Understanding Autoencoders

Intuition to Applications

The Magic Wardrobe

The Problem - A Messy Pile of Clothes

Imagine all your clothes **dumped** in a pile.

You need a particular shirt, but it’s **lost** in the chaos.

Your stylist, **Brian**, is **frustrated** with finding the right piece quickly.

Organizing the Wardrobe = Learning an Embedding Space

Brian asks you to arrange clothes in an **infinite wardrobe.**

Similar clothes are placed close to each other.

Each item gets a **location** in this space.

Brian Learns to Reconstruct from Coordinates

Now you just give Brian a **location**.

He can recreate the exact item from scratch using his **sewing machine**.

You and Brian have "**learned**" a shared understanding of the space.

The Magic Begins - Generating New Clothes

You point to a location with no clothes.

Brian stitches a **new, never-seen-before item.**

It’s not perfect, but it’s original.

Mapping the Analogy to Machine Learning Concepts

Clothes = Data

Wardrobe = Latent Space

Coordinates = Embeddings/Vectors

Brian = Generator Model (e.g., Decoder, Generator)

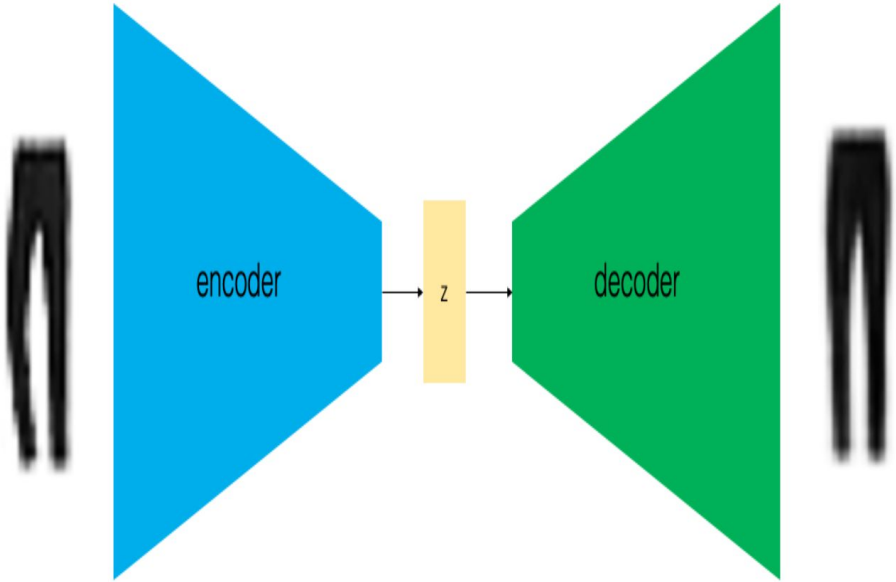
Sewing Machine = Learned Decoding Mechanism

Autoencoders: A Two-Part Neural Network

Comprised of an **Encoder and a Decoder**

Encoder **compresses** input (like images) into a smaller representation

Decoder reconstructs the original data from this compressed form



A Skilled Sketch Artist

Imagine describing a photo to an artist in just a few words

The artist draws the image based only on your description

Better descriptions (embeddings) = more accurate drawings

What’s the point of recreating something we already have?

