Autoencoder

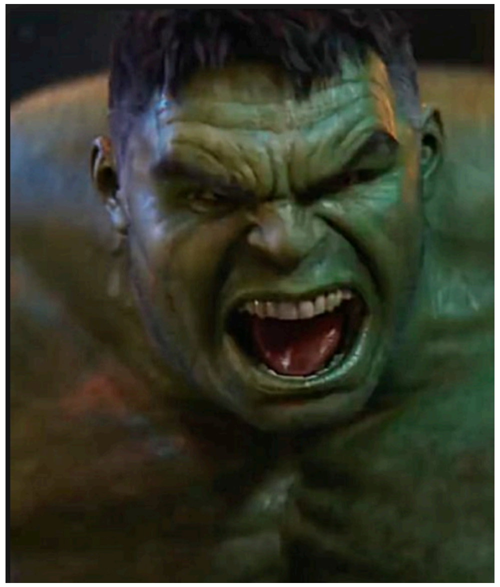
Foundations of Data Analysis

April 28, 2020

Slide credits to Jian Wang, PhD student, CS, UVA

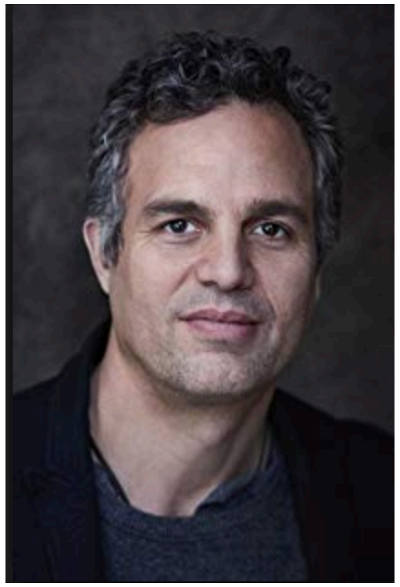
Overview: Data dimensionality

High-dimensional



Complicated;
Hard to handle

Low-dimensional



Simplified clear;
Property preserved

Dimensionality reduction techs:
Autoencoder/PCA

Applications of Autoencoder: Colorization

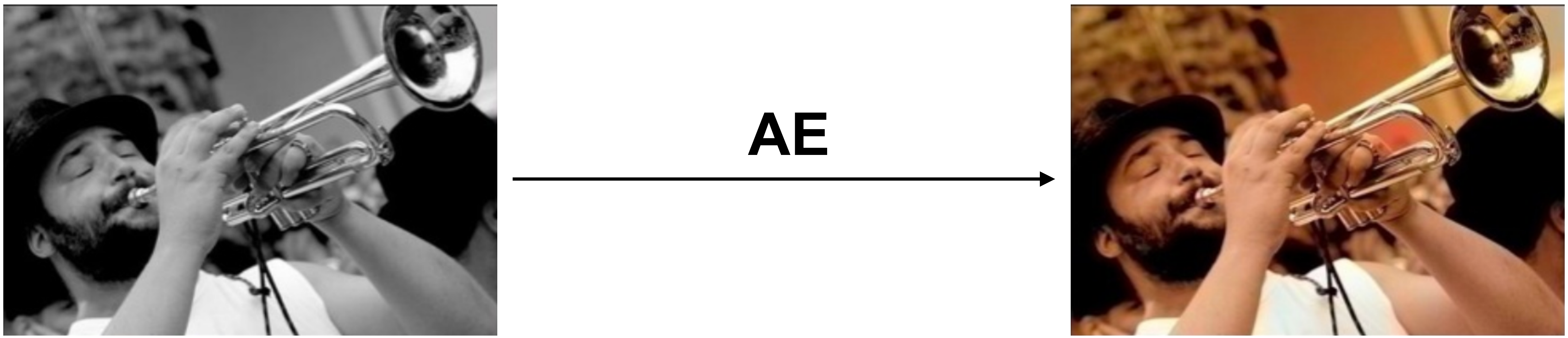
- Image coloring:

Autoencoder →

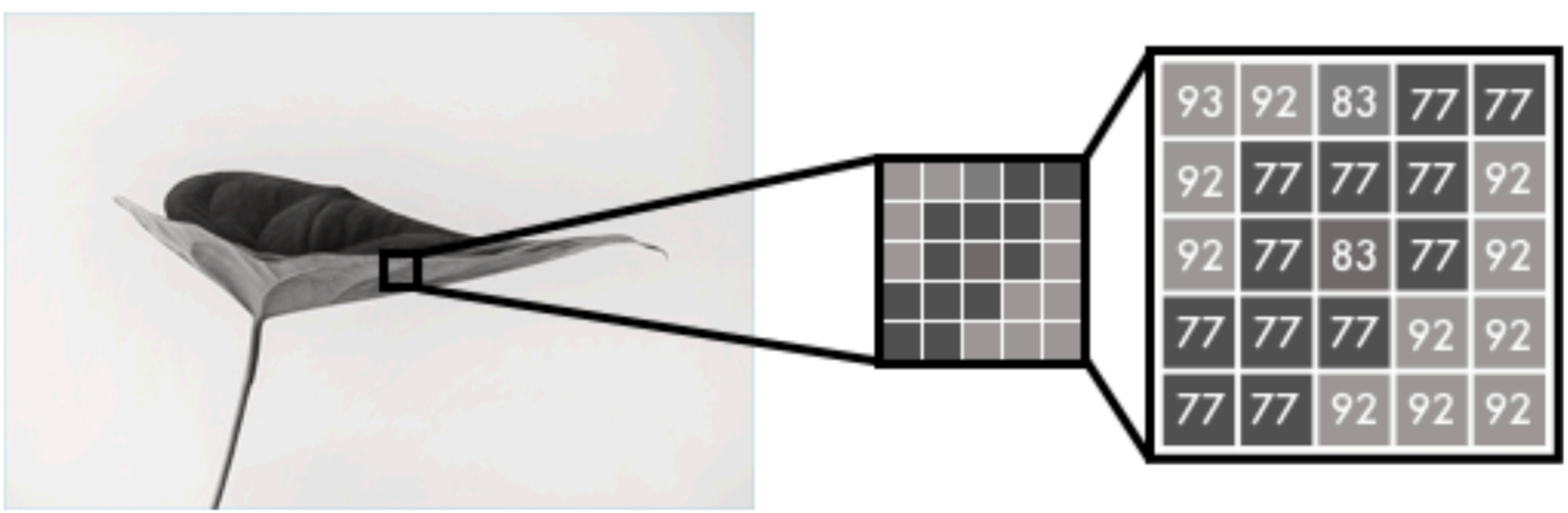
How?

- Reconstruct an image from grey color space to RGB/HSV.

Example: Image coloring



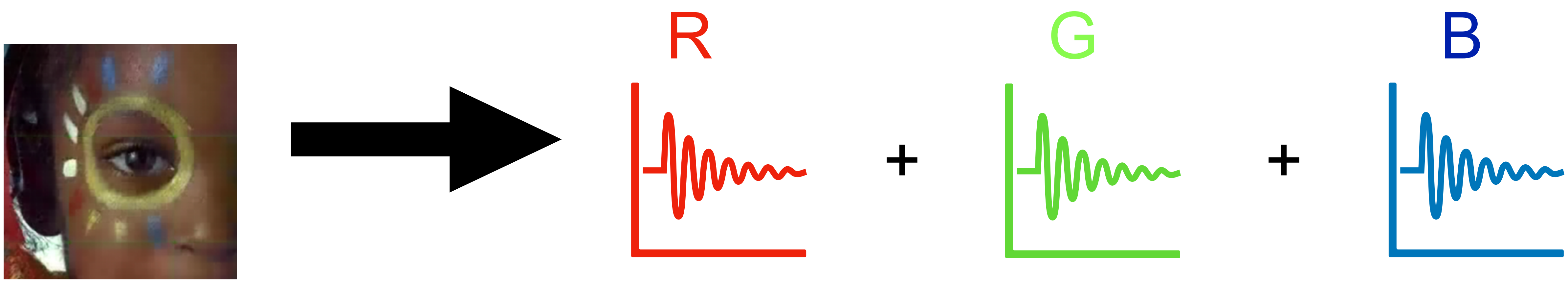
- Grey image color scale:



- Learn the intensity pattern!

Example: Image coloring

- Autoencoder training:



Train autoencoders for each channel from natural images.

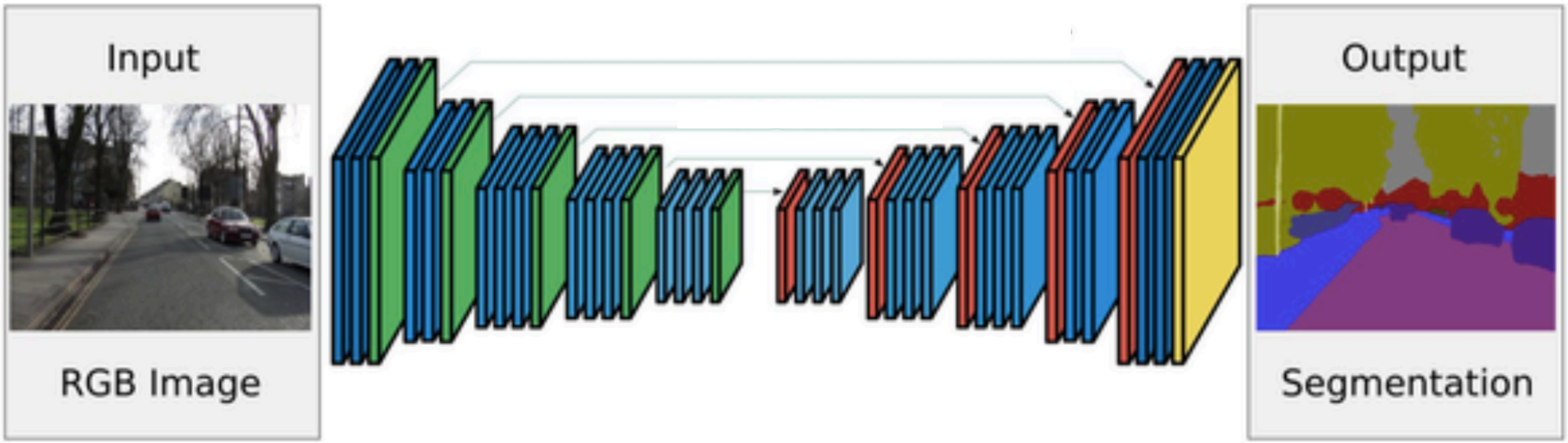
- Autoencoder prediction:

Well-trained
Autoencoder function

Predictions for each channel

Applications of Autoencoder: Segmentation

- Image segmentation:

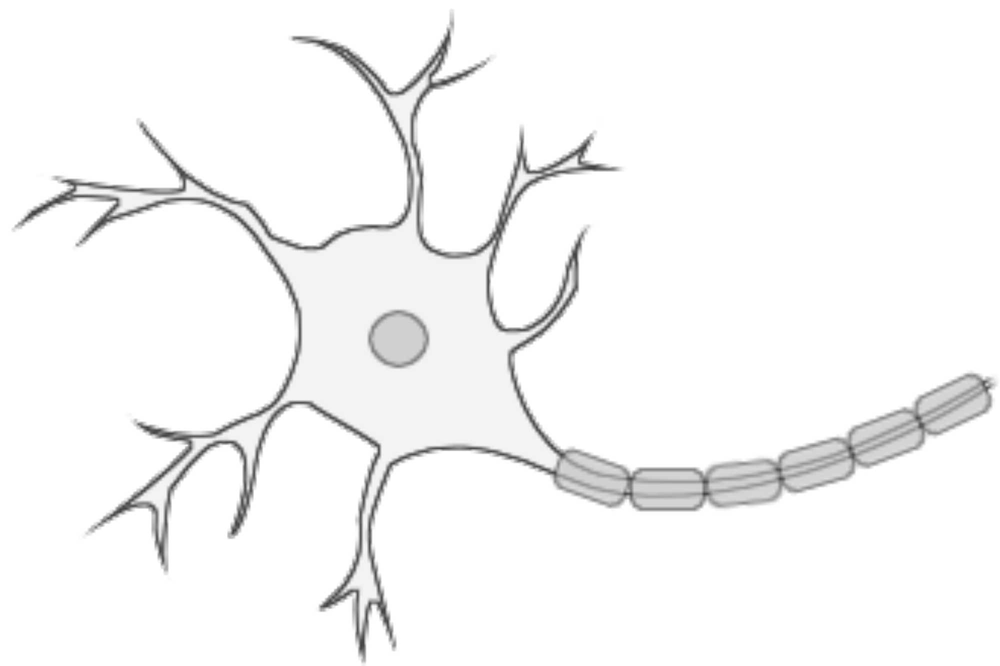


- What do we need ?

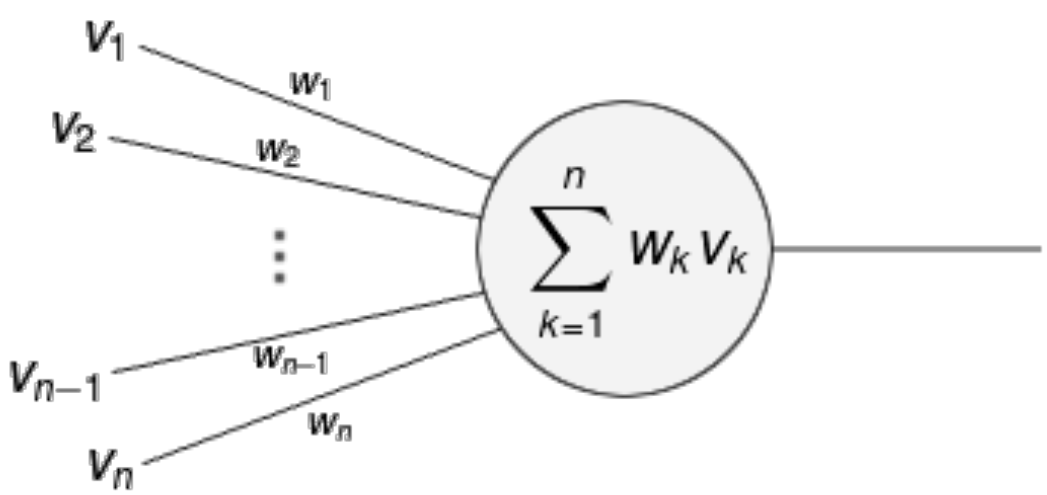
  Ground-truth, training, testing and validation.

