Deep Learning with Keras and TensorFlow

Getting Started with Autoencoders

Learning Objectives

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

- Utilize the Fashion MNIST dataset to explore and identify the shape and structure of fashion item images
- Apply TensorFlow and Keras libraries to construct and manage neural network models for image data compression
- Train the model on fashion images, adjusting parameters like epochs and shuffle to enhance learning accuracy

Business Scenario

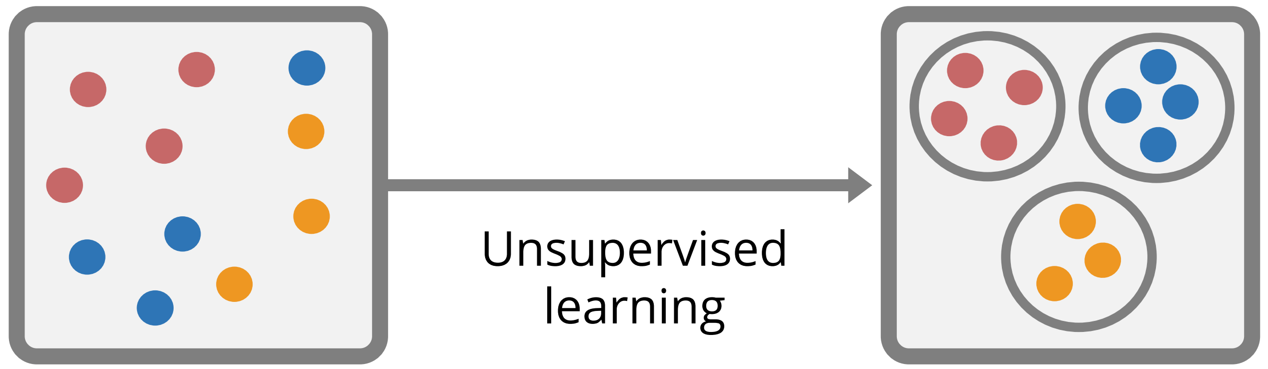
A large logistics company, facing challenges in managing and interpreting its vast amounts of complex, unlabeled data, aims to enhance operational efficiency. The data includes numerous variables that influence delivery times, costs, and route efficiency, making traditional analytical methods inadequate for extracting meaningful insights.

The primary challenge is the complexity and volume of the data, which resist straightforward clustering or manual analysis techniques. The company needs a solution that can both simplify and reveal the underlying patterns in the data to optimize logistics operations.

Introduction to Unsupervised Deep Learning

Unsupervised Learning

It is a subset of machine learning techniques where models are trained using data that has not been labeled, categorized, or classified.



Models identify patterns, relationships, and structures within the input 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.

Common Unsupervised Learning Approaches

Unsupervised learning models are employed for three fundamental tasks:

Clustering

The techniques covered under it include:

- K-means clustering
- Hierarchical clustering
- DBSCAN (Density-based spatial clustering of applications with noise)

Association

The algorithms used to generate association rules include:

- Apriori algorithm
- Eclat algorithm

Dimensionality reduction

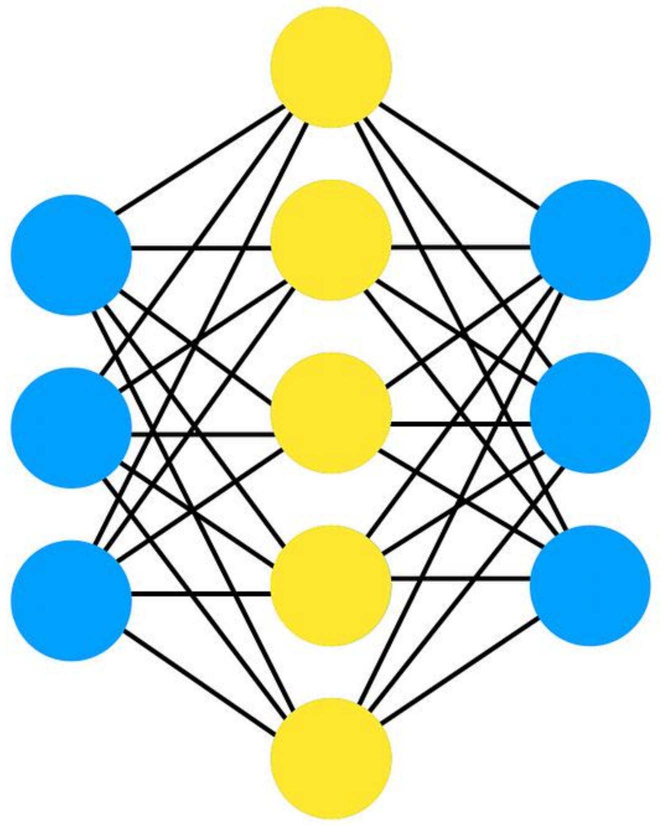
The dimensionality reduction methods that can be used include:

