Deep Learning

Getting Started with Autoencoders

Learning Objectives

By the end of this lesson, you will be able to:

- Discuss the basics of autoencoders
- Analyze the architecture of autoencoders
- Evaluate the use cases of autoencoders
- Apply hyperparameters to train an autoencoder in Tensorflow

Business Scenario

A logistics company with vast unlabeled data seeks to identify patterns for operational optimization. The data is complex and resists manual clustering, necessitating a more advanced approach.

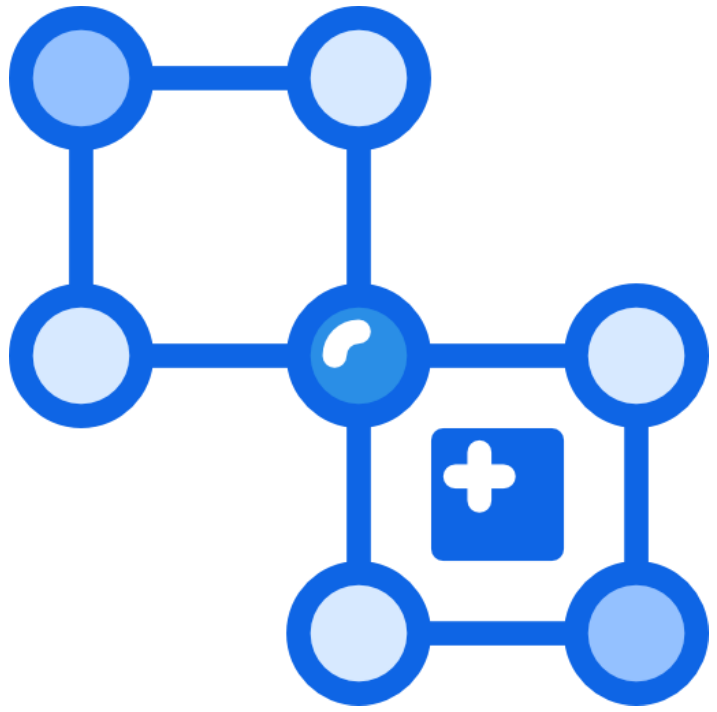
The company chooses unsupervised learning and partners with a deep learning provider to develop a multi-layered autoencoder. This autoencoder's design balances data complexity capture with an efficient, compact representation.

Upon optimizing the loss function, the autoencoder's output undergoes additional clustering. This step identifies patterns that facilitate route optimization, cost reduction, and improved delivery times.

Introduction to Unsupervised Deep Learning

Unsupervised Learning

It is a method of modeling the inherent structure or distribution in unlabeled data to understand its characteristics.



Unsupervised learning

- The model isn’t provided with labeled data.
- The model is trained to learn by itself to find structures in the data.

Unsupervised Learning: Example

It is difficult to group everyone's photos when there is a huge collection.

Unsupervised learning is used to group the photos based on some structural patterns it finds in the photos.

Grouping can be done based on the semantic information it learns from the images.

Google Photos uses this technique to group photos on the cloud.

Algorithms in Unsupervised Learning

Some of the popular algorithms used in unsupervised learning are:

Autoencoders	K-means in conjunction with autoencoders	Deep embedded clustering
It helps leverage neural networks to get a representation from the given input.	It uses autoencoders along with K-means clustering.	It learns deep feature representations that aid clustering in neural networks.

