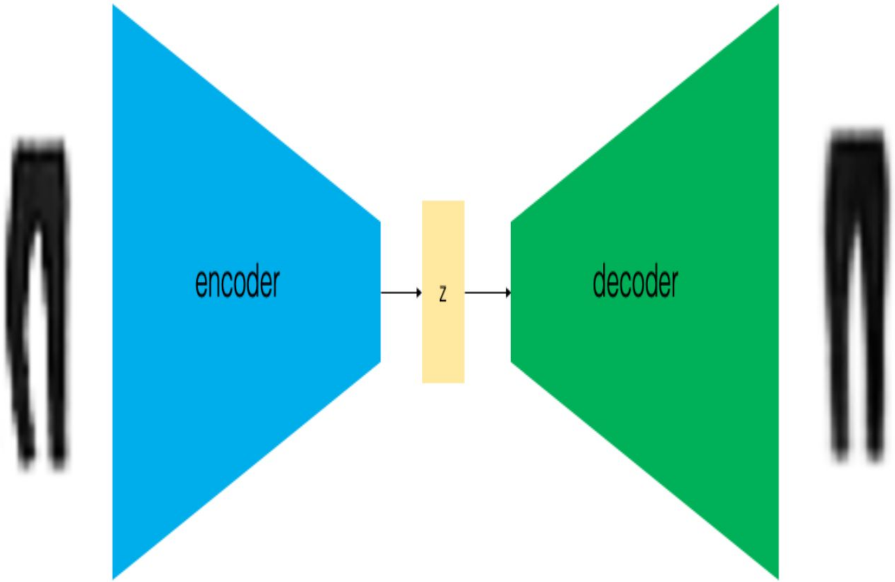
Autoencoder Numerical



Encoder Setup

1. Input

- A simple grayscale 1D image (1 channel), size: `1 × 4`

$$x = [1.0, 2.0, 3.0, 4.0]$$

2. Encoder

- 1 **convolutional layer:**
  - Filter size: 2
  - Stride: 1
  - No padding
  - 1 filter only
- Activation: None (for simplicity)
- Output shape: `1 × (4 − 2 + 1) = 1 × 3`

$$W_{conv} = [0.5, -0.5] \quad \text{(our kernel)}$$

$$b = 0.0$$

3. Latent Layer

- Fully connected (Linear) → Output dimension = 2

$$z = W_{fc} \cdot \text{flattened feature map} + b$$

Let's define:

$$W_{fc} = \begin{bmatrix} 1 & 0 & -1 \\ 0.5 & 0.5 & 0.5 \end{bmatrix}, \quad b = [0, 0]$$

Let’s Solve

🖼️ Step 1: Input

$$x = [1.0, 2.0, 3.0, 4.0]$$

🧠 Step 2: Convolution Layer (Encoder)

Apply the kernel:

$$W_{conv} = [0.5, -0.5]$$

We slide over the input in steps of 1:

- Step 1: $0.5 \cdot 1.0 + (-0.5) \cdot 2.0 = 0.5 - 1.0 = -0.5$
- Step 2: $0.5 \cdot 2.0 + (-0.5) \cdot 3.0 = 1.0 - 1.5 = -0.5$
- Step 3: $0.5 \cdot 3.0 + (-0.5) \cdot 4.0 = 1.5 - 2.0 = -0.5$

$$\text{Conv Output} = [-0.5, -0.5, -0.5]$$

📦 Step 3: Flatten and Fully Connected Layer

We flatten conv output:

$$\text{Input to FC} = [-0.5, -0.5, -0.5]$$

Now apply the FC layer:

$$z = W_{fc} \cdot [-0.5, -0.5, -0.5]^T$$

$$z_1 = 1 \cdot (-0.5) + 0 \cdot (-0.5) + (-1) \cdot (-0.5) = -0.5 + 0 + 0.5 = 0$$

$$z_2 = 0.5 \cdot (-0.5) + 0.5 \cdot (-0.5) + 0.5 \cdot (-0.5) = -0.25 - 0.25 - 0.25 = -0.75$$

✅ Latent vector z = [0, -0.75]

✅ Summary Table

Step	Operation	Output
Input	Raw Image	`[1.0, 2.0, 3.0, 4.0]`
Conv (W = [0.5, -0.5])	Conv output	`[-0.5, -0.5, -0.5]`
Flatten		`[-0.5, -0.5, -0.5]`
FC Layer	Latent vector	`[0, -0.75]`

Decoder Setup

📦 Step 1: Fully Connected (Latent → Feature Map)

We map the 2D latent vector to a 3D feature map — matching the size of the encoder output `[-0.5, -0.5, -0.5]`.

Let's define:

$$W_{fc\_decoder} = \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \\ -1.0 & 0.5 \end{bmatrix}, \quad b = [0, 0, 0]$$

Apply:

$$z = \begin{bmatrix} 0 \\ -0.75 \end{bmatrix}$$

Compute output:

- $f_1 = 1.0 \cdot 0 + 0.0 \cdot (-0.75) = 0$
- $f_2 = 0.0 \cdot 0 + 1.0 \cdot (-0.75) = -0.75$
- $f_3 = -1.0 \cdot 0 + 0.5 \cdot (-0.75) = -0.375$

✅ Decoder FC Output = $[0, -0.75, -0.375]$

🔁 Step 2: Transpose Convolution (3 → 4)

We now upsample from size 3 → 4 using a **Transpose Convolution** with:

- Filter: `[0.5, -0.5]` (same shape as encoder, for symmetry)
- Stride = 1
- No padding

We apply **transpose convolution**, which essentially flips the kernel and slides it over the input.

To manually compute transpose convolution, we **insert 0s** between input elements (for stride > 1), but since our stride = 1, this is simpler.

We'll slide the kernel `[0.5, -0.5]` across the decoder FC output $[0, -0.75, -0.375]$ to get an output of size 4:

Manual calculation:

Let:

$$x = [0, -0.75, -0.375] \quad \text{and} \quad w = [0.5, -0.5]$$

Then:

- Output[0] = $0 \cdot 0.5 = 0$
- Output[1] = $0 \cdot (-0.5) + (-0.75) \cdot 0.5 = -0.375$
- Output[2] = $-0.75 \cdot (-0.5) + (-0.375) \cdot 0.5 = 0.375 - 0.1875 = 0.1875$
- Output[3] = $-0.375 \cdot (-0.5) = 0.1875$

✅ **Reconstructed image =** `[0, -0.375, 0.1875, 0.1875]`

🧮 Step 3: Compute Reconstruction Loss

Let's compare the reconstructed image with original input `[1.0, 2.0, 3.0, 4.0]` using **Mean Squared Error (MSE)**:

$$\text{MSE} = \frac{1}{4}\sum_{i=1}^{4}(\hat{x}_i - x_i)^2$$

$$= \frac{1}{4}\left[(0-1)^2 + (-0.375-2)^2 + (0.1875-3)^2 + (0.1875-4)^2\right]$$

$$= \frac{1}{4}[1 + 5.6406 + 7.8976 + 14.496] = \frac{1}{4}\cdot 29.0342 = \boxed{7.2586}$$

✅ Final Summary

Step	Operation	Output
Latent Vector	Input	`[0, -0.75]`
FC Layer	Expand	`[0, -0.75, -0.375]`
Transposed Conv	Reconstruct	`[0, -0.375, 0.1875, 0.1875]`
Compare with Original	MSE Loss	`7.26`