- Principal component analysis (PCA)
- Singular value decomposition (SVD)
- Autoencoders

Autoencoders

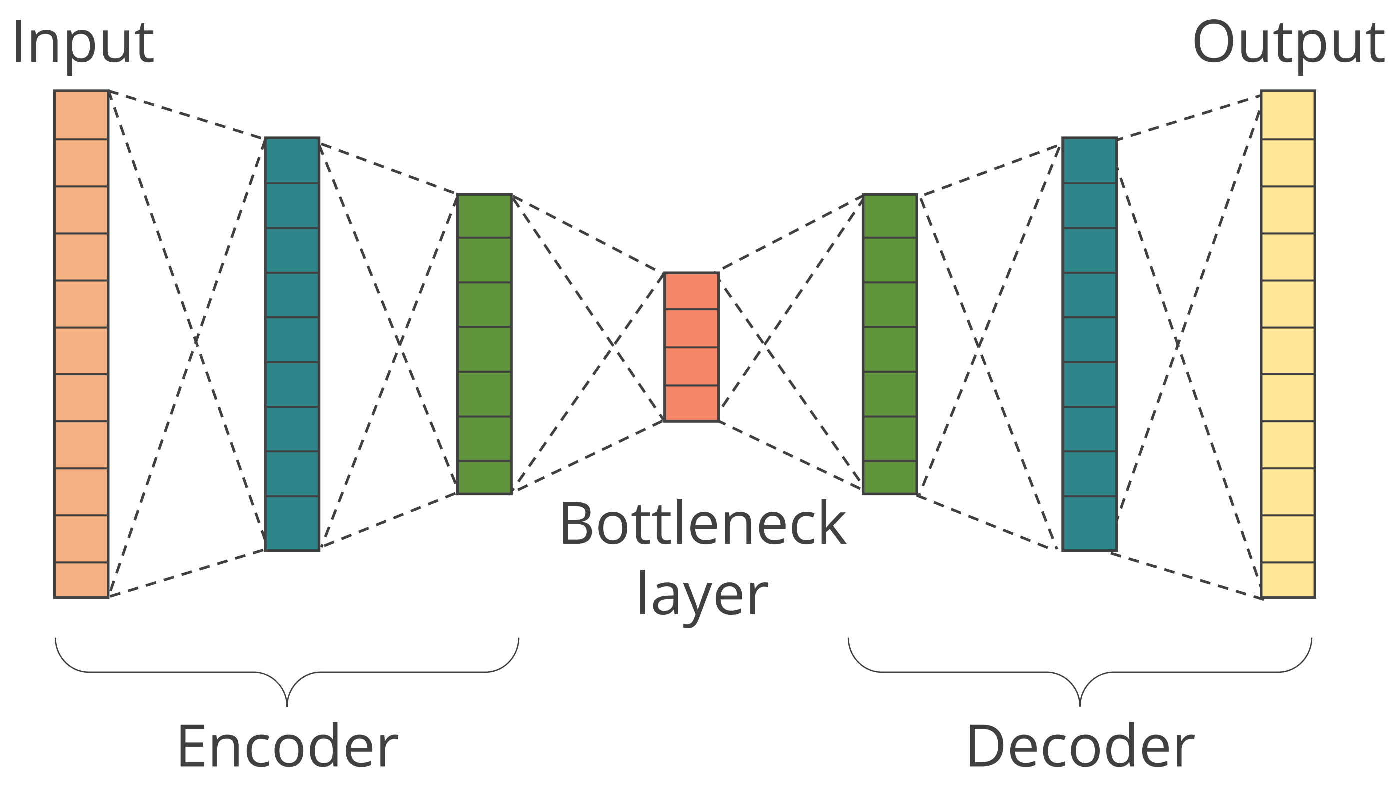
Autoencoders

Autoencoders are a type of neural network used in unsupervised learning that are designed to efficiently compress and encode data and reconstruct it back from the reduced encoded form.