Autoencoders

Discussion

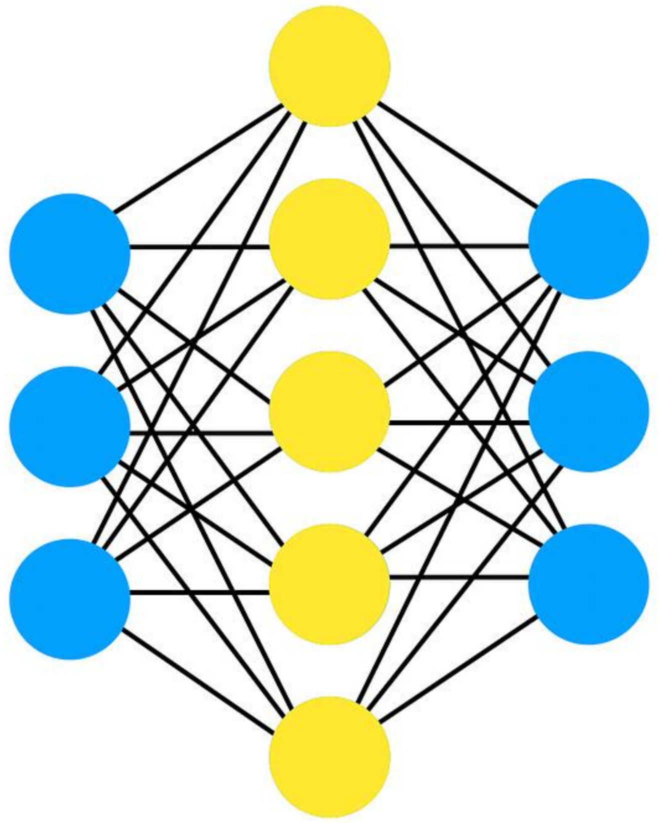
Discussion: Autoencoders

Duration: 10 minutes

- What are the advantages of using autoencoders?
- How do autoencoders work?

Autoencoders

Autoencoders are a specific type of neural network architecture used in unsupervised learning where the input is the same as the output.



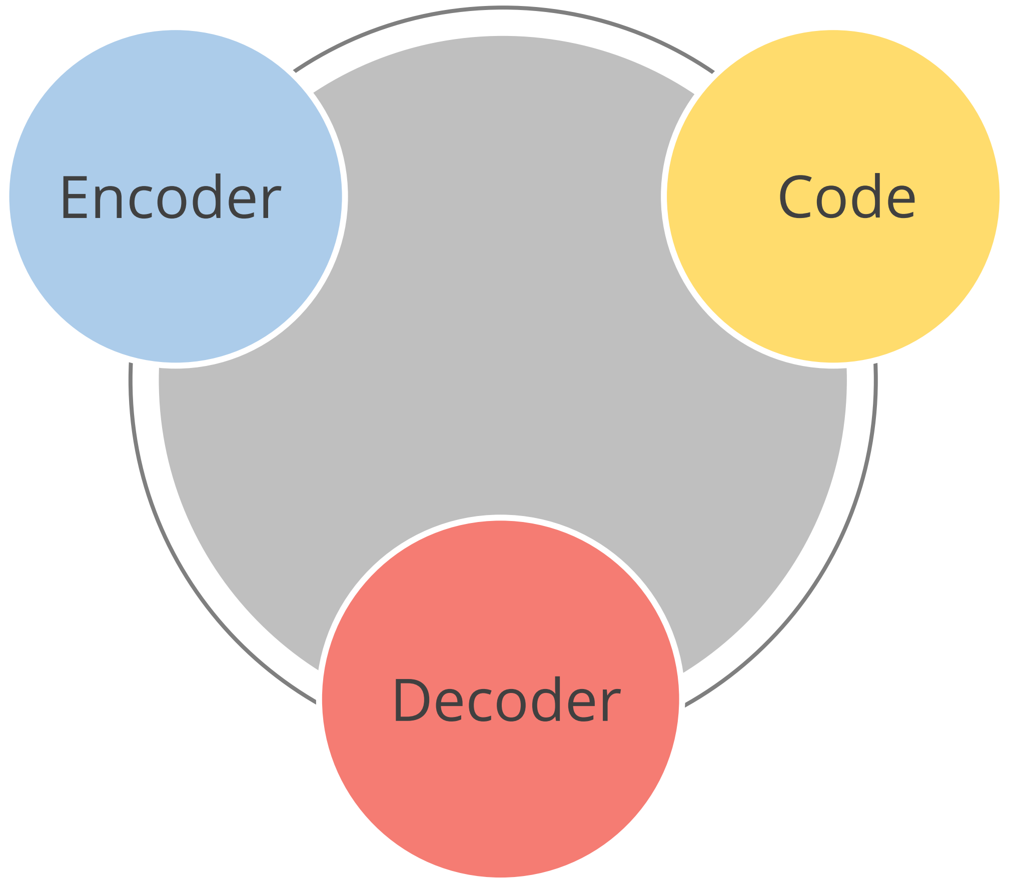
They compress the given input into lower dimensional code and further construct the input with this compressed representation.

The code is a compact summary, also called latent space representation.