Seems strange to reconstruct existing data…

But this forces the model to learn key features of the data

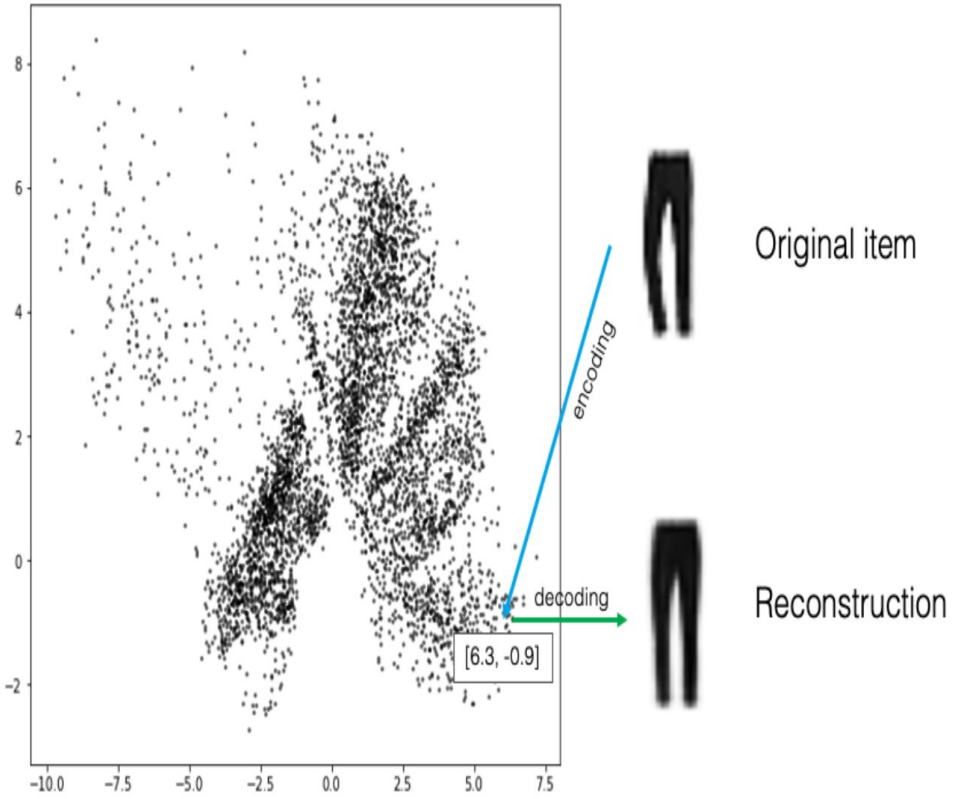
The real power lies in the latent space (embedding)

The Embedding (Latent Space)

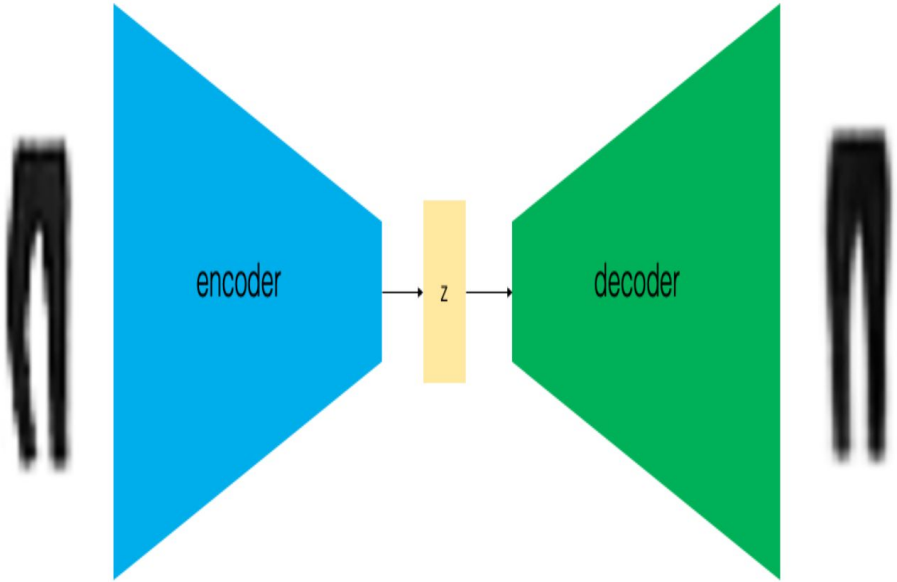
A compressed representation of the input

Captures essential features (e.g., shape, color, orientation)

Each input gets mapped to a point in this “**latent space**”