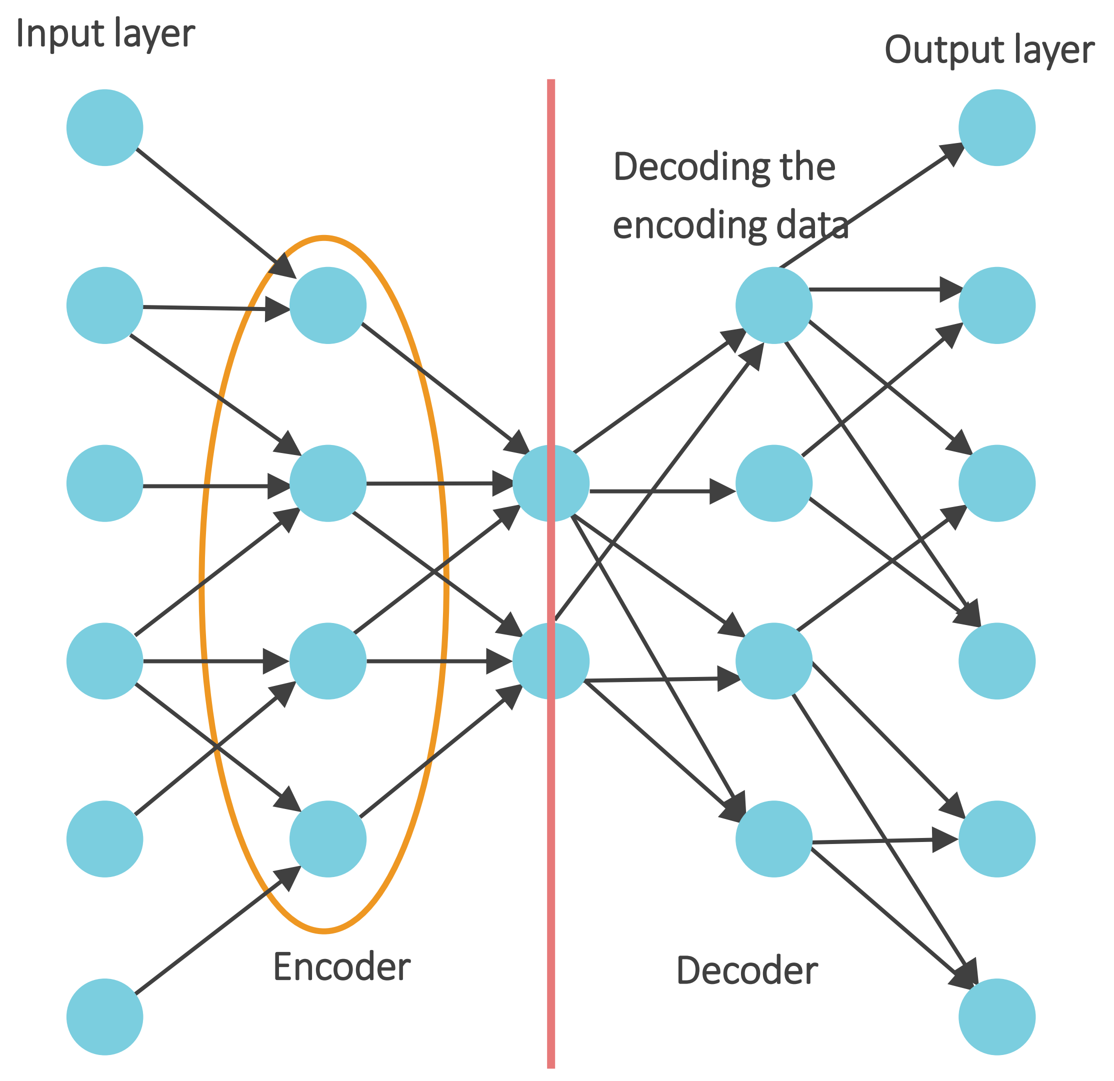
Autoencoders: Components

The general architecture of an autoencoder includes the encoder, a bottleneck layer, and the decoder.