Autoencoders

It is composed of three main parts:

Compresses the input into a lower-dimensional code

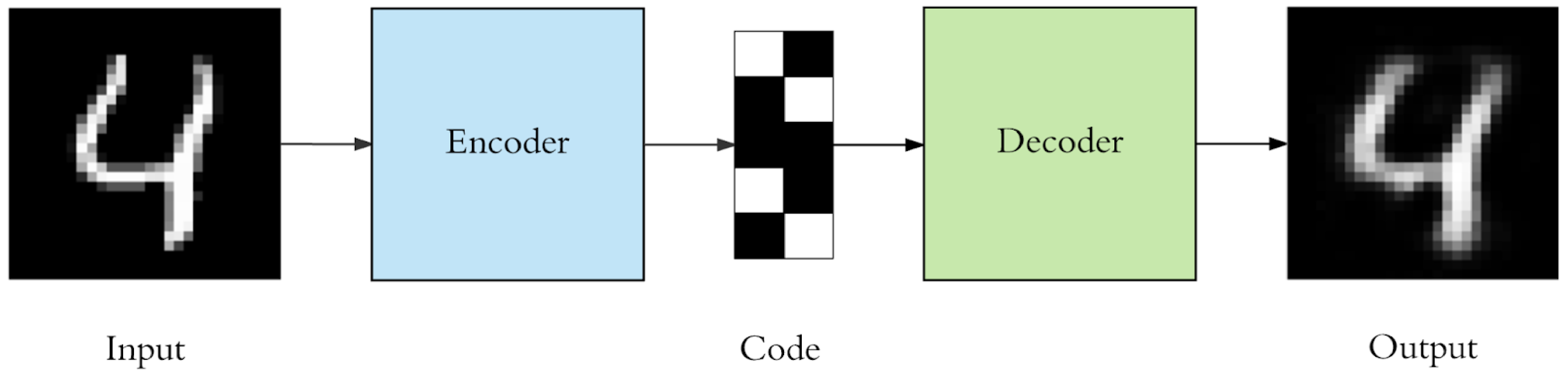


Is the compressed representation of the input data

Reconstructs the input from the code, aiming to match the original input as closely as possible

Autoencoders

Consider the image below to understand the workings of an autoencoder.



Encoder: Compresses the input and produces the code

Decoder: Generates the reconstructed input by using only the code

Properties of Autoencoders

The important properties of autoencoders are:

They are data-specific.

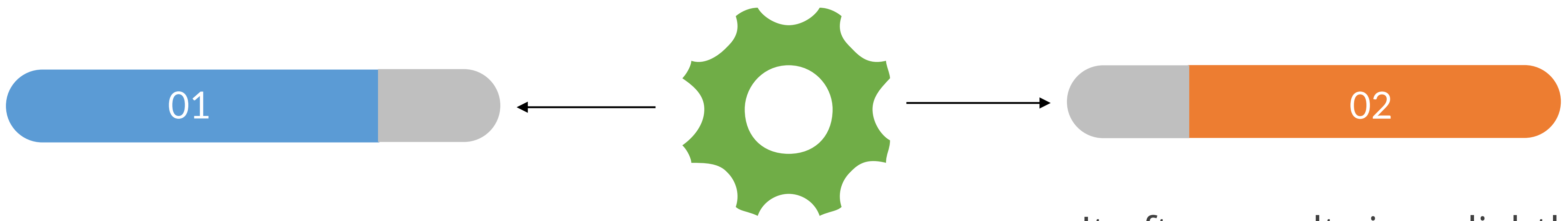
They can compress only those images on which they are trained.

Example

One cannot expect them to meaningfully compress landscape images if they are trained on handwritten digits.

Properties of Autoencoders

The important properties of autoencoders are:

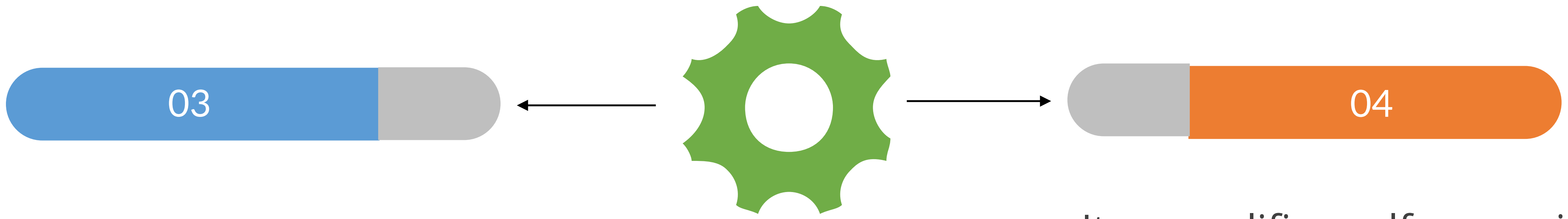


It works with the goal of having the output closely resemble the input.

