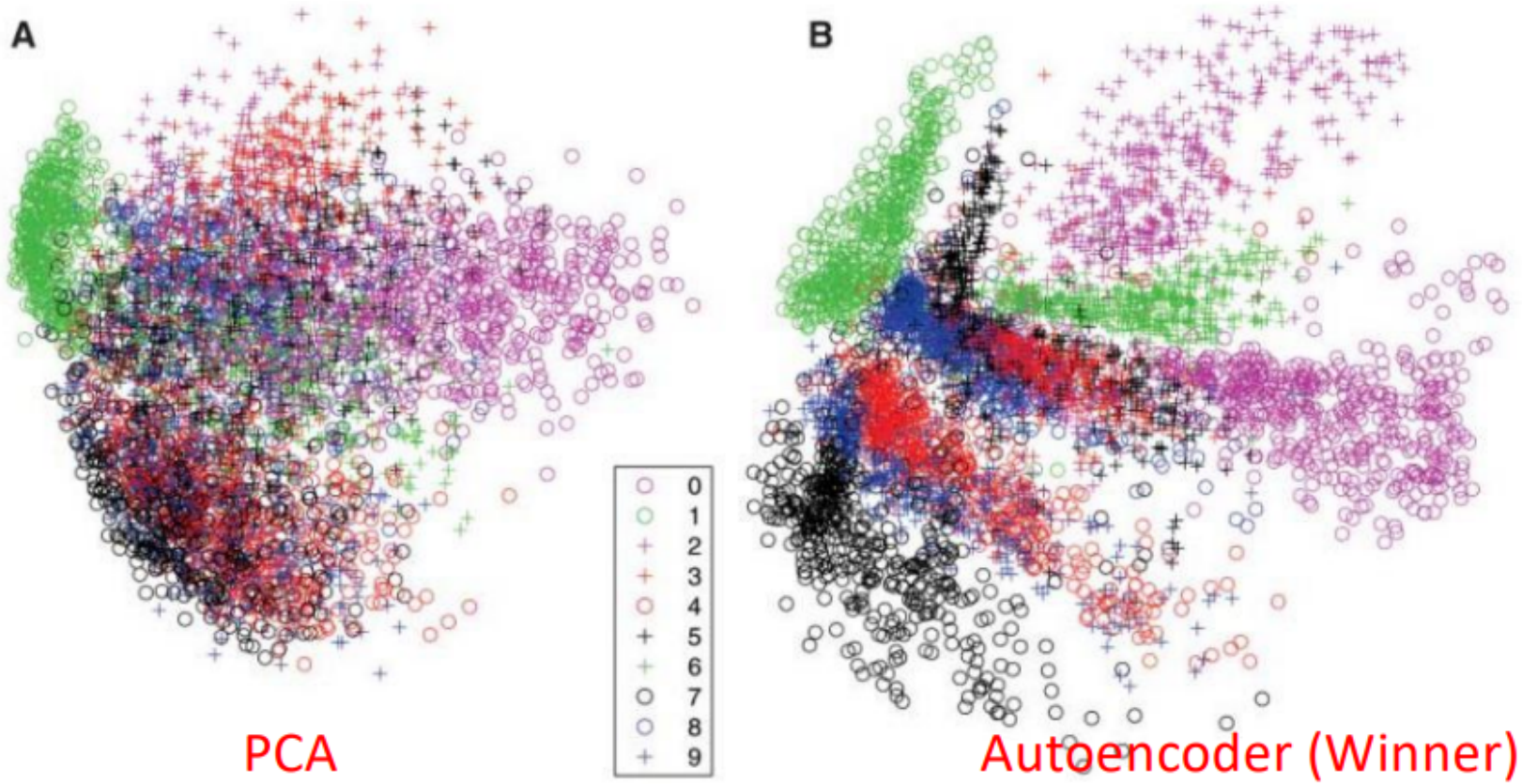


Figure 6: t-SNE visualization on MNIST digits dataset. PCA vs. Autoencoders. The image vector is projected into $\mathbb{R}^2$.

Super-Resolution

Autoencoders can be used to reconstruct high-resolution images from their low-resolution counterparts.

- Input: Low-resolution image.
- Output: High-resolution image.
- Often implemented with convolutional layers for spatial pattern learning.

Use Case: Enhancing medical images, satellite images, or upscaling low-resolution photos.

Autoencoders: Applications (cont.)

Figure 7: Image super-resolution using Autoencoders

Image Colorization

Autoencoders can learn to predict color information for grayscale images.

- Input: Grayscale image (1 channel).
- Output: Colorized image (3 channels — RGB).
- Requires learning semantic and contextual relationships to apply realistic colors.

Use Case: Restoring historical black-and-white photos.

Autoencoders: Applications (cont.)

