EXP NO:05

Karan Choudhary

Rollno:07

```
import keras
from keras import layers
from keras.datasets import mnist
import numpy as np
```

```
(x_train, _), (x_test, _)=mnist.load_data()
```

```
Downloading data from
https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 [==============================] - 0s 0us/step
```

```
x_train = x_train.astype("float32")/255.
x_test = x_test.astype("float32")/255.
x_train = x_test.reshape((len(x_train),np.prod(x_train.shape[1:])))
x_test = x_test.reshape((len(x_test),np.prod(x_test.shape[1:])))
print(x_train.shape)
print(x_test.shape)
```

```
(60000, 784)
(60000, 784)
```

```
encoding_dim=32
input_img=keras.Input(shape=(784,))
encoded=layers.Dense(encoding_dim,activation ="relu")(input_img)
decoded=layers.Dense(784,activation ="sigmoid")(encoded)
autoencoder=keras.Model(input_img,decoded)
```

```
encoder=keras.Model(input_img, encoded)



encoded_input =keras.Input(shape=(encoding_dim))
decoder_layer = autoencoder.layers[-1]
decoder=keras.Model(encoded_input,decoder_layer(encoded_input))
```

```
autoencoder.compile(optimizer="adam",loss="binary_crossentropy")
autoencoder.fit(x_train,x_train,
                epochs=20,
                batch_size=64,
                shuffle=True,
                validation_data=(x_test,x_test))
```

```
Epoch 1/20
938/938 [==============================] - 8s 8ms/step - loss: 0.1320 -
val_loss: 0.0052
Epoch 2/20
938/938 [==============================] - 7s 7ms/step - loss: 0.0024 -
val_loss: 0.0011
Epoch 3/20
938/938 [==============================] - 8s 9ms/step - loss: 6.9978e-04
- val_loss: 4.3222e-04
Epoch 4/20
938/938 [==============================] - 7s 7ms/step - loss: 3.0549e-04
- val_loss: 2.1268e-04
Epoch 5/20
938/938 [==============================] - 11s 11ms/step - loss:
1.6060e-04 - val_loss: 1.2028e-04
Epoch 6/20
938/938 [==============================] - 11s 12ms/step - loss:
9.5627e-05 - val_loss: 7.5965e-05
Epoch 7/20
938/938 [==============================] - 8s 9ms/step - loss: 6.3385e-05
- val_loss: 5.3190e-05
Epoch 8/20
938/938 [==============================] - 13s 13ms/step - loss:
4.6518e-05 - val_loss: 4.1073e-05
Epoch 9/20
938/938 [==============================] - 13s 14ms/step - loss:
3.7483e-05 - val_loss: 3.4552e-05
Epoch 10/20
938/938 [==============================] - 11s 12ms/step - loss:
3.2626e-05 - val_loss: 3.1059e-05
Epoch 11/20
938/938 [==============================] - 7s 7ms/step - loss: 3.0039e-05
- val_loss: 2.9213e-05
Epoch 12/20
938/938 [==============================] - 8s 8ms/step - loss: 2.8677e-05
- val_loss: 2.8241e-05
Epoch 13/20
```

```
938/938 [==============================] - 6s 6ms/step - loss: 2.7953e-05
- val_loss: 2.7716e-05
Epoch 14/20
938/938 [==============================] - 8s 8ms/step - loss: 2.7551e-05
- val_loss: 2.7410e-05
Epoch 15/20
938/938 [==============================] - 6s 6ms/step - loss: 2.7307e-05
- val_loss: 2.7217e-05
Epoch 16/20
938/938 [==============================] - 8s 8ms/step - loss: 2.7148e-05
- val_loss: 2.7088e-05
Epoch 17/20
938/938 [==============================] - 8s 8ms/step - loss: 2.7042e-05
- val_loss: 2.7001e-05
Epoch 18/20
938/938 [==============================] - 8s 9ms/step - loss: 2.6969e-05
- val_loss: 2.6940e-05
Epoch 19/20
938/938 [==============================] - 15s 16ms/step - loss:
2.6918e-05 - val_loss: 2.6897e-05
Epoch 20/20
938/938 [==============================] - 14s 15ms/step - loss:
2.6881e-05 - val_loss: 2.6866e-05
<keras.callbacks.History at 0x7aac8ce9a740>
```

```
encoded_imgs=encoder.predict(x_test)
decoded_imgs=decoder.predict(encoded_imgs)
```

```
1875/1875 [==============================] - 20s 10ms/step 1875/1875
[==============================] - 17s 9ms/step
```

```
import matplotlib.pyplot as plt

n=10
plt.figure(figsize=(20, 4))
for i in range(n):
  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)

  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()
```