It often results in a slightly altered version of the input through the encoding and decoding process.

Properties of Autoencoders

The important properties of autoencoders are:



It fits into the framework of unsupervised learning due to its independence from labeled data.

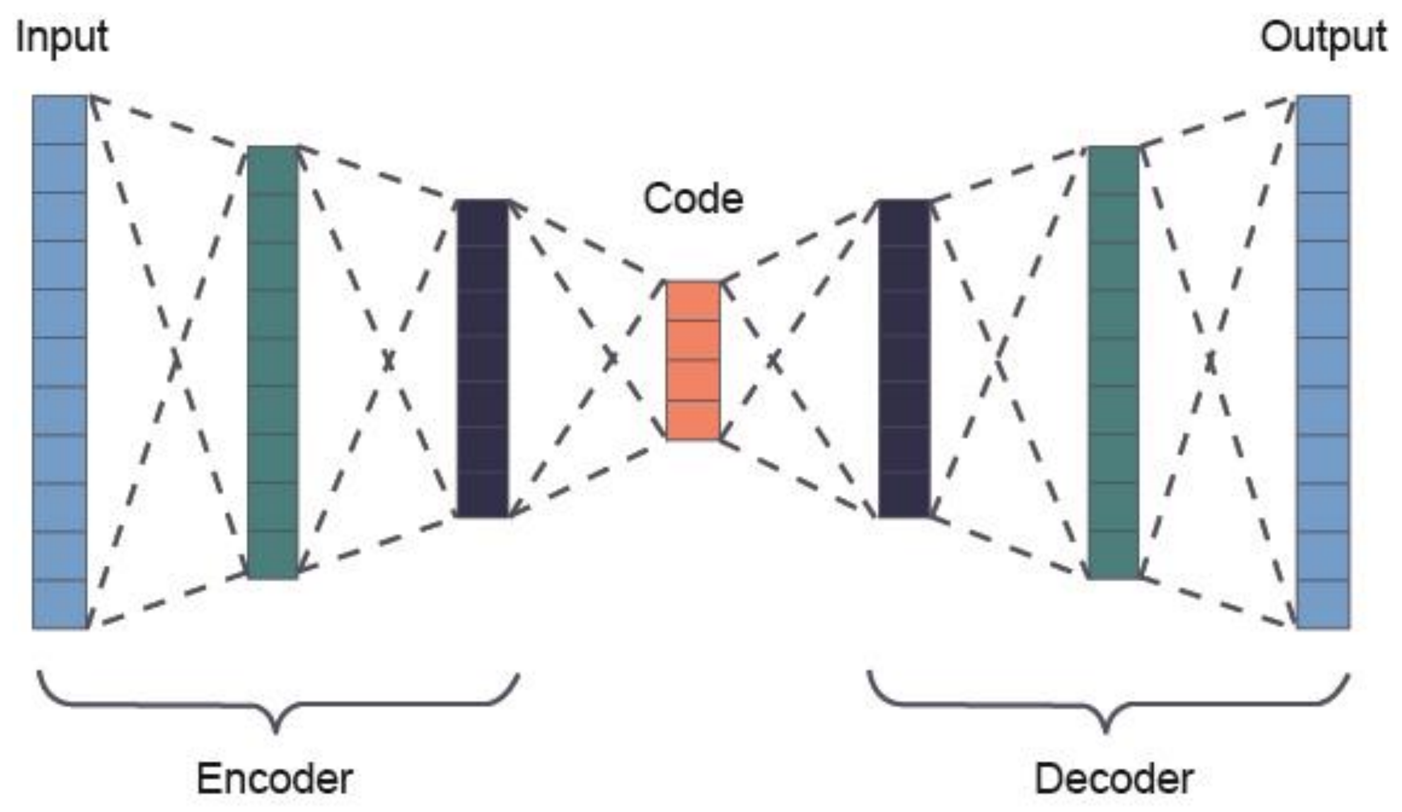
It exemplifies self-supervision by generating its own labels derived from the input data.