Overview: Autoencoder (AE)

- Relationship to neural networks: layers, neurons, and weights;



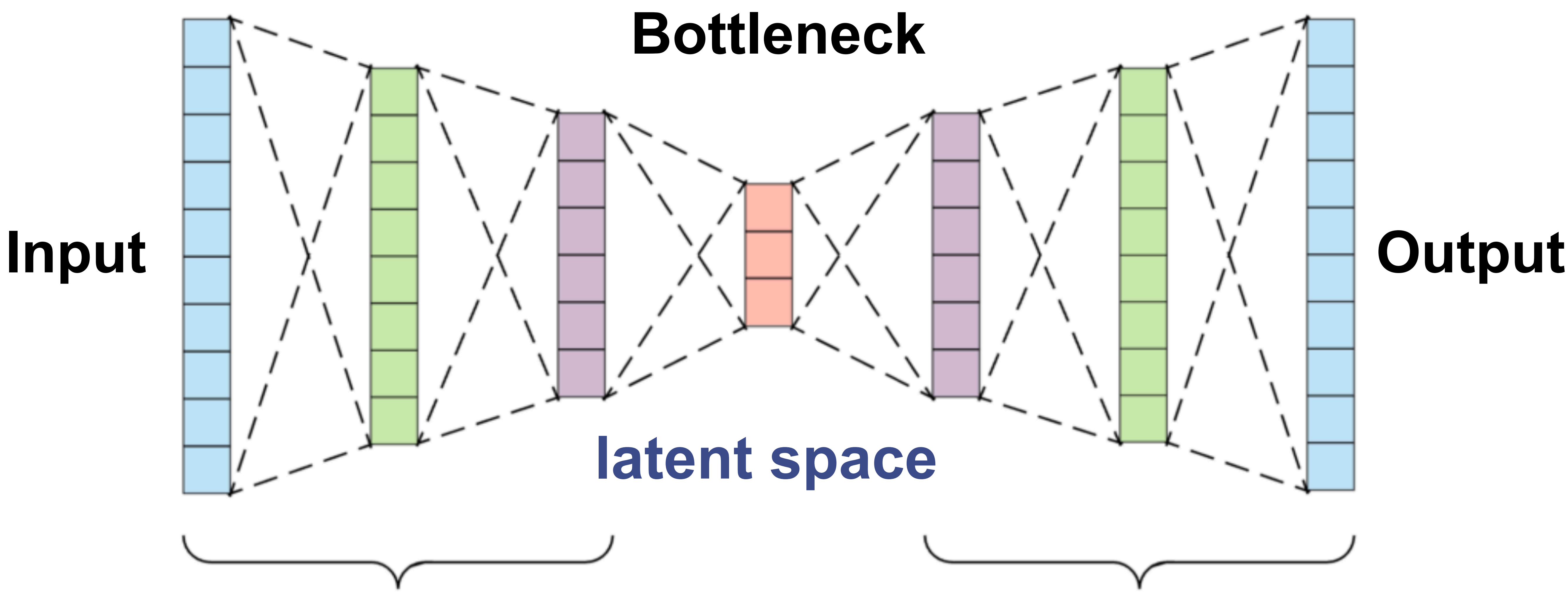
Brain neuron



Simulated neuron with input ***V*** and weights ***W***

Architecture : Autoencoder

- Core structure: **Encoder, Decoder** and **Bottleneck**

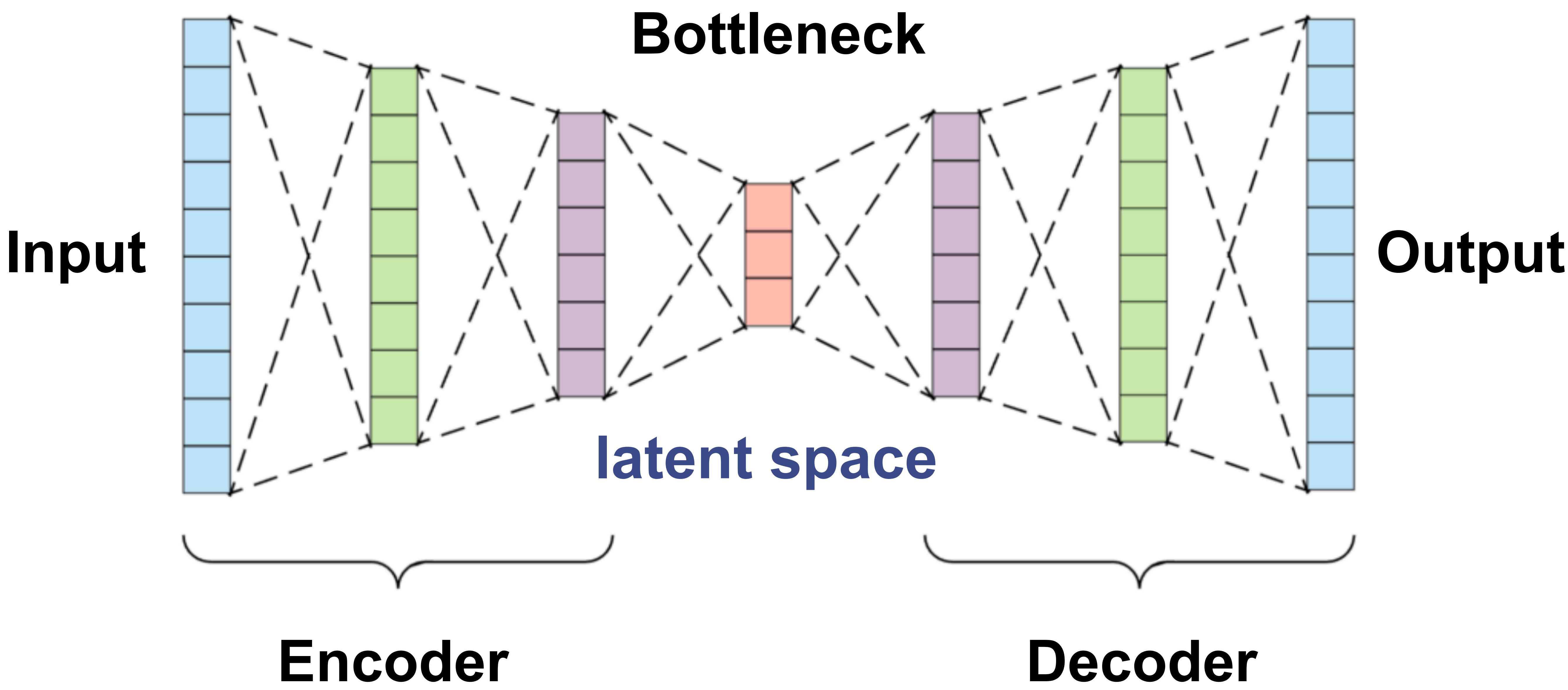


Encoder
Compresses the input into Bottleneck.

Decoder
Decodes the encoded image back to the original dimension.

Architecture : Autoencoder

- Core structure: **Encoder, Decoder** and **Bottleneck**



- Goal: minimize the reconstruction error.