The Decoder: Rebuilding from the Essence



Takes a **latent vector (z)**

Learns how to **reconstruct** the input image

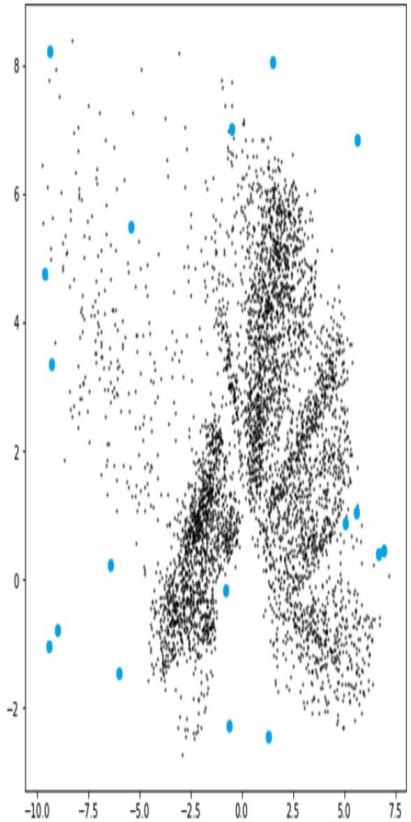
Quality of reconstruction shows how well the model understands

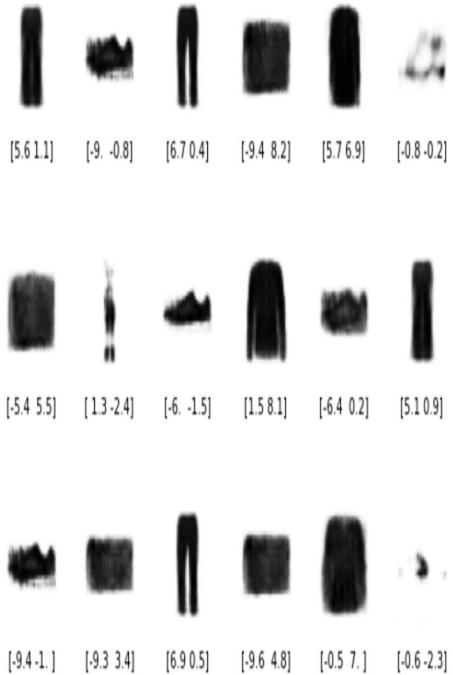
Creating Novel Data From Latent Space

We can choose any point in the latent space

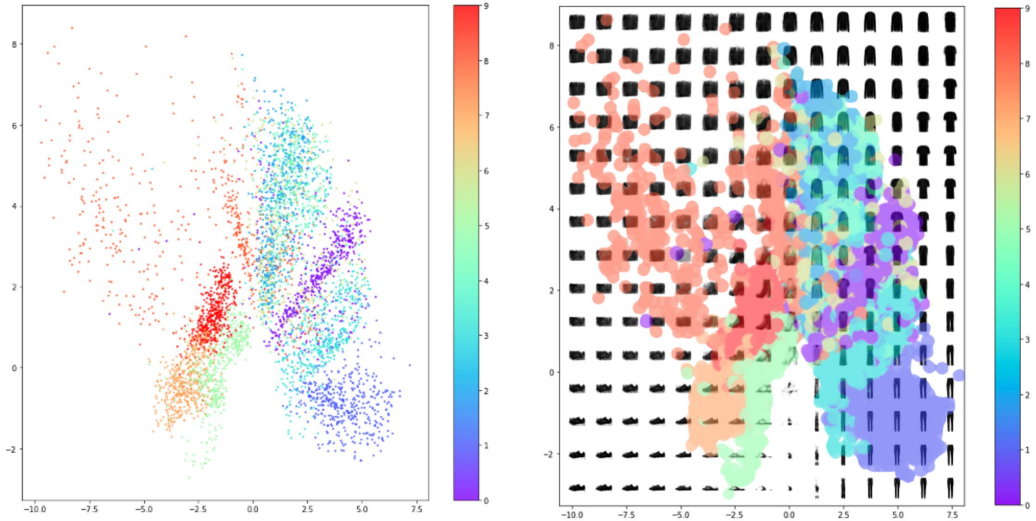
Passing it through the decoder gives us a new image