Encoder

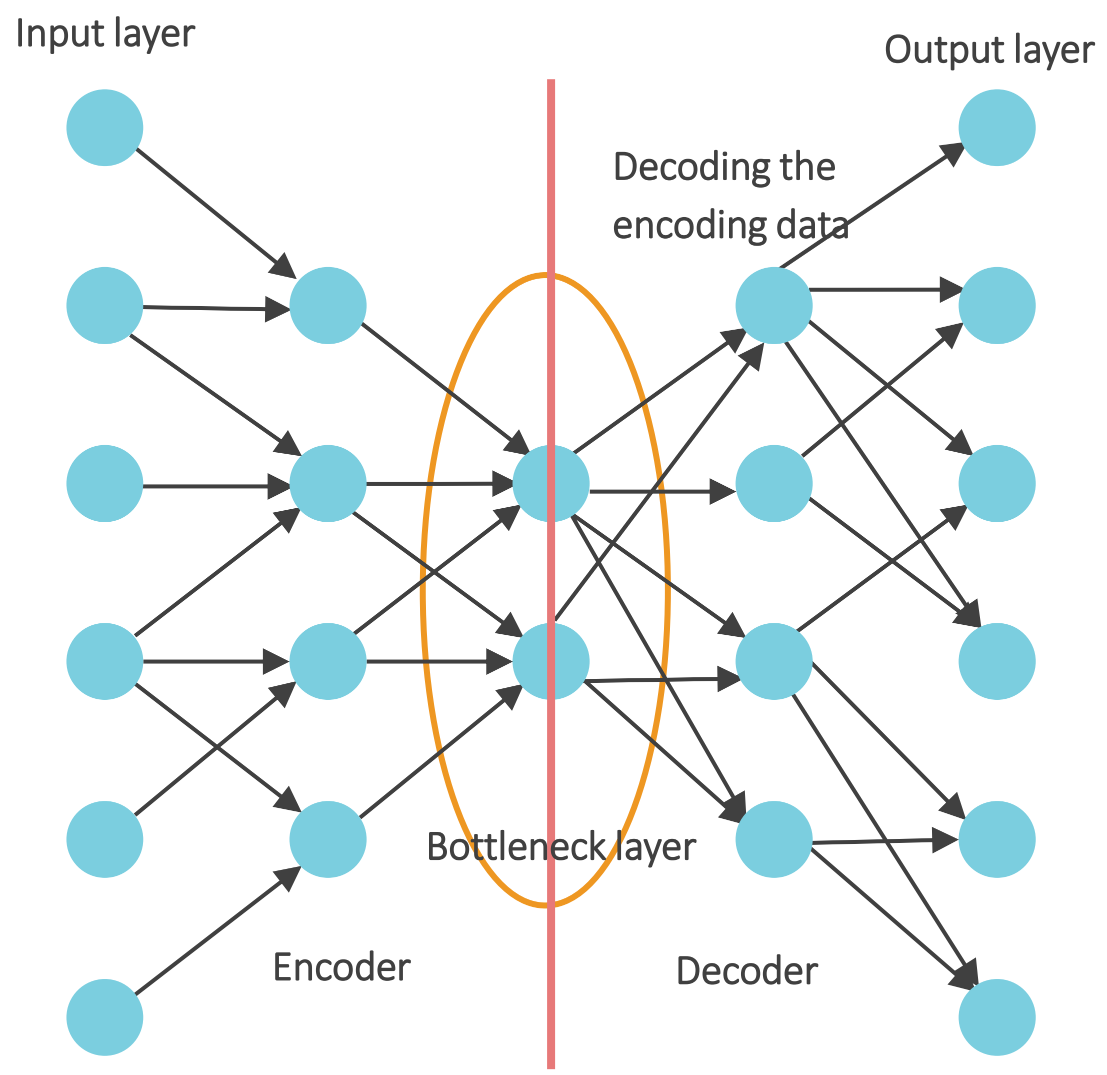
The encoder maps the input data to a lower dimensional representation, referred to as the code.



The encoder typically consists of one or more layers (for example, fully connected layers, convolutional layers, and so on) that gradually decrease in size, leading to the bottleneck layer.

