

Machine Learning (CE 40717)

Fall 2024

Ali Sharifi-Zarchi

CE Department
Sharif University of Technology

December 15, 2025



1 Autoencoder Fundamentals

2 Types of Autoencoders

3 References

1 Autoencoder Fundamentals

② Types of Autoencoders

3 References

Unlabeled Data Usage in Deep Learning



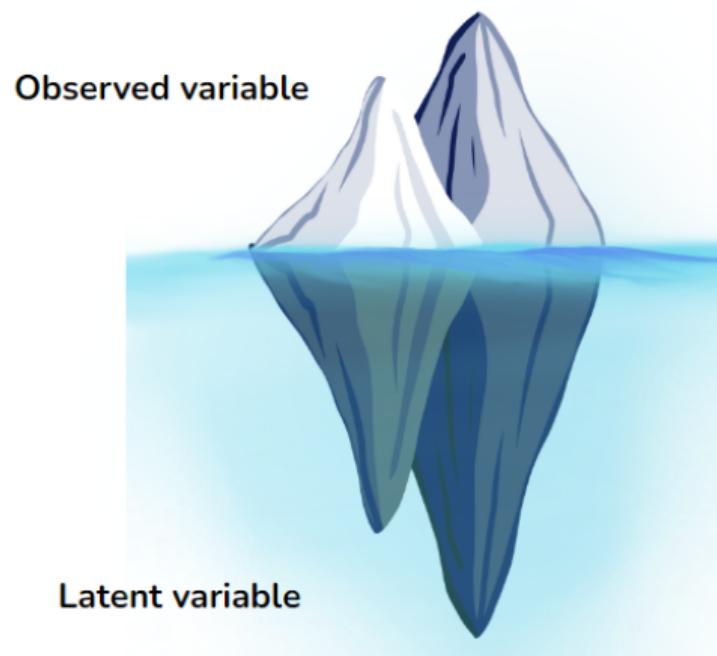
How can we use unlabeled data in our deep neural networks?

Image: Source

What Is Latent Variable?

- **Definition:** Latent variables are variables that are not directly observed but are inferred from other variables that are observed (measured).
 - **Origin:** Derived from the Latin word *latēre* meaning "hidden".
 - **Examples:**
 - Personality traits (in psychology)
 - Economic factors like inflation (in economics)
 - Topic of a document (in natural language processing)

What Is Latent Variable?



Latent Variables in Deep Learning

We can train **deep neural networks** using **unlabeled data** to infer **latent variables** in order to:

- Compress data
 - Generate new data
 - Denoise Data
 - Using latent variables as features for various tasks like classification
 - ...

Autoencoder: Background

Using latent variables as a foundation, an unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data.



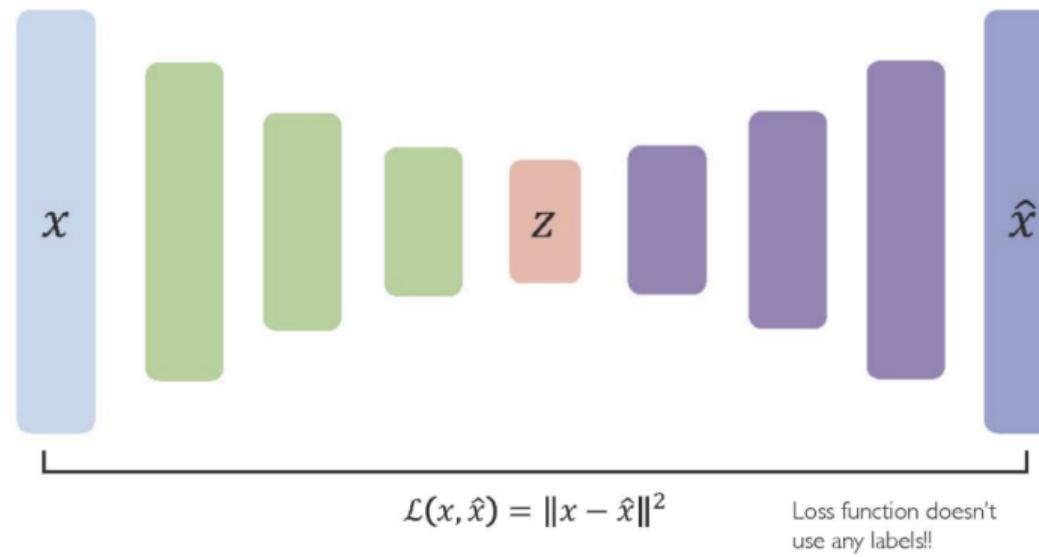
Why do we care about a low-dimensional z ?

"Encoder" learns mapping from the data, x , to a low-dimensional latent space, z

Autoencoders: Background

How can we learn this latent space?

Train the model to use these features to **reconstruct the desired (usually original) data.**



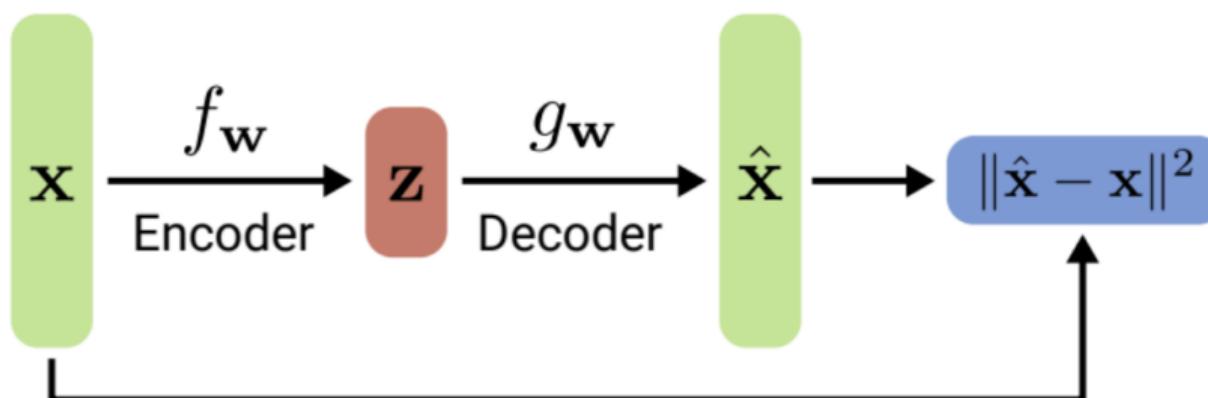
"Decoder" learns mapping back from latent z , to a reconstructed observation, x