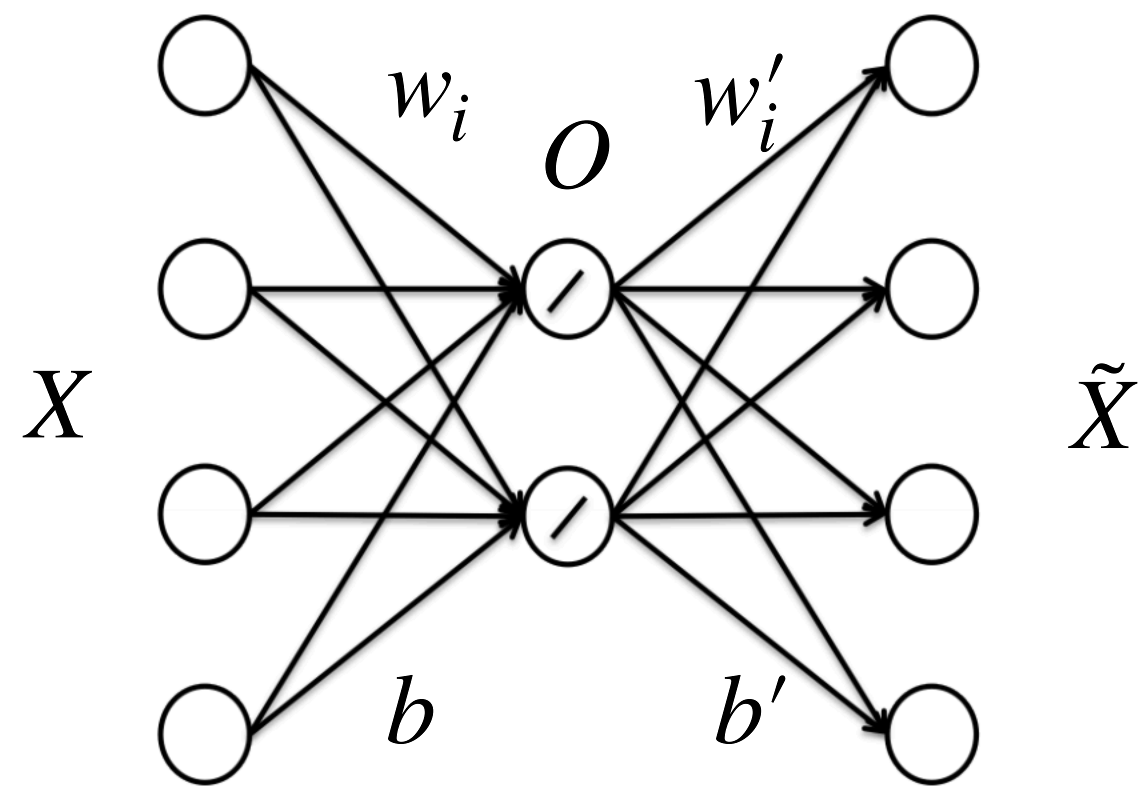
Formulation: Simple Autoencoder

- Encoder and decoder

Original Data：$[x_1, x_2, ... x_n]$ Output Data：$[x_1, x_2, ... x_n]'$

Latent input/encoder output：

$$O = \sum_{i=1}^{n} w_i x + b$$



Latent output/decoder output：

$$\tilde{X} = \sum_{i=1}^{n} w_i' O + b'$$

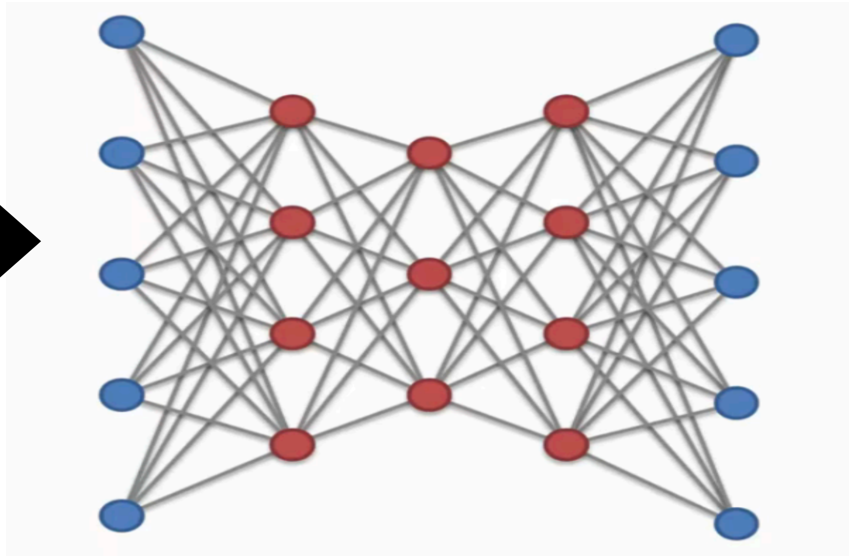
- Optimization:

Loss function： $f = argmin(X - \tilde{X}) = ||X - (w'x + b)||^2$

Major parameters in Autoencoder

- Layer size: more layers come with better accuracy?