Bottleneck Layer

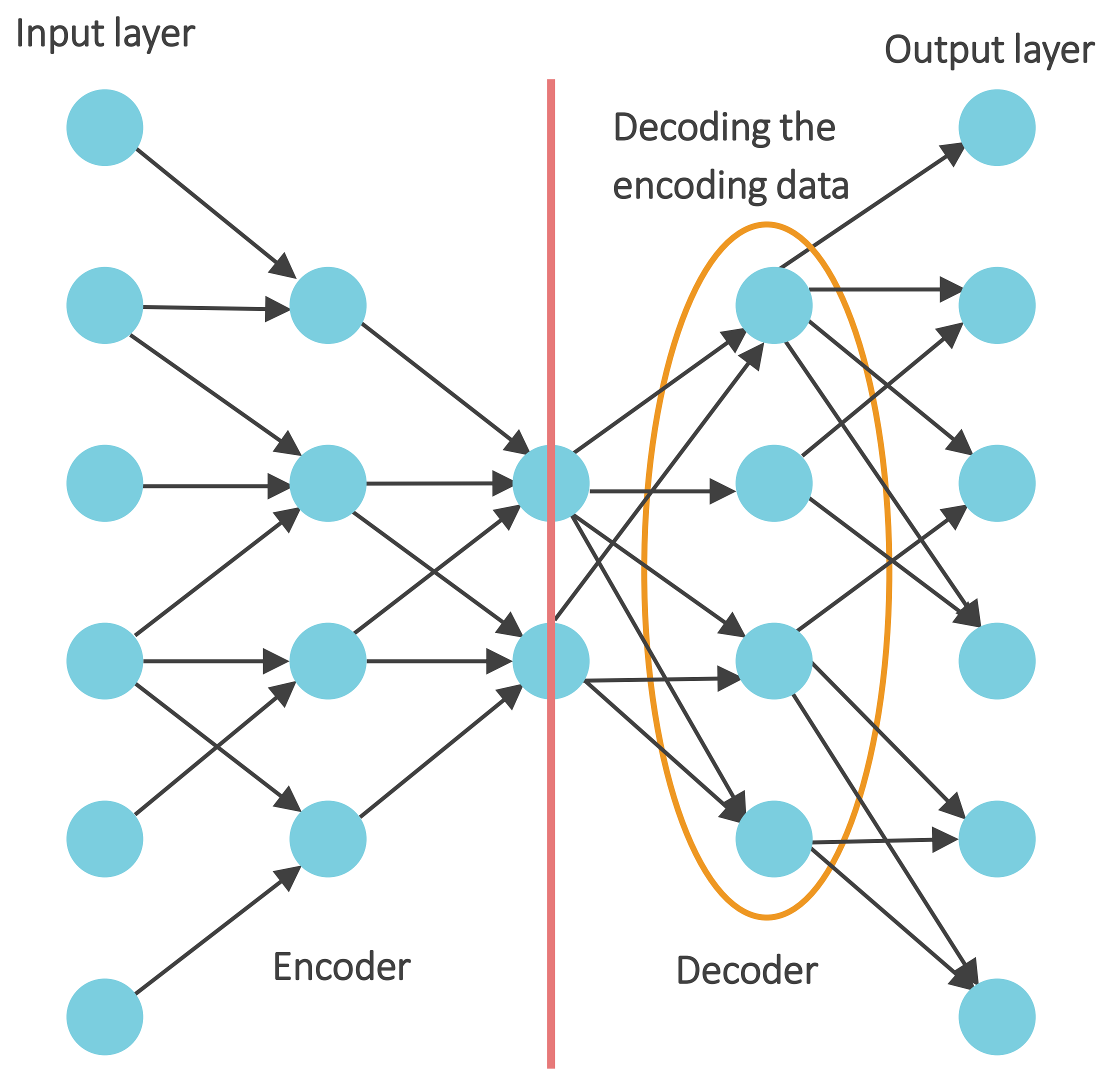
The bottleneck layer contains the compressed representation of the input data (code).



- 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.

Decoder

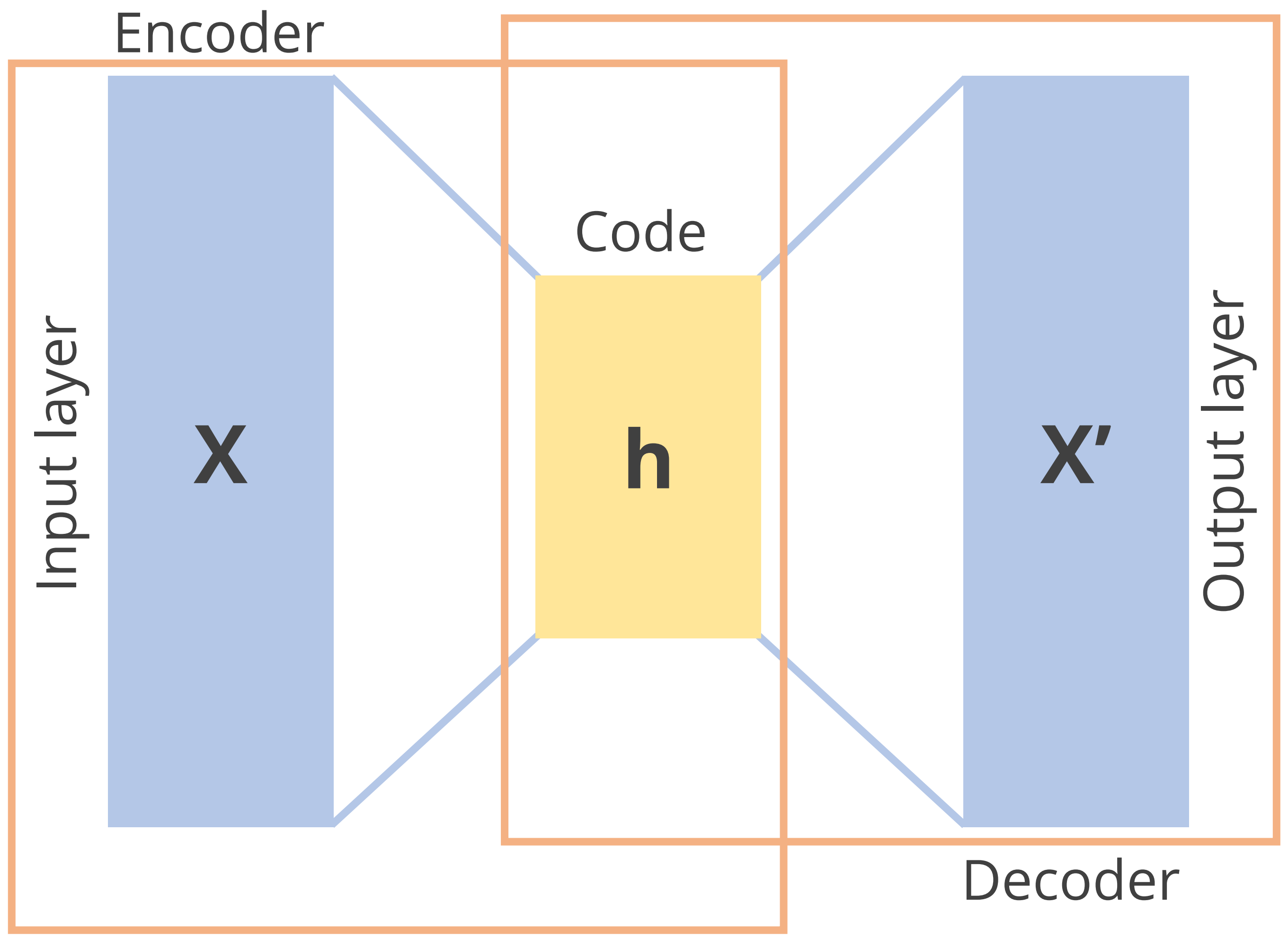
A decoder maps the lower-dimensional representation code back to the original input data.