Autoencoders

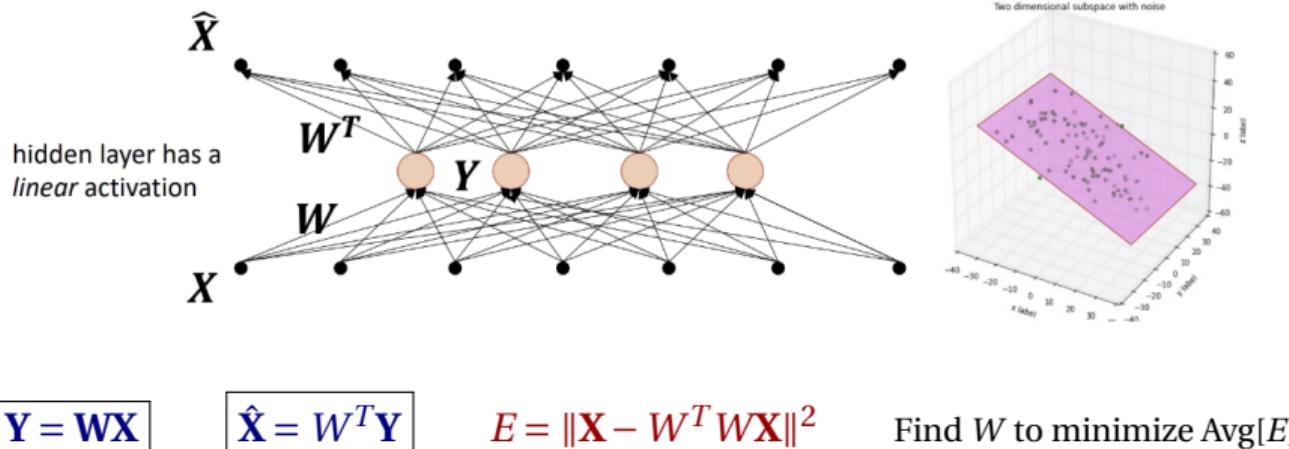
① These models map between **observation space $\mathbf{x} \in \mathbb{R}^D$** and **latent space $\mathbf{z} \in \mathbb{R}^Q$** :

- **Encoder** $f_w : \mathbf{x} \mapsto \mathbf{z}$ The "Analysis" net which computes the hidden representation
- **Decoder** $g_w : \mathbf{z} \mapsto \hat{\mathbf{x}}$ The "Synthesis" which recomposes the data from the hidden representation

② Models are linear or non-linear, deterministic or stochastic, with/without encoder.



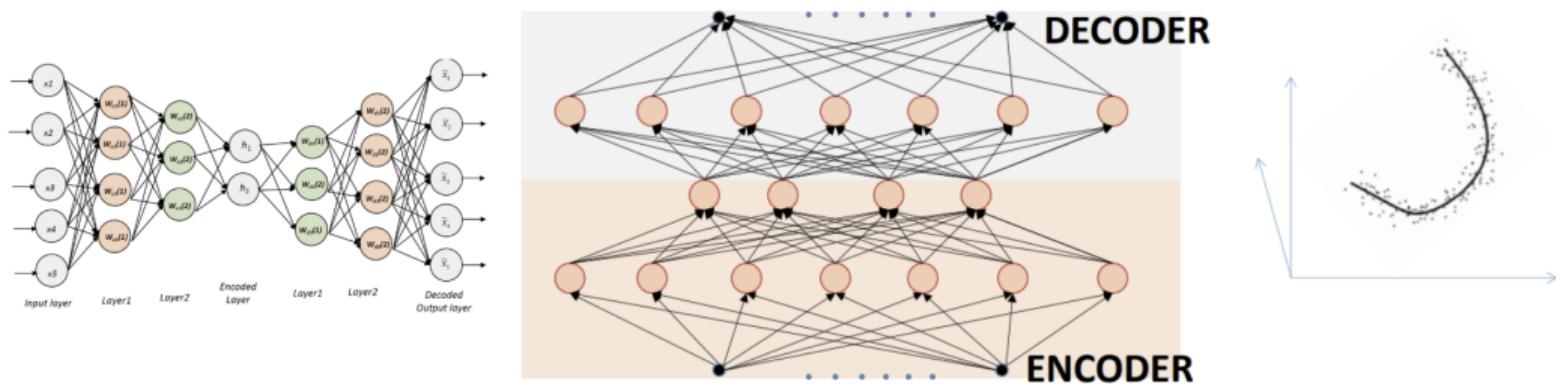
Autoencoder Without a Non-linearity



- This is similar to PCA
 - The output of the hidden layer will be in the principal subspace
 - Even if the recombination weights are different from the "analysis" weights

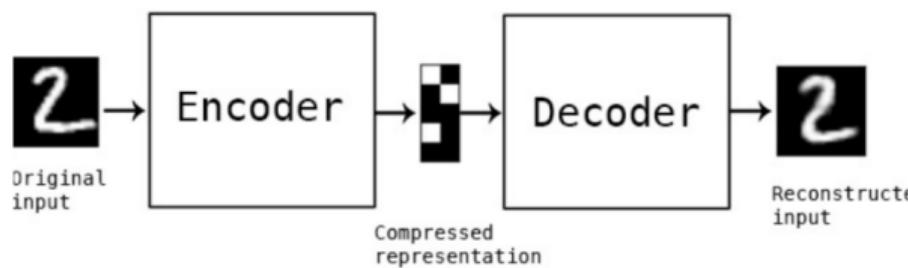
“Deep” AE

- **Deep nonlinear autoencoders** learn to project the data, not onto a subspace, but onto a nonlinear **manifold**.
 - This manifold is the **image of the decoder**.
 - This is a kind of **nonlinear dimensionality reduction**.



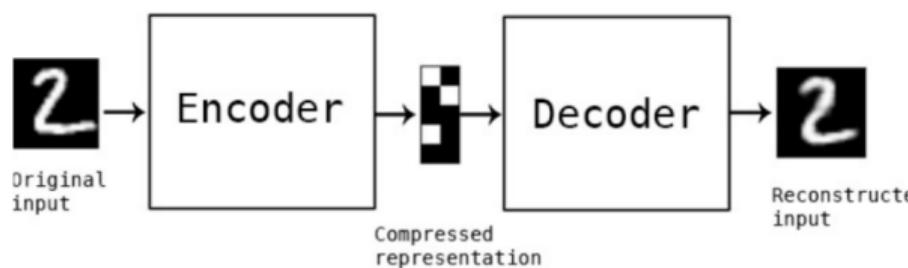
AE vs PCA

As we said, **PCA** can be described as



$$\begin{aligned} & \min_W \sum \|\hat{x} - x\|^2 \\ & W^T W = I \\ & \min_W \sum \|W^T W x - x\|^2 \end{aligned}$$