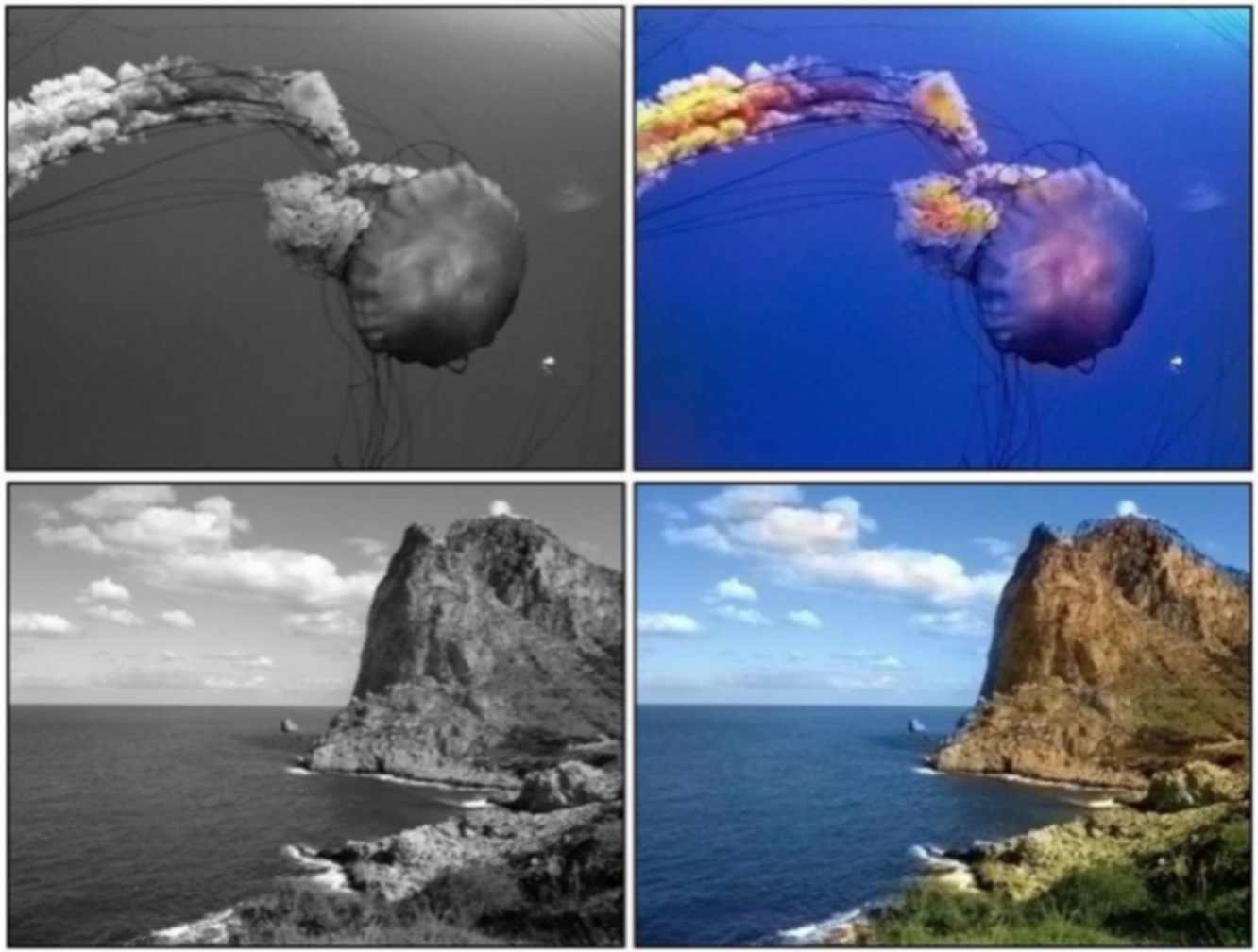


Figure 8: Image colorization using Autoencoders

Introduction: Variational Autoencoders (VAE)

- While autoencoders themselves have very low generative power, we will soon talk about a type of autoencoders called **Variational Autoencoders** which are specifically designed for generative modeling.
- **Variational Autoencoders (VAEs)**: Introduces a probabilistic framework where the encoder outputs parameters of a distribution (typically Gaussian). Sampling from this distribution and decoding allows generation of new data.
- Other use cases of Autoencoders include:
  - Data encoding and dimensionality reduction
  - Image denoising and super-resolution
  - Image completion
  - Image colorization

References

Reference Slides

- Fei-Fei Li "Generative Deep Learning" CS231
- Murtaza Taj "Deep Learning" CS437

Credits

Dr. Prashant Aparajeya

Computer Vision Scientist — Director(AISimply Ltd)

p.aparajeya@aisimply.uk

This project benefited from external collaboration, and we acknowledge their contribution with gratitude.