Architecture of Autoencoders

Architecture of Autoencoders

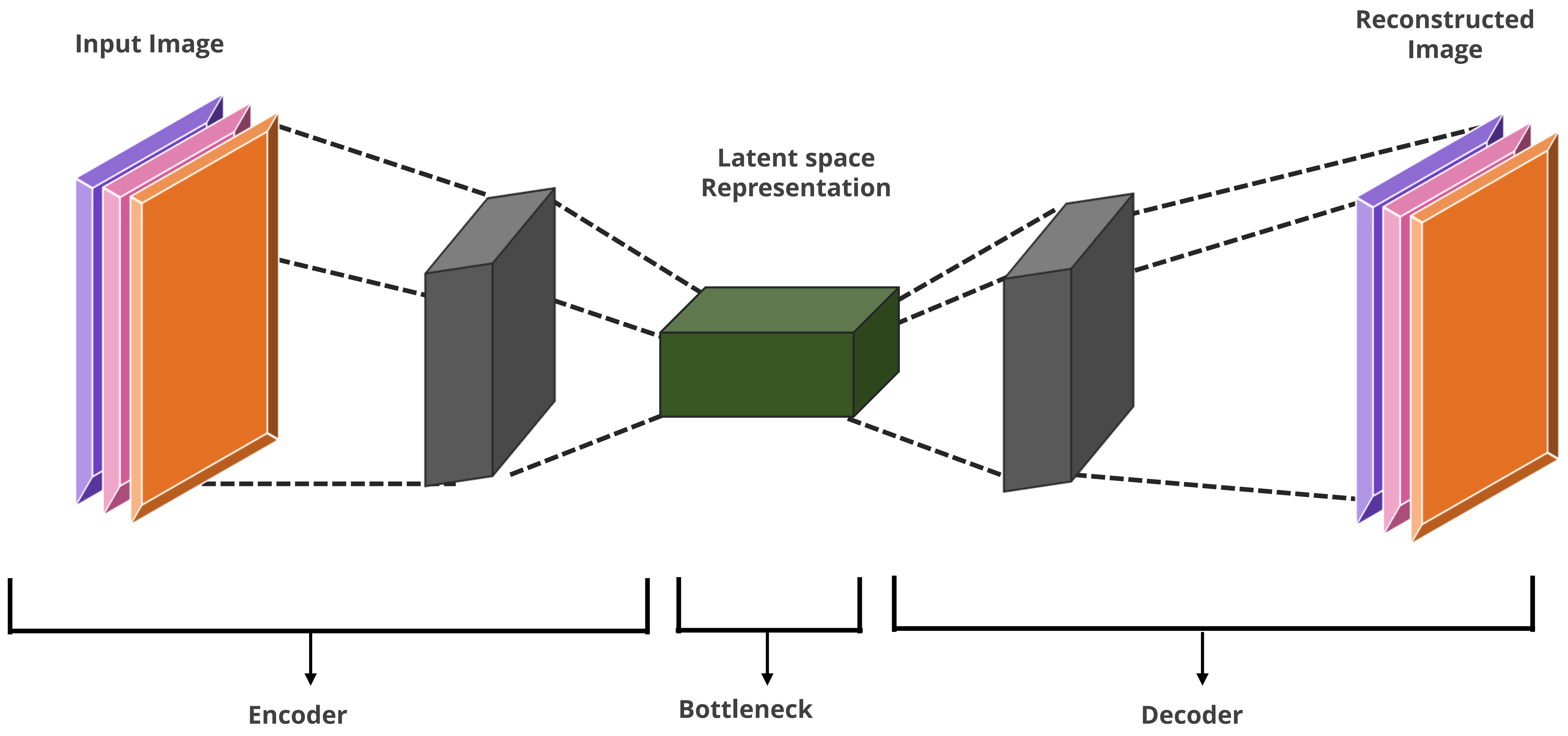
Autoencoders use symmetric encoder and decoder layers to minimize reconstruction error between input and output.