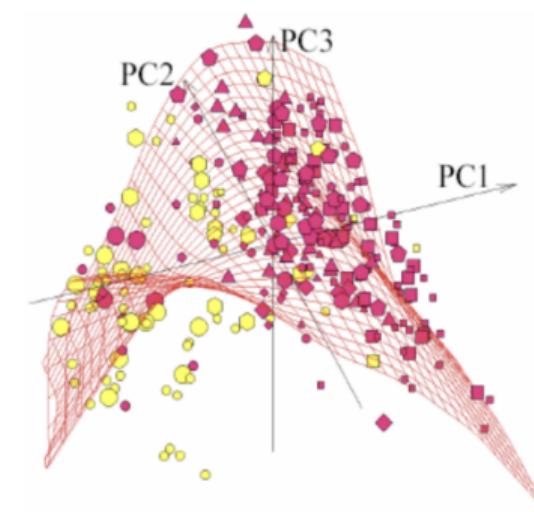
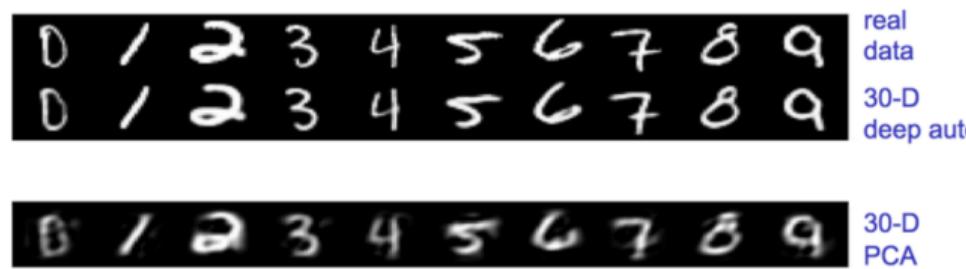
And also **Autoencoders** can be thought of as a **non-linear PCA**.



$$\begin{aligned} & \min_{h,g} \sum \|\hat{x} - x\|^2 \\ & \min_{h,g} \sum \|g(f(x)) - x\|^2 \end{aligned}$$

AE vs PCA

Nonlinear autoencoders can learn more powerful codes for a given dimensionality, compared with linear autoencoders (PCA).



Autoencoder Key Characteristics

- **Specialized for Specific Data Type**

- Autoencoders excel at recognizing patterns in the type of data they were trained on, but may fail on different data types.
- Example: A model trained on animal images will not perform well when applied to landscapes.

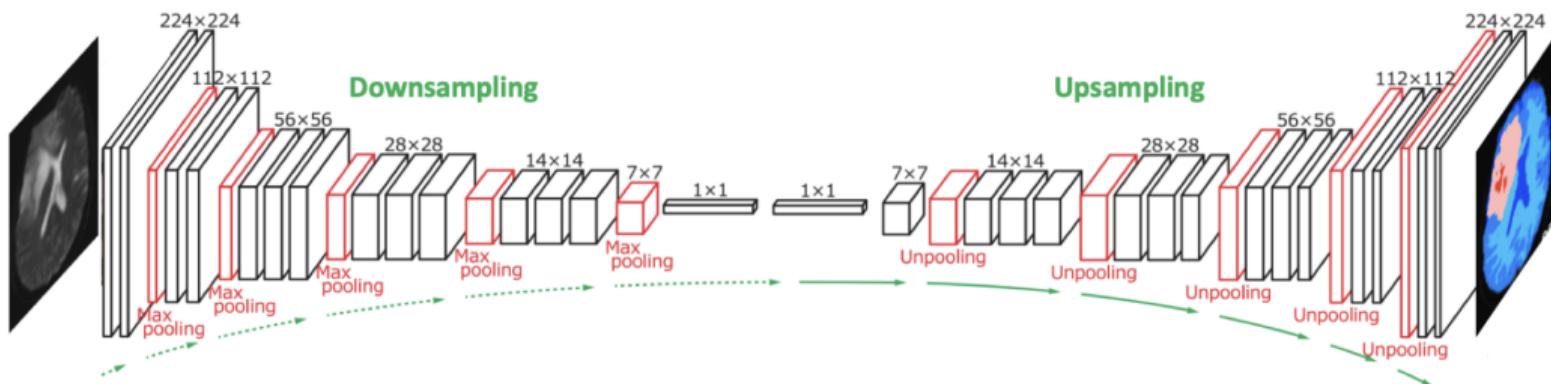
- **Tailored Compression vs. General Algorithms**

- Unlike generic compression techniques (MP3, JPEG), which apply universal rules, autoencoders customize their approach to fit the training data structure.

- **Loss of Detail in Output**

- Autoencoders don't retain all the original information, leading to some degradation in the reconstructed data.
- This makes them "lossy" – similar to formats like MP3 or JPEG, which prioritize size over perfect fidelity.

Fully Convolutional AE

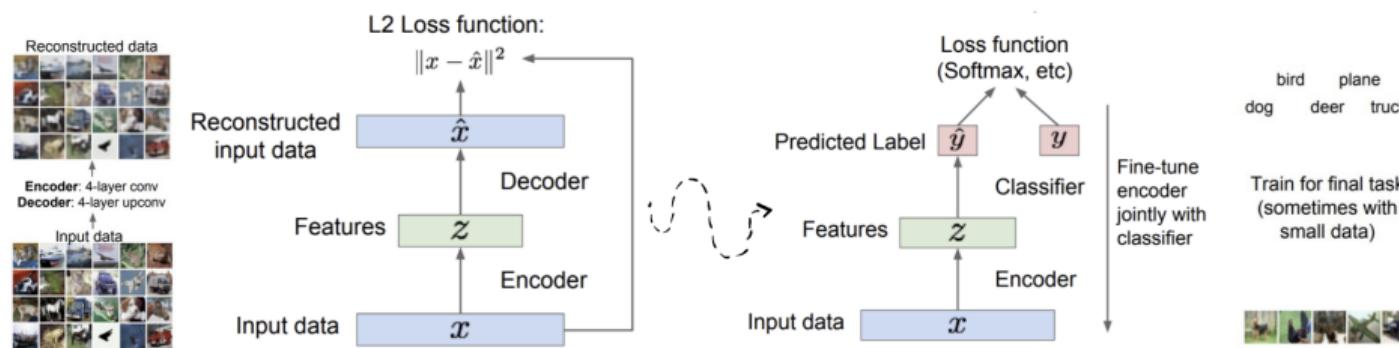


We will discuss more about "**Downsampling**" and "**Upsampling**" in the next section

[Image from this link](#)

Autoencoders As Pretrained Models

- **Feature Extraction:** Encoder **learns** data patterns and **compresses** them into useful features.
- **Supervised Model Initialization:** Use the **pretrained encoder** to initialize a supervised learning model.
- **Fine-Tuning:** Train the encoder and classifier **together** to improve performance.
- **Useful with Small Data:** Effective when training on **small datasets** by leveraging pretrained features.