This allows for controlled, creative data generation





Visualizing the Latent Space



2D space is easy to plot and interpret

Each image = a point

Helps us understand structure and similarity between inputs

Real Autoencoders Use Higher Dimensions

Real-world problems require **10s or 100s of dimensions**

More dimensions = more expressive power

Still, the idea remains the same: **encode → compress → decode**

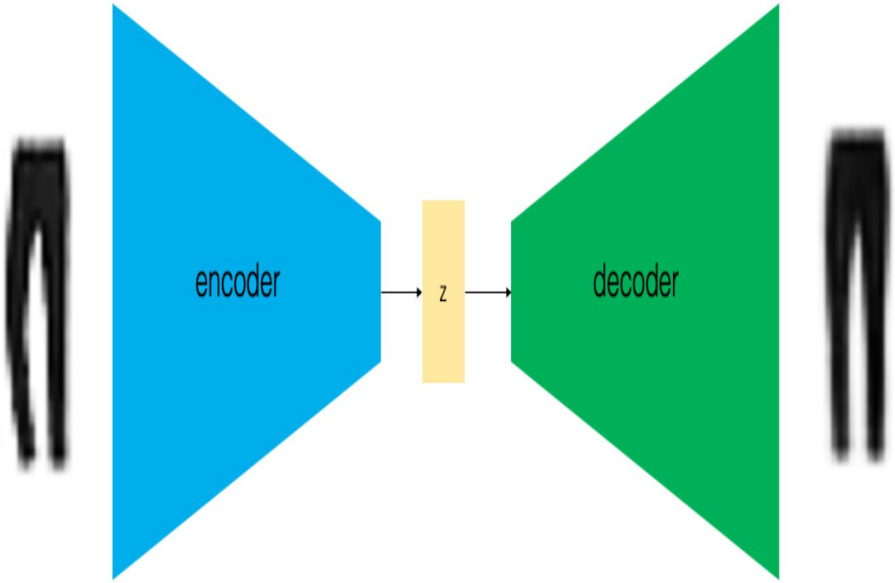
Reconstruction Loss

Measures how well the autoencoder can recreate the input from its latent representation.

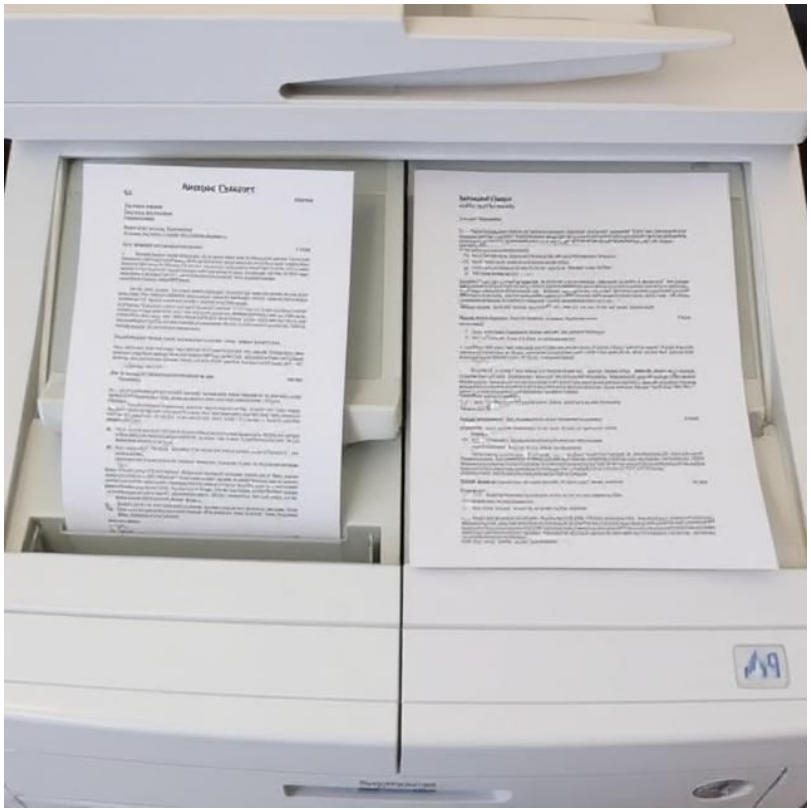
It’s the core objective for standard autoencoders.

