USN NUMBER: 1RVU22CSE071
NAME: Joselyn Riana Manoj

Ex No: 6 Date: 10/09/2024	MNIST Autoencoder

Objective:

The objective is to build and train a simple autoencoder for image reconstruction using the MNIST dataset. The autoencoder will consist of an encoder and a decoder network, where the encoder compresses the input images into a lower-dimensional latent space, and the decoder reconstructs the original images from this latent representation.

Descriptions:

An autoencoder is a specialized type of neural network designed for the purpose of learning efficient representations of data, often used for tasks such as dimensionality reduction and feature extraction. In this project, we aim to construct and train a simple autoencoder to reconstruct images using the MNIST dataset, which contains grayscale images of handwritten digits.

The MNIST dataset, a well-known benchmark in the fields of machine learning and computer vision, consists of 60,000 training images and 10,000 test images of handwritten digits, each with a resolution of 28x28 pixels. To utilize this dataset for training an autoencoder, each image needs to be preprocessed appropriately. This involves normalizing pixel values from their original range of 0 to 255 down to a range of 0 to 1, which helps stabilize the training process and improve model performance. Additionally, the 28x28 images are flattened into vectors of size 784, transforming the two-dimensional images into a one-dimensional format suitable for input into the neural network.

The autoencoder is structured into two primary components: the encoder and the decoder. The encoder's role is to compress the input images into a more compact latent space representation. This is achieved by applying a series of dense layers with activation functions. In this implementation, the input images are first passed through a dense layer with 128 units and ReLU activation. The output of this layer is then fed into another dense layer with 32 units, creating the latent space where the compressed representation of the images is encoded.

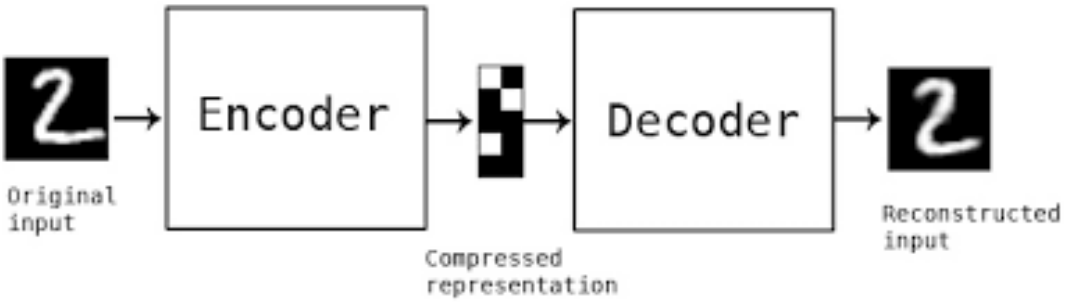
On the flip side, the decoder's task is to reconstruct the original images from the latent space representation. It mirrors the encoder's architecture but in reverse, utilizing dense layers to expand the compressed representation back to the original image size. The latent space representation is processed through a dense layer with 128 units and ReLU activation, followed by another dense layer with 784 units and a sigmoid activation function, which outputs the reconstructed image in the same flattened format as the input.

USN NUMBER: 1RVU22CSE071
NAME: Joselyn Riana Manoj

The autoencoder model is built by combining the encoder and decoder components. It starts with an input layer that accepts vectors of size 784, passes them through the encoder to obtain a latent space representation, and then through the decoder to reconstruct the images. The model is compiled using the Adam optimizer and binary cross-entropy loss function, which is suitable for evaluating the difference between the original and reconstructed images as a pixel-wise binary classification problem.

During training, the autoencoder learns to minimize the reconstruction loss by comparing the original images to their reconstructed counterparts. This process helps the model learn an effective representation of the images in the latent space. After training, the autoencoder's performance can be evaluated by applying it to the test dataset and visualizing the reconstructed images to assess how well the model has learned to reproduce the original images from the compressed representations.

Model:



USN NUMBER: 1RVU22CSE071
NAME: Joselyn Riana Manoj

Building the parts of algorithm

1. Preprocessing Function:

The first step in the algorithm involves preprocessing the MNIST dataset to make it suitable for training the autoencoder. The preprocessing function normalizes the pixel values of the images to a range between 0 and 1. This normalization helps in stabilizing the training process and improving the model's performance. Additionally, the function flattens each 28x28 image into a one-dimensional vector of size 784. This transformation is necessary because the autoencoder expects input in a flat, vectorized format.\

2. Dataset Preparation:

The MNIST dataset is loaded using TensorFlow Datasets (TFDS). The dataset is split into training and testing sets. After loading, the preprocessing function is applied to both the training and testing datasets. The training dataset is further shuffled and batched to improve the training process and ensure that the model sees diverse samples in each epoch. It is also repeated to allow for multiple passes through the data. The test dataset is batched but not shuffled or repeated, as it is used for evaluation purposes.

3. Encoder and Decoder Construction:

The autoencoder consists of two main components: the encoder and the decoder. The encoder is responsible for compressing the input image into a lower-dimensional latent space. It typically involves a series of dense layers with activation functions that progressively reduce the dimensionality of the input. The decoder, on the other hand, reconstructs the image from the latent space representation. It mirrors the encoder's architecture but in reverse, expanding the compressed representation back to the original image dimensions. The decoder uses dense layers to achieve this reconstruction.

4. Model Construction:

Once the encoder and decoder are defined, they are combined to form the complete autoencoder model. The model starts with an input layer that accepts the flattened image vectors. The input is passed through the encoder to obtain the latent space representation. This representation is then fed into the decoder, which outputs the reconstructed image. The autoencoder is thus able to learn a mapping from the input space to a compressed latent space and back to the original space.

5. Model Compilation:

After constructing the autoencoder model, it is compiled with an optimizer and a loss function. The Adam optimizer is commonly used for its efficiency and effectiveness in training neural networks. The binary cross-entropy loss function is chosen because it measures the difference between the original and reconstructed images on a

pixel-by-pixel basis, treating the problem as a binary classification task for each pixel. This setup helps the autoencoder learn to minimize the reconstruction error.

Github Link :
https://github.com/joselynrianaaa/DeepLearning_Labs