AE Performance & Bottleneck Size (MNIST Example)

- **Reconstruction Error and Bottleneck Size:**
 - A good autoencoder should minimize reconstruction error.
 - The error depends heavily on the size of the z (the dimension of the latent space).
- **Experiment Setup:**
 - CNN-based AE trained on MNIST with different bottleneck sizes:
 - $(2, 1, 1), (4, 1, 1) \text{ & } (32, 1, 1)$

AE Performance & Bottleneck Size (MNIST Example) - Results

• **Results:**

- With $d = 2$: Reconstruction is still good; a classifier could identify digits decently.
- With $d = 32$: Reconstruction is nearly perfect; despite a compression ratio of ~ 0.04 (from 32 to 28x28).

 X (original samples)

7 2 1 0 4 1 4 9 5 9 0 6
9 0 1 5 9 7 3 4 9 6 6 5
4 0 7 4 0 1 3 1 3 4 7 2

 $g \circ f(X)$ (CNN, $d = 2$)

7 2 1 0 9 1 9 9 8 9 0 6
9 0 1 5 9 7 8 9 9 6 6 5
9 0 7 9 0 1 5 1 5 9 7 2

 $g \circ f(X)$ (CNN, $d = 4$)

7 2 1 0 4 1 4 9 9 9 0 6
9 0 1 5 9 7 3 4 9 6 6 5
4 0 7 4 0 1 3 1 3 0 7 2

 $g \circ f(X)$ (CNN, $d = 32$)

7 2 1 0 4 1 4 9 5 9 0 6
9 0 1 5 9 7 3 4 9 6 6 5
4 0 7 4 0 1 3 1 3 4 7 2

1 Autoencoder Fundamentals

2 Types of Autoencoders

3 References

AE's: Undercomplete Autoencoders

- ① An autoencoder whose code dimension is less than the input dimension is called **undercomplete**.
- ② Learning an undercomplete representation forces the autoencoder to capture the most salient features of the training data.
- ③ The learning process is described simply as minimizing a loss function

$$L(x, g(f(x)))$$

where L is a loss function penalizing $g(f(x))$ for being dissimilar from x , such as the mean squared error.

AE's: Overcomplete Autoencoders

- ① consider encoder f and decoder g .

$$X \in \mathbb{R}^d \quad h \in \mathbb{R}^k$$

- ② When $k > d$, the autoencoder is called **Overcomplete Autoencoders**.
- ③ There are other ways we can constrain the reconstruction of an autoencoder than to impose a hidden layer of smaller dimension than the input.
- ④ Regularized Autoencoders use a loss function that encourages the model to have some properties besides reproducing inputs.
 - Sparsity representation (Sparse Autoencoders)
 - Smallness of derivative of representation (Contractive Autoencoders)
 - Robustness to noise or to missing inputs (Denoising Autoencoders)

AE's: Sparse Autoencoders (Intuition)

Idea: Can we describe the input with a small set of “*attributes*”?

This might be a more **compressed** and **structured** representation.

1. NOT structured - “dense”: most values non-zero



Pixel (0,0): #FE057D
Pixel (0,1): #FD0263
Pixel (0,2): #E1065F

Aside:

This idea originated in neuroscience, where researchers believe that the brain uses **sparse** representations (see “sparse coding”).

Idea: “Sparse” representations are going to be more structured!

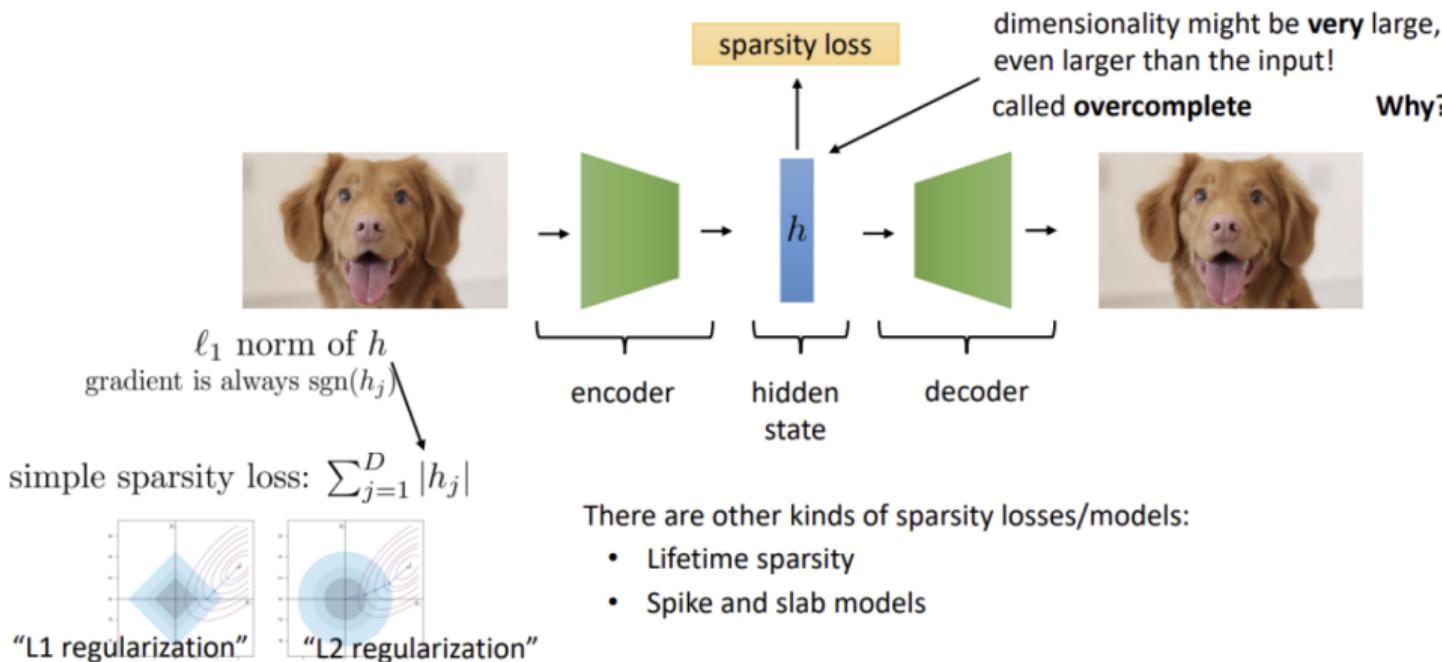
2. Very structured! - “sparse”: most values are zero



has_ears: 1
has_wings: 0
has_wheels: 0

There are many possible “*attributes*,” & most images don’t have most of the attributes.