- The decoder works like the encoder in reverse. It takes the encoded information and uses a similar structure to decode it, creating a final output.
- It consists of layers that progressively increase in size until reaching the size of the original input.
- This structure allows the decoder to expand the compressed code back into the full input.

Working Principle of Autoencoders

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



Reconstruction Loss

Reconstruction loss is 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 of Autoencoders

Hyperparameters to Train an Autoencoder

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

Code size	The code size is determined by the number of nodes in the bottleneck layer, which directly influences the quality of data compression. A smaller code size may enhance compression but at the risk of losing significant information.
Number of layers	There can be several layers on both the encoder and decoder sides. These layers facilitate the process of feature extraction and data reconstruction.

Hyperparameters to Train an Autoencoder

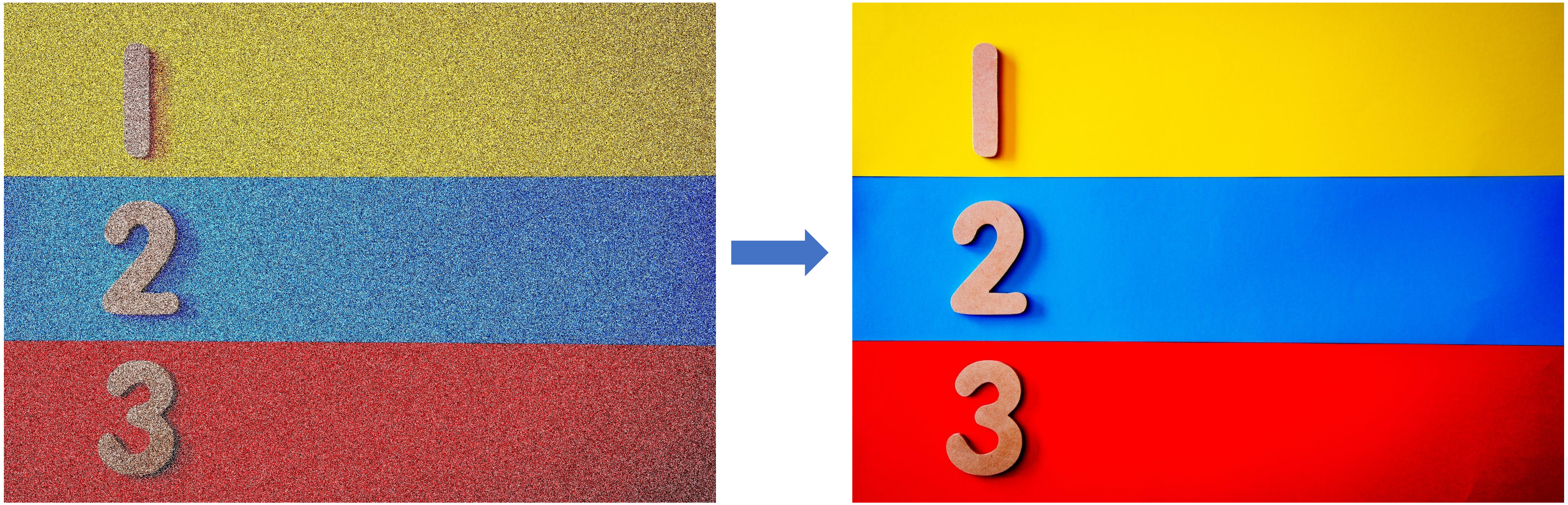
The four hyperparameters that need to be set before an autoencoder is trained are:

Number of nodes per layer	The number of nodes typically decreases in the encoder layers to achieve dimensionality reduction and increases in the decoder layers to reconstruct the input.
Loss function	The mean squared error (MSE) or cross-entropy is used as the loss function.