Commonly used metrics:

- **Mean Squared Error** (MSE) for continuous data.
- **Binary Cross-Entropy** (BCE) for images with pixel values [0,1].



Intuition Behind Reconstruction Loss



Think of it like **photocopying** a document:

- Original document = Input
- Photocopy = Reconstruction
- Blurry/altered copy = High Loss
- Sharp, accurate copy = Low Loss

Mathematical View

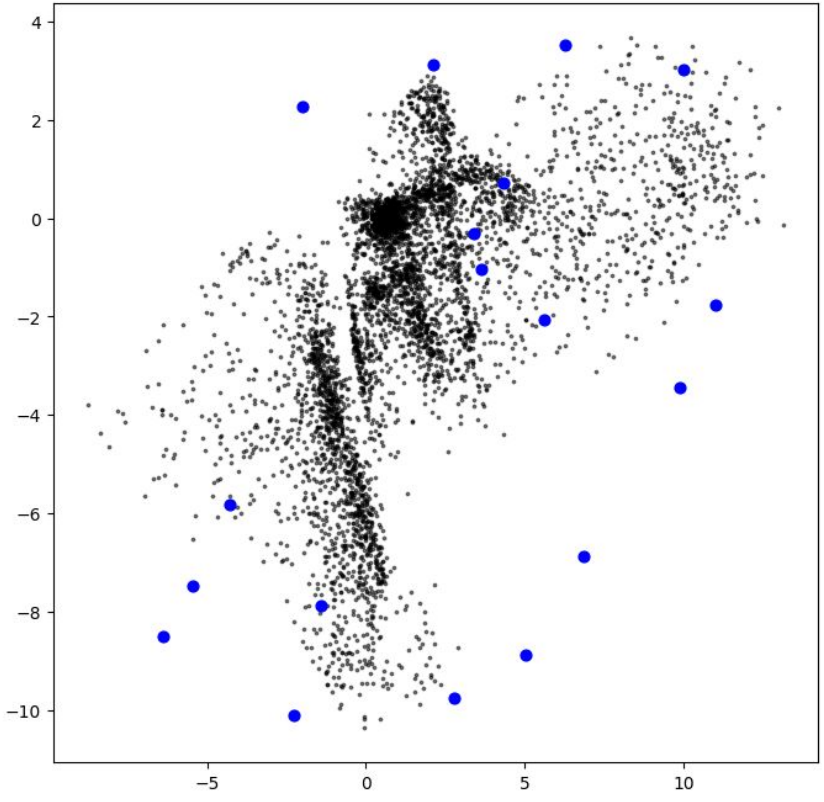
For input **x** and reconstruction **x'**:

- **MSE:** $L = \frac{1}{N} \sum (x - x')^2$
- **BCE:** $L = -\sum [x \log x' + (1 - x) \log (1 - x')]$

Goal: Minimize $L \rightarrow x' \approx x$

Why Reconstruction Loss Alone Isn’t Enough?

- Autoencoder **learns** to reproduce inputs well, but…
  - **May overfit training data.**
  - **Latent space may lack structure.**
- Leads to **blurry outputs** or poor generalization.



Wrapping It Up: The Power of Autoencoders

Autoencoders = Encoder + Decoder

Learn compressed representations of data

Enable reconstruction and generation of new data

Latent space is where insights and creativity lie

Quiz Time!

1. What are the two main components of an autoencoder, and what does each one do?
2. Why is the latent (embedding) space considered the most interesting part of an autoencoder?
3. What advantage does using a 2D latent space provide when teaching or visualizing autoencoders?
4. True or False: In practice, autoencoders typically use a 2D latent space because it's more expressive for complex data.
5. What happens when we give the decoder a point in the latent space that was never seen during training? What does this demonstrate?