AE's: Sparse Autoencoder (Big Picture)



AE's: Sparse Autoencoders (Details)

- ① Sparse Autoencoders try to minimize the following function.

$$L(x, g(f(x))) + \Omega(h)$$

- ② The first term is loss for copying inputs.
③ The second term is sparsity penalty.
④ In general neural networks, we are trying to find the **maximum likelihood**: $p(x|\theta)$.
⑤ We often use $\log p(x|\theta)$ for simplification, from which we can get the loss function without regularization.
⑥ What about MAP (Maximum a posteriori)?

$$p(\theta|x) \propto p(x|\theta) \times p(\theta)$$

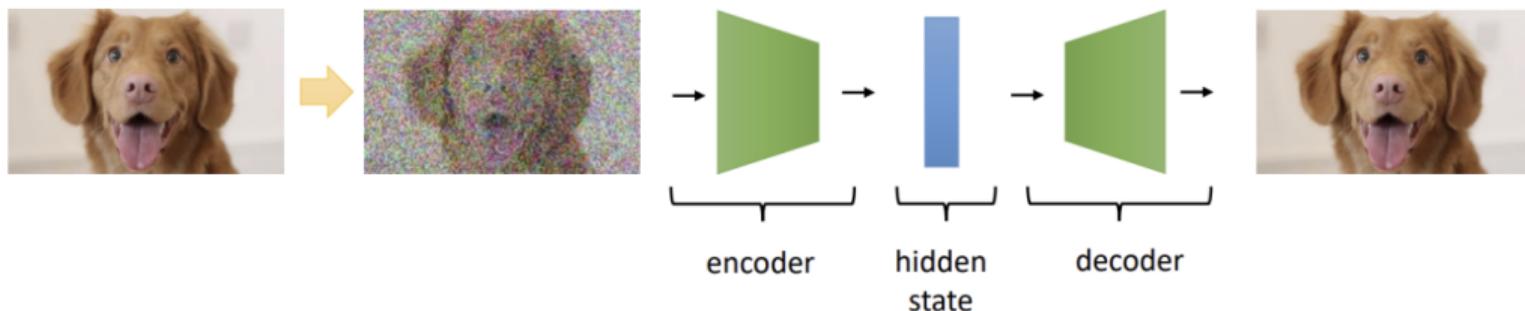
- ⑦ Maximizing the log of the above function yields:

$$\text{maximize } (\log p(x|\theta) + \log p(\theta))$$

- ⑧ The first term is the loss function, and the second term is the regularization penalty.

AE's: Denoising Autoencoders (Intuition)

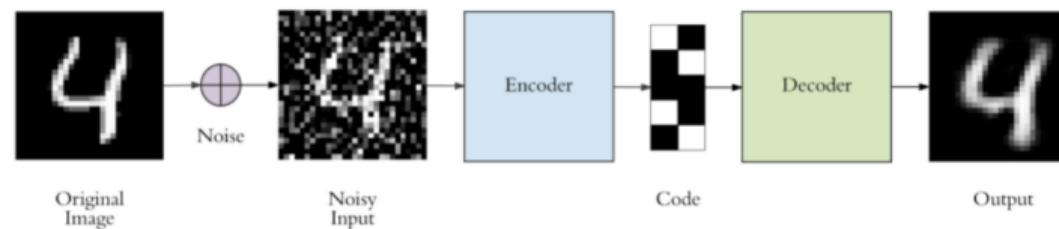
Idea: a good model that has learned meaningful structure should “fill in the blanks”



There are **many variants** on this basic idea, and this is one of the most widely used simple autoencoder designs.

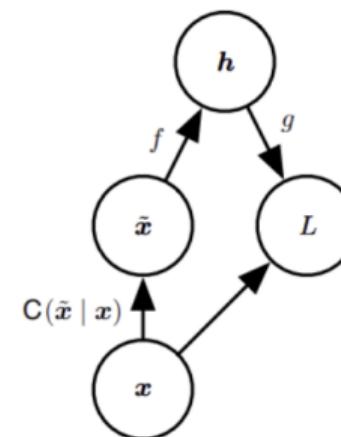
AE's: Denoising Autoencoders (Details)

- ① **The denoising autoencoder (DAE)** is an autoencoder that receives a corrupted data point as input and is trained to predict the original, uncorrupted data point as its output.
- ② **Traditional autoencoders minimize $L(x, g(f(x)))$**
 - L is a loss function penalizing $g(f(x))$ for being dissimilar from x , such as L_2 norm of difference: mean squared error.
- ③ **A DAE minimizes $L(x, g(f(\tilde{x}))$)**
 - \tilde{x} is a copy of x that is corrupted by some form of noise.
 - The autoencoder must undo this corruption rather than simply copying their input.



AE's: Denoising Autoencoders (Training Procedure)

- ① The DAE training procedure is

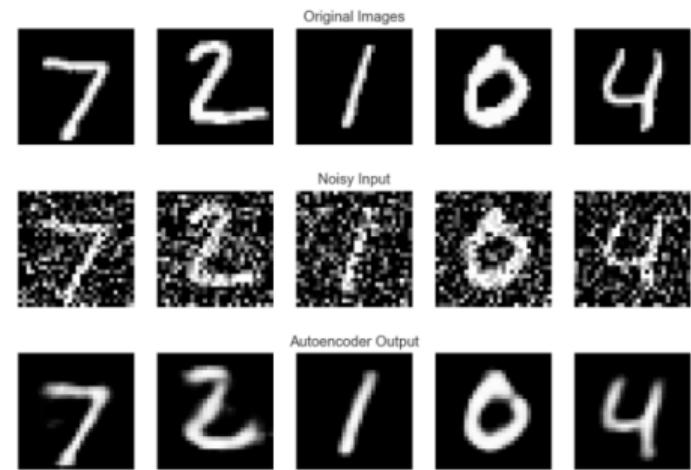
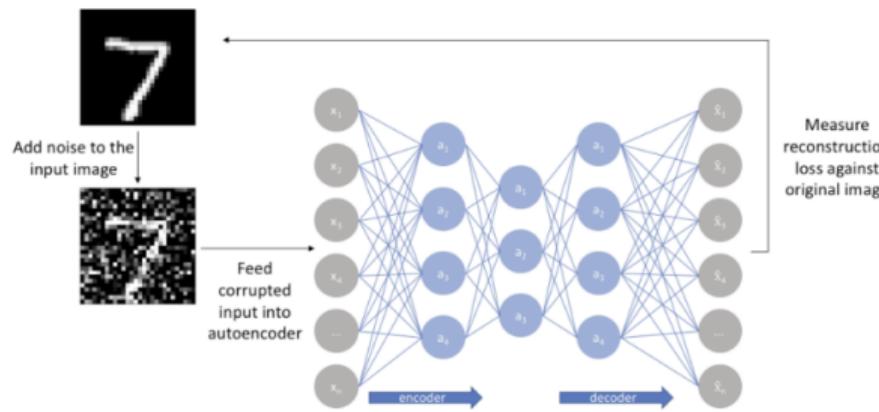


- ② We introduce a corruption process $C(\tilde{x}|x)$.