Use Cases of Autoencoders

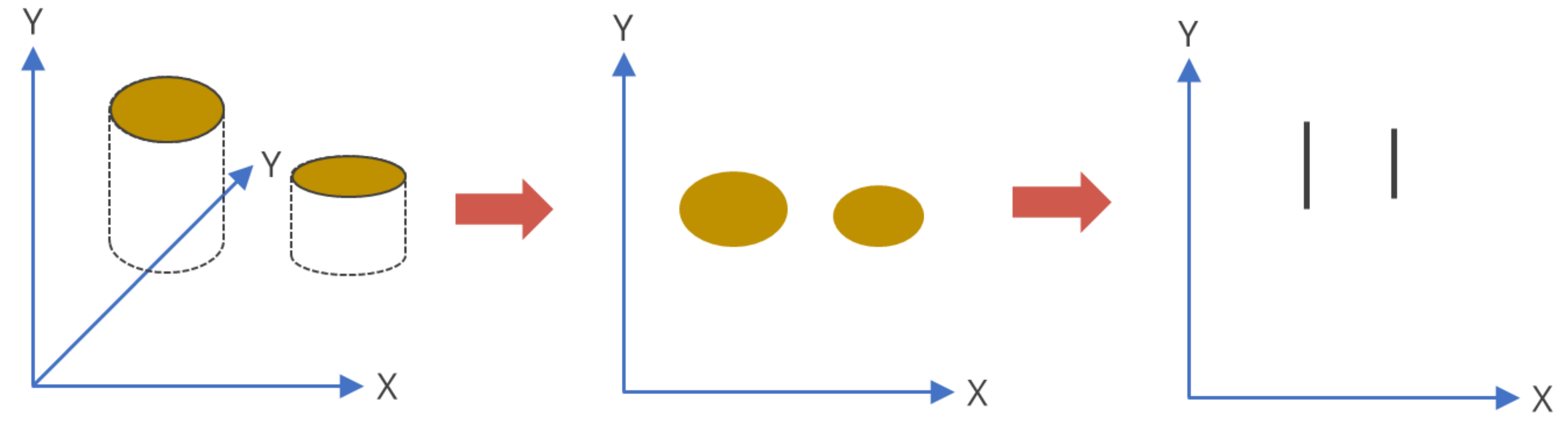
Use Cases of Autoencoders

Data denoising: Autoencoders can effectively remove noise from images, reconstructing them to their original, clear format.



Use Cases of Autoencoders

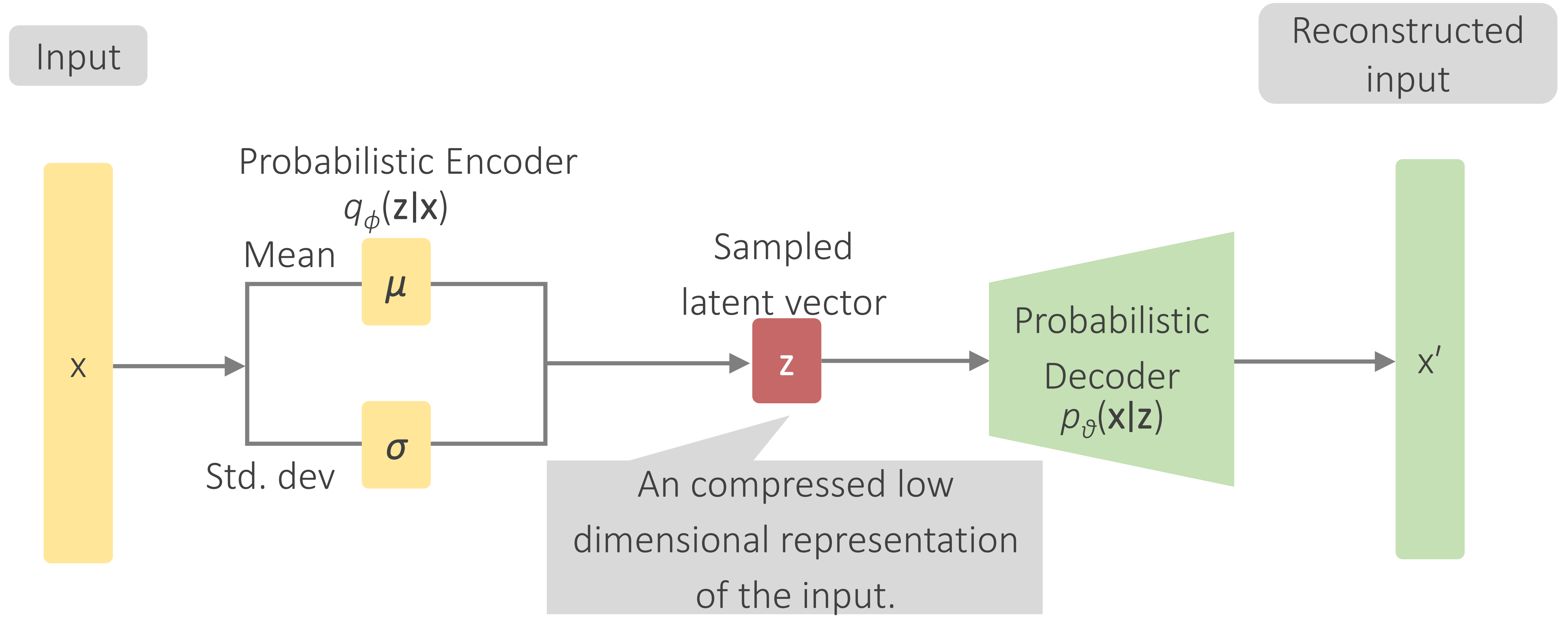
Dimensionality reduction: Autoencoders can reduce the dimensionality of data, preserving essential features while significantly compressing the data.