The following image shows the architecture of an autoencoder:



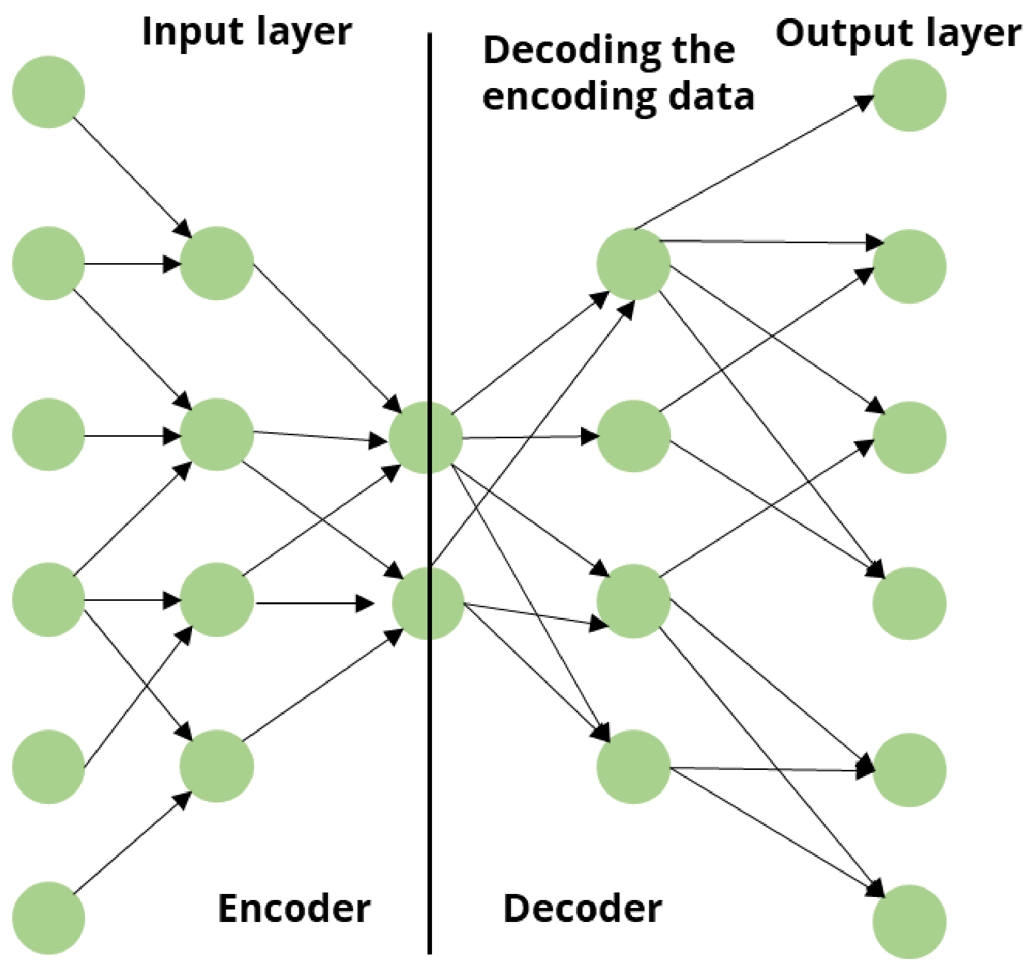
Components of Autoencoders

It consists of three main components: an encoder, a latent space (also known as a bottleneck), and a decoder.



Encoder

The encoder maps the input data to a lower dimensional representation called the latent space or bottleneck layer.



Each hidden layer typically employs activation functions such as ReLU, sigmoid, or tanh to introduce non-linearity and capture complex patterns in the data.

Encoder

It can be represented mathematically as:

$$z = f(Wx + b)$$

Here,

- **Z**: Compressed representation in the latent space
- **X**: Input data
- **W:** Weight matrix of the encoder
- **B**: Bias vector of the encoder
- **f():** Activation function applied element-wise to the linear transformation **Wx + b**

Decoder

The decoder takes the compressed representation from the encoder and reconstructs the original input data.

The decoding process of an autoencoder can be represented as:

$$x' = g(W'z + b')$$

Here,

- **x’**: Reconstructed output
- **W’**: Weight matrix of the decoder
- **b’**: Bias vector of the decoder
- **g()**: Activation function applied element-wise to the linear transformation **W'z + b'**

Bottleneck Layer

The bottleneck layer, located in the latent space, represents the compressed representation of the input data.