AE's: Denoising Autoencoders (Training Procedure)

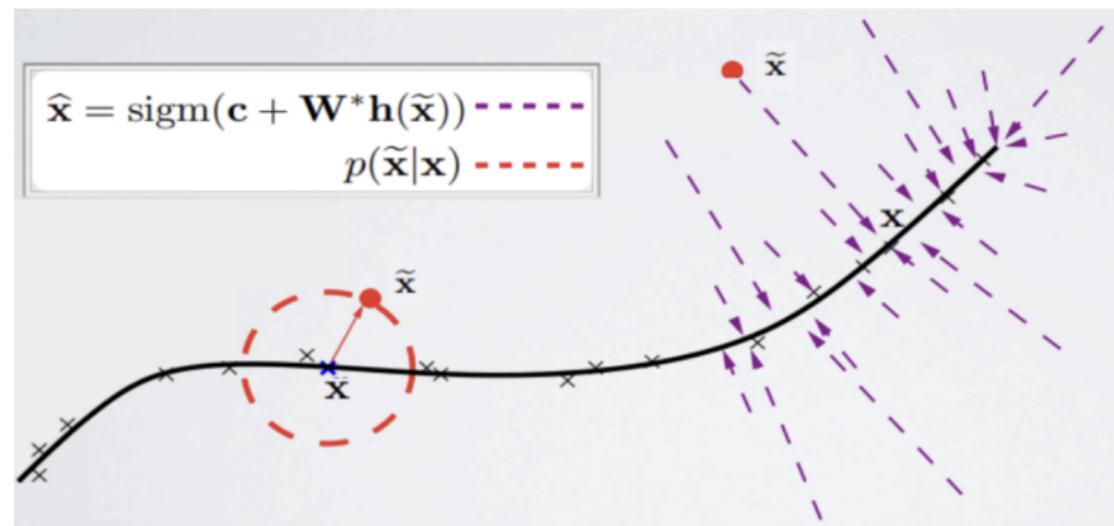
- ① We introduce a corruption process $C(\tilde{x}|x)$.
- ② The autoencoder then learns a reconstruction distribution $p_{\text{reconstruct}}(x|\tilde{x})$ estimated from training pairs (x, \tilde{x}) as follows:
 - Sample a training example x_i from the training data.
 - Sample a corrupted version \tilde{x}_i from $C(\tilde{x}_i|x = x_i)$.
 - Use (x, \tilde{x}) as a training example for estimating the autoencoder reconstruction distribution $p_{\text{reconstruct}}(x|\tilde{x}) = p_{\text{decoder}}(x|h)$ with h the output of encoder $f(\tilde{x})$ and p_{decoder} typically defined by a decoder $g(h)$.
 - Typically we can simply perform gradient-based approximate minimization on the negative log-likelihood $-\log p_{\text{decoder}}(x|h)$.

AE's: Denoising Autoencoders (Result On MNIST)



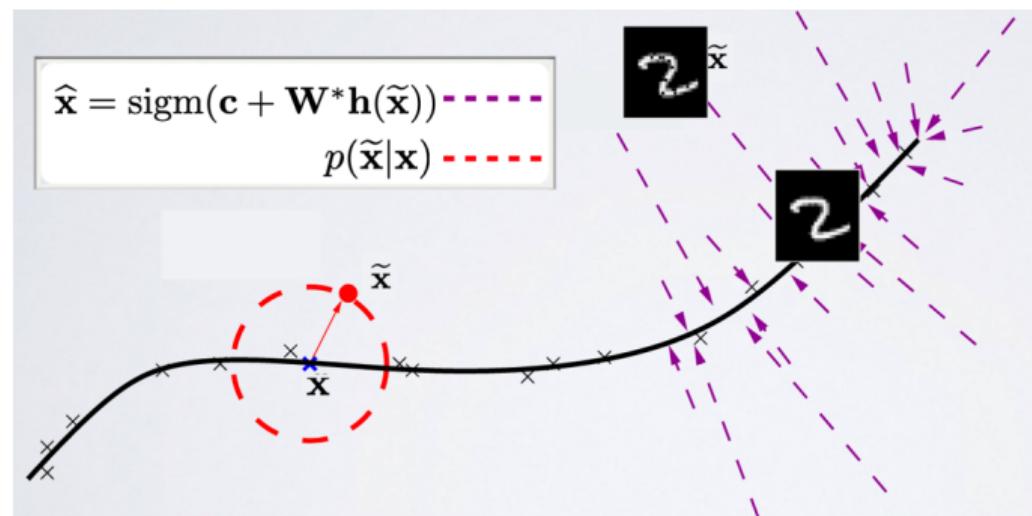
AE's: Denoising Autoencoders (It Isn't Lazy Network!)

- DAEs won't simply memorize the inputs and outputs (More robustness)
- Intuitively, a DAE learns a **projection** from a neighborhood of our training data back onto the training data



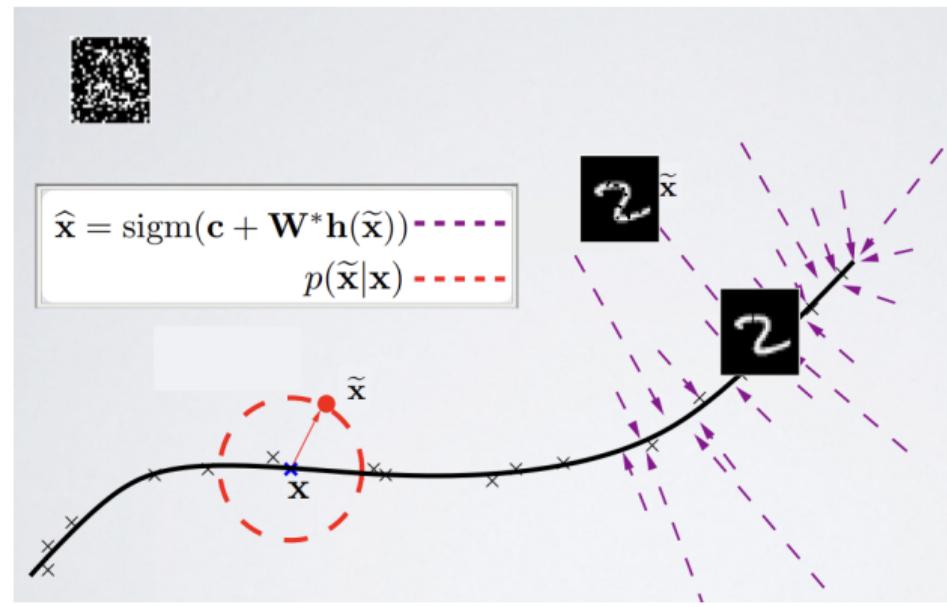
AE's: Denoising Autoencoders (It Isn't Lazy Network!)

- DAEs won't simply memorize the inputs and outputs (More robustness)
- Intuitively, a DAE learns a **projection** from a neighborhood of our training data back onto the training data



AE's: Denoising Autoencoders (It Isn't Lazy Network!)

