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 fashion\_mnist # Import Fashion MNIST dataset  
  
# Load the Fashion MNIST dataset  
(x\_train, \_), (x\_test, \_) = fashion\_mnist.load\_data() # TRAIN AND TESTING  
  
# Normalize pixel values to be between 0 and 1 # MATRIX  
x\_train = x\_train.astype('float32') / 255.0  
x\_test = x\_test.astype('float32') / 255.0  
  
# Flatten the images for the autoencoder # VECTOR  
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:])))  
  
# Define the autoencoder model  
encoding\_dim = 32 # Size of the encoded representations  
input\_img = Input(shape=(784,))  
encoded = Dense(encoding\_dim, activation='relu')(input\_img)  
decoded = Dense(784, activation='sigmoid')(encoded)  
  
autoencoder = Model(input\_img, decoded)  
  
# Compile the autoencoder  
autoencoder.compile(optimizer='adam', loss='binary\_crossentropy')  
  
# Train the autoencoder # HYPERPARAMETER  
autoencoder.fit(x\_train, x\_train, epochs=50, batch\_size=256, shuffle=True, validation\_data=(x\_test, x\_test))  
  
# Create a separate encoder model  
encoder = Model(input\_img, encoded)  
  
# Encode the test images  
encoded\_imgs = encoder.predict(x\_test)  
  
# Decode the encoded images  
decoded\_imgs = autoencoder.predict(x\_test)  
  
# Display original, encoded, and reconstructed images  
n = 10 # Number of images to display  
plt.figure(figsize=(20, 6))  
for i in range(n):  
 # Original images  
 ax = plt.subplot(3, 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)  
  
 # Encoded images  
 ax = plt.subplot(3, n, i + 1 + n)  
 plt.imshow(encoded\_imgs[i].reshape(4, 8)) # Display encoded representation  
 plt.gray()  
 ax.get\_xaxis().set\_visible(False)  
 ax.get\_yaxis().set\_visible(False)  
  
 # Reconstructed images  
 ax = plt.subplot(3, n, i + 1 + 2 \* 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()

Epoch 1/50  
235/235 [==============================] - 3s 10ms/step - loss: 0.4146 - val\_loss: 0.3405  
Epoch 2/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.3261 - val\_loss: 0.3193  
Epoch 3/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.3115 - val\_loss: 0.3079  
Epoch 4/50  
235/235 [==============================] - 3s 14ms/step - loss: 0.3018 - val\_loss: 0.3004  
Epoch 5/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2958 - val\_loss: 0.2957  
Epoch 6/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2920 - val\_loss: 0.2927  
Epoch 7/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2895 - val\_loss: 0.2907  
Epoch 8/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2878 - val\_loss: 0.2893  
Epoch 9/50  
235/235 [==============================] - 3s 12ms/step - loss: 0.2866 - val\_loss: 0.2885  
Epoch 10/50  
235/235 [==============================] - 3s 12ms/step - loss: 0.2858 - val\_loss: 0.2876  
Epoch 11/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2852 - val\_loss: 0.2872  
Epoch 12/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2847 - val\_loss: 0.2867  
Epoch 13/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2843 - val\_loss: 0.2864  
Epoch 14/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2840 - val\_loss: 0.2861  
Epoch 15/50  
235/235 [==============================] - 3s 13ms/step - loss: 0.2838 - val\_loss: 0.2859  
Epoch 16/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2835 - val\_loss: 0.2857  
Epoch 17/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2834 - val\_loss: 0.2856  
Epoch 18/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2832 - val\_loss: 0.2855  
Epoch 19/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2831 - val\_loss: 0.2853  
Epoch 20/50  
235/235 [==============================] - 3s 13ms/step - loss: 0.2829 - val\_loss: 0.2853  
Epoch 21/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2828 - val\_loss: 0.2851  
Epoch 22/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2827 - val\_loss: 0.2850  
Epoch 23/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2826 - val\_loss: 0.2849  
Epoch 24/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2825 - val\_loss: 0.2848  
Epoch 25/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2824 - val\_loss: 0.2847  
Epoch 26/50  
235/235 [==============================] - 3s 13ms/step - loss: 0.2824 - val\_loss: 0.2847  
Epoch 27/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2823 - val\_loss: 0.2848  
Epoch 28/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2822 - val\_loss: 0.2846  
Epoch 29/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2822 - val\_loss: 0.2845  
Epoch 30/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2821 - val\_loss: 0.2845  
Epoch 31/50  
235/235 [==============================] - 3s 14ms/step - loss: 0.2821 - val\_loss: 0.2844  
Epoch 32/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2820 - val\_loss: 0.2844  
Epoch 33/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2820 - val\_loss: 0.2843  
Epoch 34/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2819 - val\_loss: 0.2843  
Epoch 35/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2819 - val\_loss: 0.2843  
Epoch 36/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2818 - val\_loss: 0.2842  
Epoch 37/50  
235/235 [==============================] - 3s 12ms/step - loss: 0.2818 - val\_loss: 0.2842  
Epoch 38/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2818 - val\_loss: 0.2841  
Epoch 39/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2817 - val\_loss: 0.2842  
Epoch 40/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2817 - val\_loss: 0.2842  
Epoch 41/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2817 - val\_loss: 0.2841  
Epoch 42/50  
235/235 [==============================] - 3s 14ms/step - loss: 0.2817 - val\_loss: 0.2840  
Epoch 43/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2816 - val\_loss: 0.2840  
Epoch 44/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2816 - val\_loss: 0.2841  
Epoch 45/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2816 - val\_loss: 0.2840  
Epoch 46/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2816 - val\_loss: 0.2840  
Epoch 47/50  
235/235 [==============================] - 3s 14ms/step - loss: 0.2815 - val\_loss: 0.2840  
Epoch 48/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2815 - val\_loss: 0.2839  
Epoch 49/50  
235/235 [==============================] - 2s 10ms/step - loss: 0.2815 - val\_loss: 0.2839  
Epoch 50/50  
235/235 [==============================] - 2s 9ms/step - loss: 0.2815 - val\_loss: 0.2839  
313/313 [==============================] - 0s 1ms/step  
313/313 [==============================] - 1s 1ms/step