Its dimensionality is typically much lower than the dimensionality of the input data.

This compression forces the autoencoder to capture the most important features and discard noise or less relevant information.

Reconstruction Loss

During training, the autoencoder aims to minimize a reconstruction loss, which measures the difference between the reconstructed output and the original input.

$$\text{MSE} = \frac{1}{n}\sum_{i=1} (y_i - \tilde{y}_i)^2$$

MSE = Mean squared error

N = Number of data points

$\mathbf{y_i}$ = Observed values

$\tilde{\mathbf{y}}_\mathbf{i}$ = Predicted values

Commonly used reconstruction loss functions include mean squared error (MSE) or binary cross-entropy, depending on the nature of the input data.

Hyperparameters to Train an Autoencoder

The four hyperparameters that need to be set before training an autoencoder are:

Code size	Middle layer node count based on code size affects compression quality.
Number of layers	Autoencoders can have multiple layers on the encoder and decoder sides for feature extraction and reconstruction.

Hyperparameters to Train an Autoencoder

The four hyperparameters that need to be set before an autoencoder is trained are:

Number of nodes per layer	Encoder layers decrease nodes for dimensionality reduction, while decoder layers increase nodes for input reconstruction.
Loss function	The mean squared error (MSE), or cross-entropy is used as the loss function.

Discussion: Autoencoders

Duration: 10 minutes

- What are the advantages of using autoencoders?

Answer: Autoencoders offer advantages in unsupervised learning, dimensionality reduction, feature extraction, data denoising, anomaly detection, and transfer learning.

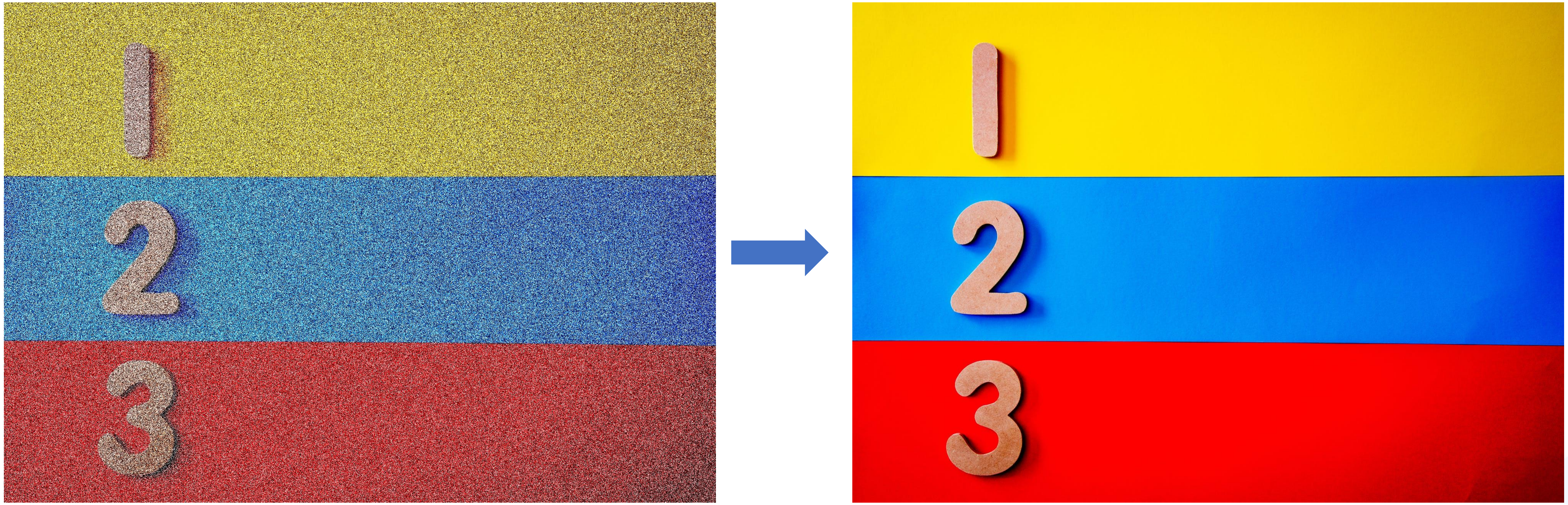
- How do autoencoders work?