- DAEs won't simply memorize the inputs and outputs (More robustness)
- Intuitively, a DAE learns a **projection** from a neighborhood of our training data back onto the training data

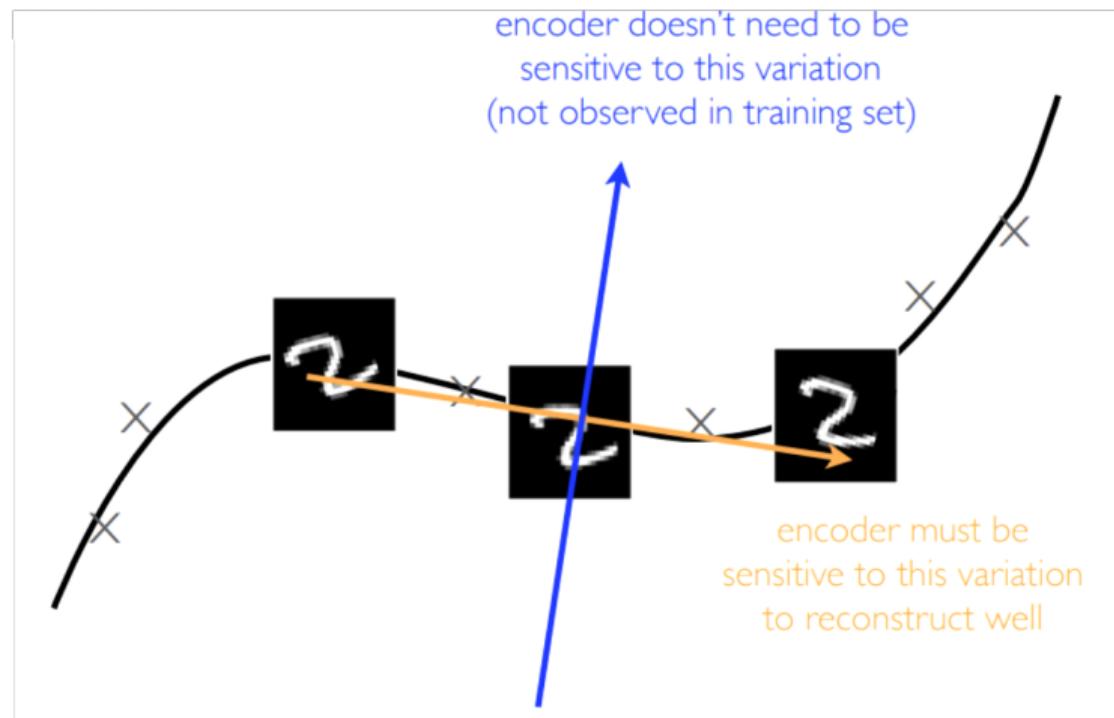


AE's: Contractive Autoencoder (Intuition)

- ① Contractive autoencoders are explicitly encouraged to learn a manifold through their loss function (Alain and Bengio 2014).
- ② **Desirable property:** Points close to each other in input space maintain that property in the latent space.
- ③ Method to avoid uninteresting solutions
- ④ Add an explicit term in the loss that penalizes that solution
- ⑤ We wish to extract features that only reflect variations observed in the training set
- ⑥ We would like to be invariant to other variations

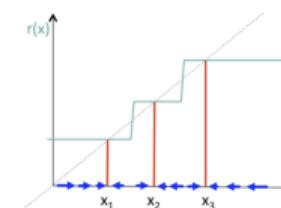


AE's: Contractive Autoencoder (Intuition)



AE's: Contractive Autoencoder (Details)

- 1 Contractive autoencoder has an explicit regularizer on $h = f(x)$, encouraging the derivatives of f to be as small as possible.
- 2 This will be true if $f(x) = h$ is continuous, has small derivatives.



- 3 We can use the **Frobenius Norm** of the **Jacobian Matrix** as a regularization term:

$$\Omega(f, x) = \lambda \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2$$

- 4 These autoencoders are called *contractive* because they contract the neighborhood of input space into a smaller, localized group in latent space.
- 5 **Exercise:** What is the difference between DAE and CAE?

AE's: Contractive Autoencoder (Details)

- New loss function:

$$\underbrace{l\left(f\left(x^{(t)}\right)\right)}_{\text{autoencoder reconstruction}} + \lambda \underbrace{\|\nabla_{x^{(t)}} h(x^{(t)})\|_F^2}_{\text{Jacobian of encoder}}$$

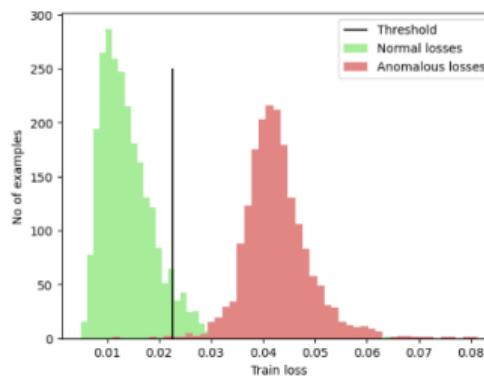
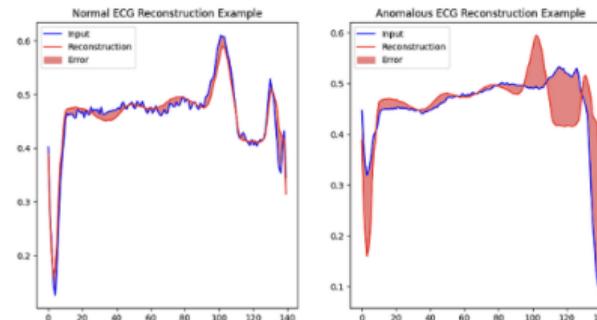
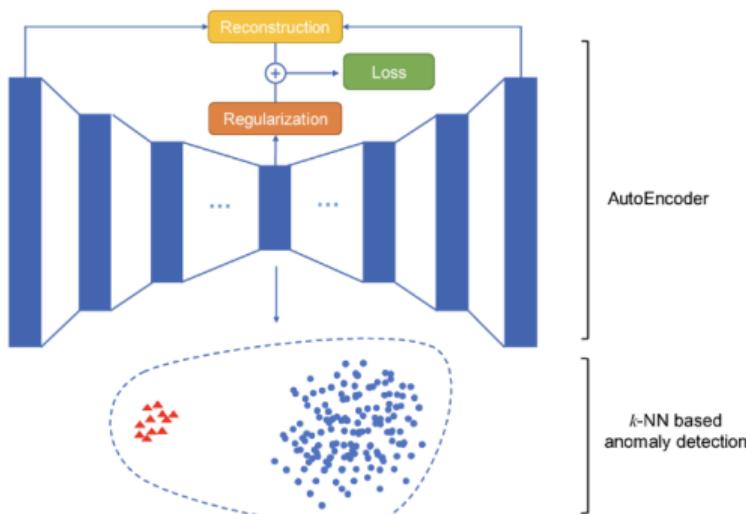
- where, for binary observations:

$$l\left(f\left(x^{(t)}\right)\right) = -\sum_k (x_k^{(t)} \log(\hat{x}_k^{(t)}) + (1-x_k^{(t)}) \log(1-\hat{x}_k^{(t)})) \quad \left. \begin{array}{l} \text{encoders keeps} \\ \text{good information} \end{array} \right\}$$