- Size of bottleneck: number of nodes in the middle layer, (smaller size results in more compression);
- Loss function: mean squared error or binary cross-entropy;
- Penalty term: L_1 and L_2 regularity;

Deep autoencoder



Autoencoder Types

- Vanilla Autoencoder

  Simplest form, two layers with one hidden layer;

- Regularized Autoencoder

  Add regularization term to constraint loss function;

- Multilayer Autoencoder

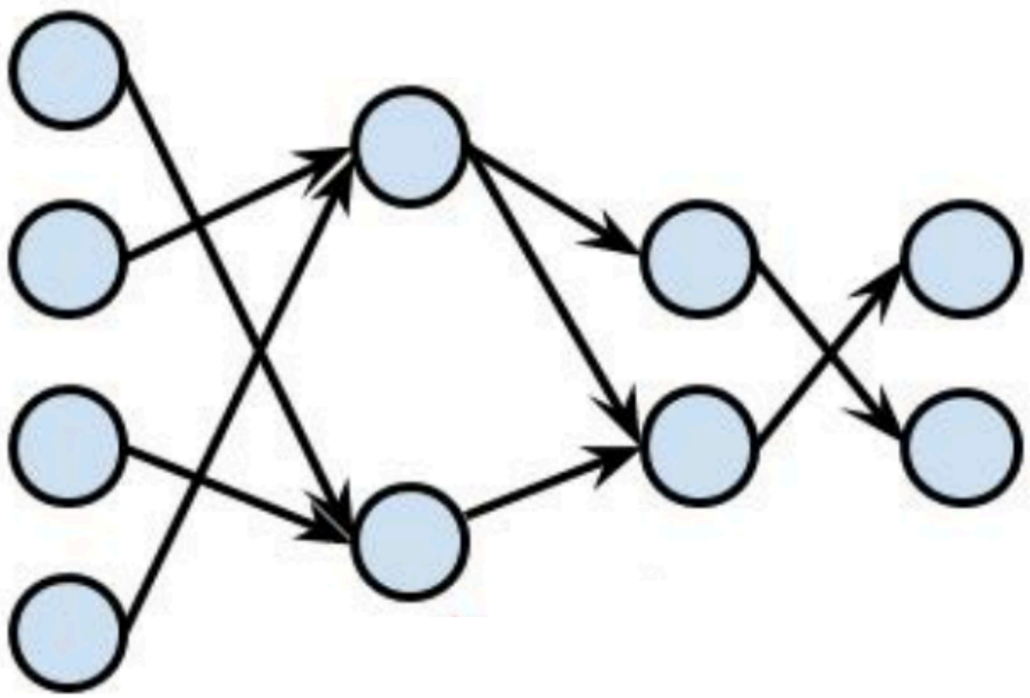
  Extended form based on Vanilla, multiple layers with multiple hidden layers;