Answer: Autoencoders compress data into a lower-dimensional representation through encoding and reconstruct it back through decoding. This process minimizes reconstruction error and enables the model to learn significant features within the data.

Use Cases of Autoencoders

Use Cases of Autoencoders

1. Data denoising can be used to reconstruct a noisy image to its original format.



Use Cases of Autoencoders

2. It can be used for dimensionality reduction.

t-SNE is a commonly used technique for the visualization of high dimensional data.

It struggles when the data is of larger dimensions (typically above 32).

Autoencoders used in preprocessing help reduce the dimensionality, and t-SNE uses this compressed representation output for 2D data visualization.

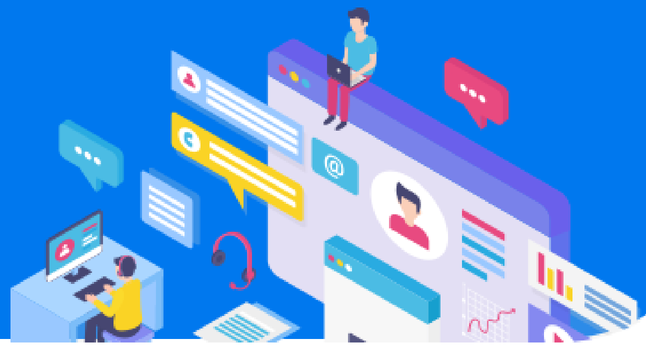
Use Cases of Autoencoders

3. It can be used as a variational autoencoder.

It learns the parameters of the probability distribution modeling the input data.

This is done as an alternative to learning an arbitrary function, as is done in plain vanilla autoencoders.

Assisted Practice



Let's understand the concept of autoencoders using Jupyter Notebooks.

- 12.06_Building and Visualizing an Autoencoder with the Fashion-MNIST Dataset

Note: Please refer to the Reference Material section to download the notebook files corresponding to each mentioned topic

Key Takeaways

- Unsupervised learning employs machine learning algorithms to analyze and cluster unlabeled data sets.
- Autoencoders are a type of neural network where the input is the same as the output.
- A typical autoencoder has three components: the encoder, code, and decoder.
- The four hyperparameters used to train an autoencoder are code size, number of layers, number of nodes per layer, and loss function.

Knowledge Check

Knowledge Check

1

What is unsupervised learning?

A. A technique that employs machine learning algorithms to analyze labeled datasets

B. A technique that employs machine learning algorithms to analyze and cluster unlabeled datasets

C. A technique that employs machine learning algorithms to analyze both labeled and unlabeled datasets

D. None of the above

Knowledge Check

1

What is unsupervised learning?

A. A technique that employs machine learning algorithms to analyze labeled datasets

B. A technique that employs machine learning algorithms to analyze and cluster unlabeled datasets

C. A technique that employs machine learning algorithms to analyze both labeled and unlabeled datasets

D. None of the above

The correct answer is **B**

It is a technique that employs machine learning algorithms to analyze and cluster unlabeled datasets.

Knowledge Check

2

What are the components of a typical autoencoder?

A. Encoder, code, and classifier

B. Decoder, classifier, and pooling layer

C. Encoder, code, and decoder

D. None of the above

Knowledge Check

2

What are the components of a typical autoencoder?

A. Encoder, code, and classifier

B. Decoder, classifier, and pooling layer

C. Encoder, code, and decoder

D. None of the above

The correct answer is **C**

A typical autoencoder has three components: encoder, code, and decoder.

Knowledge Check

3

What are the four hyperparameters that need to be set before training an autoencoder?

A. Number of layers, nodes per layer, color channels, and input size

B. Number of nodes per layer, learning rate, batch size, and code size

C. Code size, number of layers, number of nodes per layer, and loss function

D. Learning rate, batch size, code size, and input size

Knowledge Check

3

What are the four hyperparameters that need to be set before training an autoencoder?

A. Number of layers, nodes per layer, color channels, and input size

B. Number of nodes per layer, learning rate, batch size, and code size

C. Code size, number of layers, number of nodes per layer, and loss function

D. Learning rate, batch size, code size, and input size

The correct answer is **C**

Four hyperparameters need to be set before training an autoencoder: code size, number of layers, number of nodes per layer, and loss function.

Thank You!