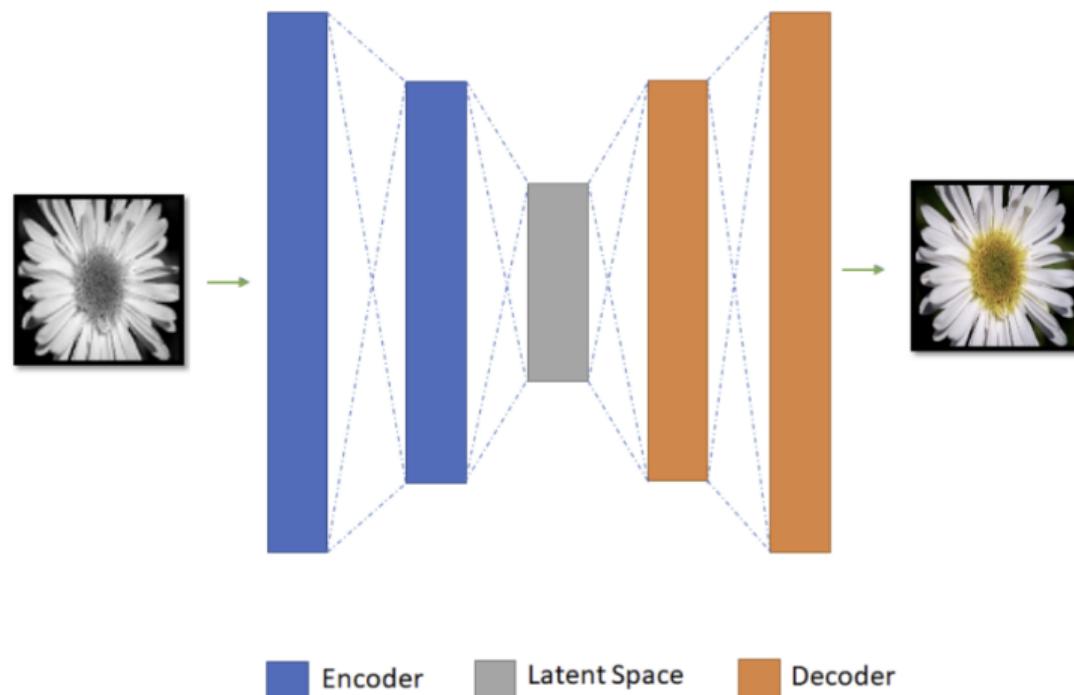
→ encoder keep only good information

$$\|\nabla_{x^{(t)}} h(x^{(t)})\|_F^2 = \sum_j \sum_k \left(\frac{\partial h(x^{(t)})_j}{\partial x_k^{(t)}} \right)^2 \quad \left. \begin{array}{l} \text{encoder throws} \\ \text{away all information} \end{array} \right\}$$

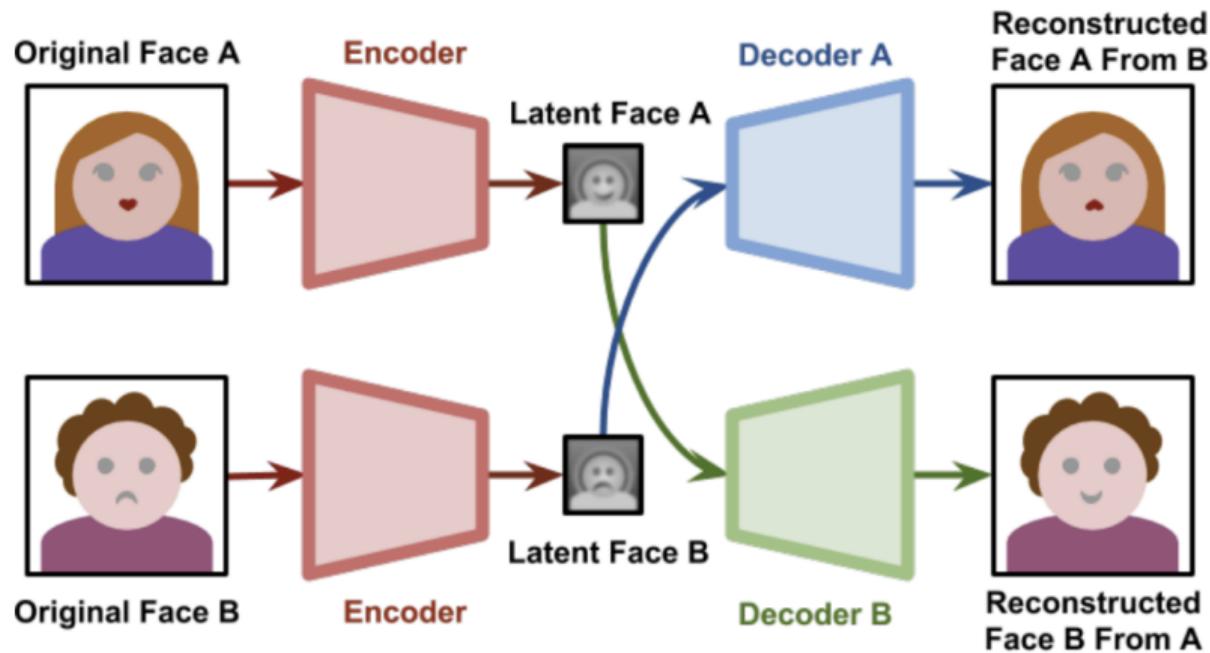
AE other Applications: Anomaly Detection



AE other Applications: Image Colorization



AE other Applications: Fake Laugh



1 Autoencoder Fundamentals

2 Types of Autoencoders

3 References

Slides by: Amirhossein Izadi

- F.-F. Li, J. Wu, and R. Gao, “CS231n” Lecture slides, 2024, Stanford
- A. Amini, “6s191” Lecture slides, 2024, MIT.
- M. Soleymani, “Deep Learning” Lecture slides, 2024, Sharif University of Technology.
- H. Beigy, “Deep Learning” Lecture slides, 2022, Sharif University of Technology.
- S. Levine, “CS W182/282A” Lecture slides, 2024, UC Berkeley
- Y. Maziane, “Diffusion models: Seek of information and structure in latent space,” 2022, University of Liège.
- G. Buzzard, “Mathematical Aspects of Neural Networks” Lecture slides, 2019, Purdue University.

Any Questions?