- Convolutional Autoencoder

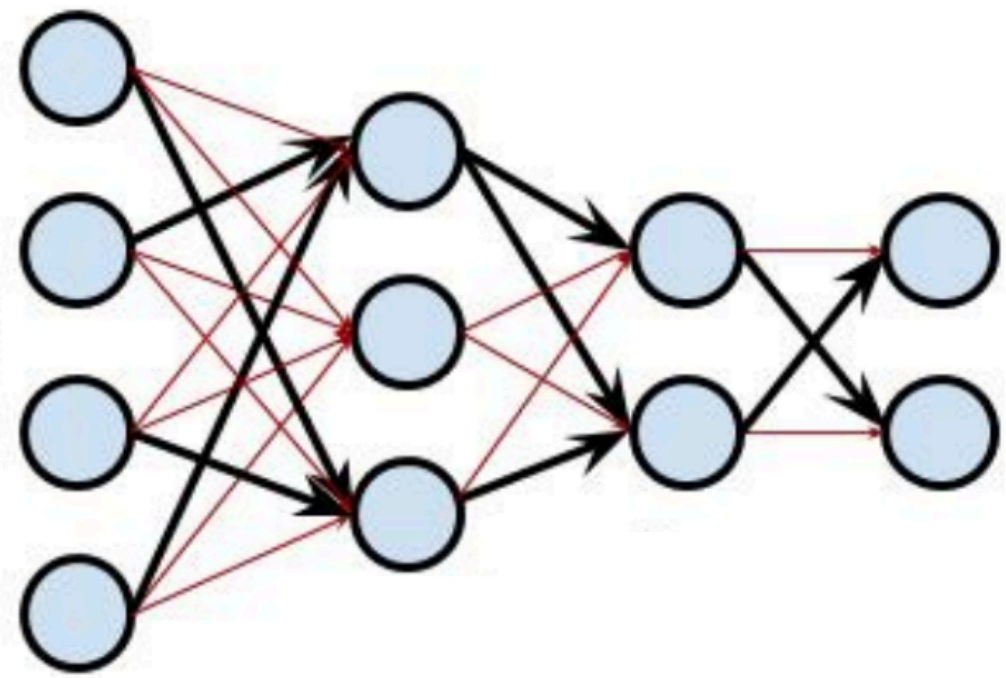
  Using pooling layers and convolution instead of fully connected layers;

Types: Regularized (Sparse) Autoencoder

- Sparse representation:



Sparse autoencoder



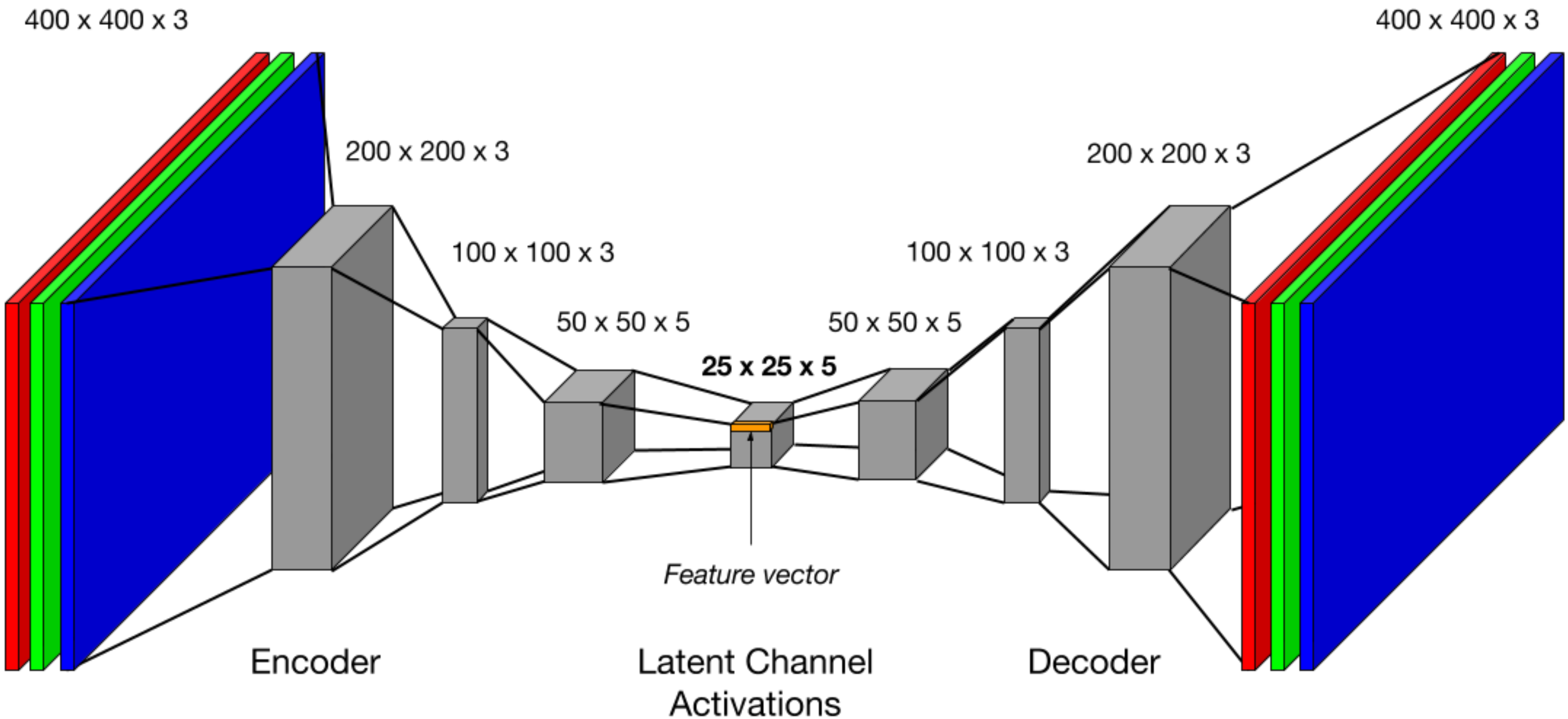
Fully-connected autoencoder

- Optimization and Loss function (Penalty term):

$$f = argmin(||X - (w'x + b)||^2 + \lambda||(w'x + b)||)$$

Types: Convolutional Autoencoder

- Structure

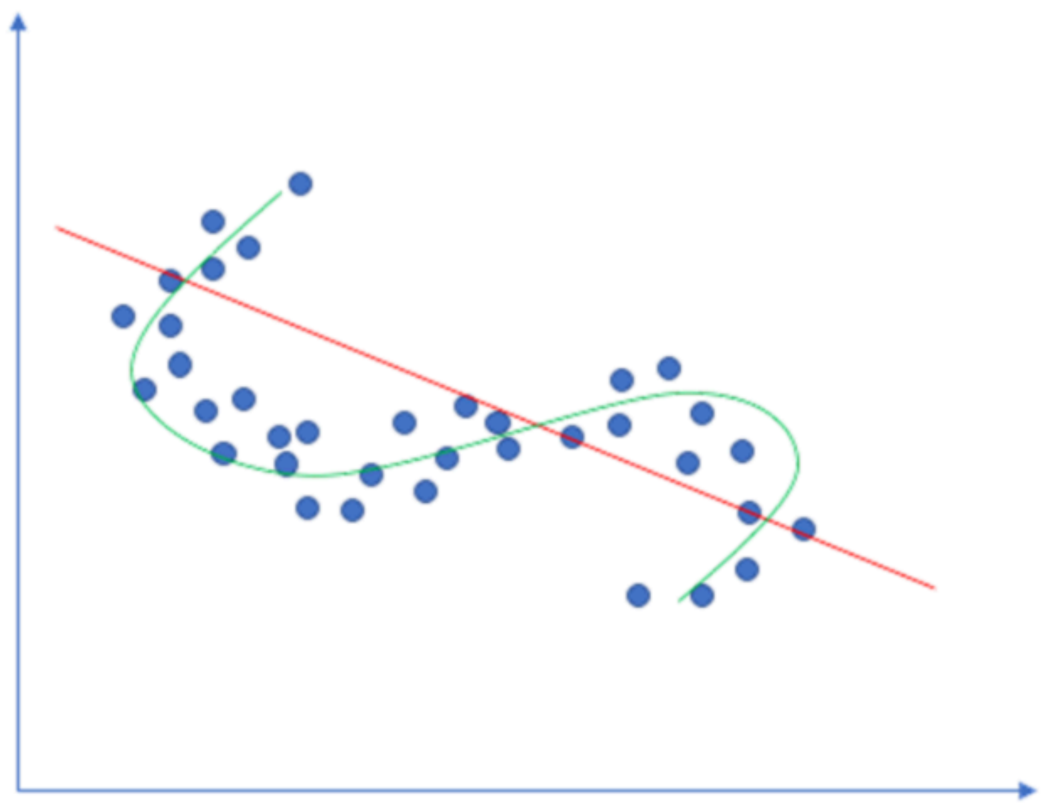


- Same operations with convolutional neuron network(CNN).
- More advanced autoencoders: Variational Autoencoder

Advantages: Autoencoder

- Producing pre-trained layers for another model; (clean)
- Doesn't have to learn dense layers; (cost)
- An autoencoder can capture non-linear features/patterns;

Difference between PCA and AE?



PCA vs AE