The reduced dimensional representation can also be used for visualization, making it easier to identify patterns and relationships in the data.

Use Cases of Autoencoders

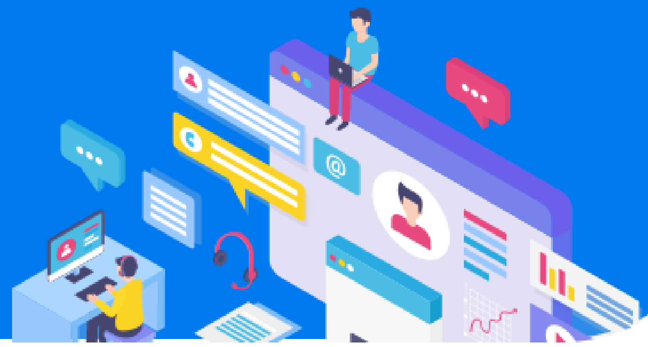
Variational Autoencoders (VAEs) are a specialized type of autoencoders that enhance the basic autoencoder architecture by incorporating probabilistic latent variables and a unique objective function.



Input
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
Mean
μ
Std. dev
σ
x
Sampled latent vector
z
An compressed low dimensional representation of the input.
Probabilistic Decoder
$p_\vartheta(\mathbf{x}|\mathbf{z})$
x′

Variational autoencoders (VAEs) are autoencoders with generative capabilities.

Assisted Practice



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

- 13.05_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

- Autoencoders are a type of neural network used for learning efficient patterns of unlabeled data by aiming to replicate the input at the output.
- The general architecture of an autoencoder includes the encoder, a bottleneck layer, and the decoder.
- The four hyperparameters used to train an autoencoder are code size, number of layers, number of nodes per layer, and loss function.
- Autoencoders can effectively remove noise from images, reconstructing them to their original, clear format, and reduce the dimensionality of data while preserving essential features.

Knowledge Check

Knowledge Check

1

What are the components of a typical autoencoder?

A. Encoder, bottleneck layer, and classifier

B. Decoder, classifier, and pooling layer

C. Encoder, bottleneck layer, and decoder

D. None of the above

Knowledge Check

1

What are the components of a typical autoencoder?

A. Encoder, bottleneck layer, and classifier

B. Decoder, classifier, and pooling layer

C. Encoder, bottleneck layer, and decoder

D. None of the above

The correct answer is **C**

A typical autoencoder has three components: the encoder, a bottleneck layer, and the decoder.

Knowledge Check

2

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 2

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.

Knowledge Check

3

Which of the following is a primary use case of autoencoders in the field of image processing?

A. Image classification

B. Image denoising

C. Image enhancement

D. Image resizing

Knowledge Check

3

Which of the following is a primary use case of autoencoders in the field of image processing?

A. Image classification

B. Image denoising

C. Image enhancement

D. Image resizing

The correct answer is **B**

Autoencoders are particularly effective in the task of image denoising. They are trained to remove noise from images by learning to represent and reconstruct the underlying clean image from the noisy input.

Thank You