#### EXPERIMENT – 3.2

**Mapped Course Outcome**

CO1: Identify and describe soft computing techniques and their roles in building intelligent machines.

**AIM:**  
Write a program to implement an Autoencoder.

**Theory**  
An Autoencoder is a type of artificial neural network used to learn efficient codings of unlabeled data. The aim of an autoencoder is to learn a representation (encoding) for a set of data, typically for dimensionality reduction, by training the network to ignore noise and reconstruct the input data. It consists of an encoder that compresses the input into a latent-space representation and a decoder that reconstructs the input from this representation.

**Procedure:**

**Step 1: Setup and Installation**

1. **Install Anaconda:**
   * Follow the same installation steps as provided in EXPERIMENT – 1.1.
2. **Install Required Libraries:**
   * Open Anaconda Navigator.
   * Ensure that tensorflow, keras, numpy, matplotlib, and pandas are installed.

**Step 2: Implementing the Autoencoder**

1. **Import Necessary Libraries:**

python

Copy code

import numpy as np

import matplotlib.pyplot as plt

from tensorflow.keras.layers import Input, Dense

from tensorflow.keras.models import Model

from tensorflow.keras.datasets import mnist

1. **Load and Preprocess the Dataset:**

Use the MNIST dataset, which contains images of handwritten digits.

python

Copy code

(x\_train, \_), (x\_test, \_) = mnist.load\_data()

x\_train = x\_train.astype('float32') / 255.

x\_test = x\_test.astype('float32') / 255.

x\_train = x\_train.reshape((len(x\_train), np.prod(x\_train.shape[1:])))

x\_test = x\_test.reshape((len(x\_test), np.prod(x\_test.shape[1:])))

1. **Build the Autoencoder Model:**

python

Copy code

input\_dim = x\_train.shape[1]

encoding\_dim = 32 # Dimension of the latent space

# Input layer

input\_img = Input(shape=(input\_dim,))

# Encoder layers

encoded = Dense(encoding\_dim, activation='relu')(input\_img)

# Decoder layers

decoded = Dense(input\_dim, activation='sigmoid')(encoded)

# Autoencoder model

autoencoder = Model(input\_img, decoded)

# Encoder model

encoder = Model(input\_img, encoded)

# Creating a placeholder for the encoded input

encoded\_input = Input(shape=(encoding\_dim,))

# Retrieve the last layer of the autoencoder model

decoder\_layer = autoencoder.layers[-1]

# Decoder model

decoder = Model(encoded\_input, decoder\_layer(encoded\_input))

1. **Compile the Model:**

python

Copy code

autoencoder.compile(optimizer='adam', loss='binary\_crossentropy')

1. **Train the Model:**

python

Copy code

autoencoder.fit(x\_train, x\_train,

epochs=50,

batch\_size=256,

shuffle=True,

validation\_data=(x\_test, x\_test))

1. **Visualize the Results:**

python

Copy code

# Encode and decode some digits

encoded\_imgs = encoder.predict(x\_test)

decoded\_imgs = decoder.predict(encoded\_imgs)

# Plot original and reconstructed images

n = 10

plt.figure(figsize=(20, 4))

for i in range(n):

# Display original

ax = plt.subplot(2, n, i + 1)

plt.imshow(x\_test[i].reshape(28, 28))

plt.gray()

ax.get\_xaxis().set\_visible(False)

ax.get\_yaxis().set\_visible(False)

# Display reconstruction

ax = plt.subplot(2, n, i + 1 + n)

plt.imshow(decoded\_imgs[i].reshape(28, 28))

plt.gray()

ax.get\_xaxis().set\_visible(False)

ax.get\_yaxis().set\_visible(False)

plt.show()

**Step 3: Running the Program**

1. Open Jupyter Notebook from Anaconda Navigator.
2. Create a new Python 3 notebook.
3. Copy and paste the above code sections into the notebook cells.
4. Execute each cell sequentially to build, train, and visualize the Autoencoder.

**Video Tutorial**

[Autoencoders in Keras - Tutorial](https://www.youtube.com/watch?v=G6ogBgJioWk)

**Further Reading**

Rolon-Mérette, D., Ross, M., Rolon-Mérette, T., & Church, K. (2016). Introduction to Anaconda and Python: Installation and setup. Python for research in psychology, 16(5), S5-S11.

**Prospective Viva Questions**

1. Define an Autoencoder and its primary use.
2. Explain the difference between the encoder and decoder parts of an Autoencoder.
3. Discuss the role of the latent-space representation in Autoencoders.
4. Describe how the loss function in an Autoencoder influences the training process.
5. Provide examples of real-world applications where Autoencoders can be effectively used.