Date of Performance :

Date of Submission :

EXPERIMENT NUMBER: 5

Aim: Design the architecture and implement the autoencoder model for Image denoising.

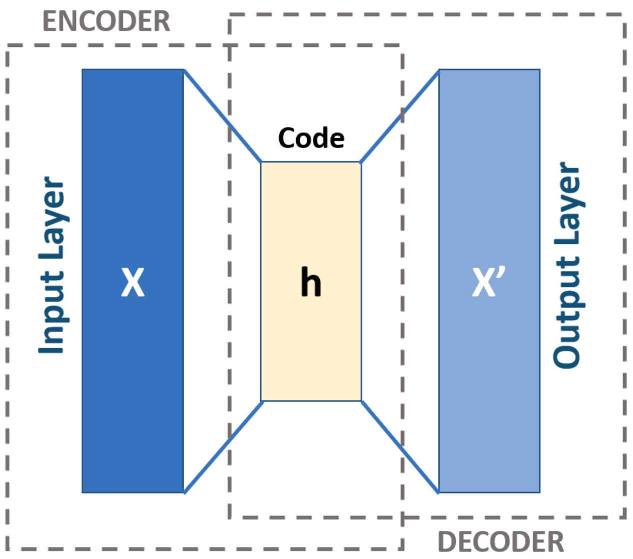
Objective:

- To acquire knowledge of advanced concepts of Auto encoders.

Software Used : Python

Theory:

Auto encoder
An **autoencoder** is a type of artificial neural network used to learn efficient codings of unlabeled data (unsupervised learning). An autoencoder learns two functions: an encoding function that transforms the input data, and a decoding function that recreates the input data from the encoded representation. The autoencoder learns an efficient representation (encoding) for a set of data, typically for dimensionality reduction.



There are 4 hyperparameters that we need to set before training an autoencoder:

- Code size: number of nodes in the middle layer. Smaller size results in more compression.
- Number of layers: the autoencoder can be as deep as we like. In the figure above we have 2 layers in both the encoder and decoder, without considering the input and output.
- Number of nodes per layer: the autoencoder architecture we're working on is called a *stacked autoencoder* since the layers are stacked one after another. Usually stacked autoencoders look like a "sandwitch". The number of nodes per layer decreases with each subsequent layer of the encoder, and increases back in the decoder. Also the decoder is symmetric to the encoder in terms of layer structure. As noted above this is not necessary and we have total control over these parameters.
- Loss function: we either use *mean squared error (mse)* or *binary crossentropy*. If the input values are in the range [0, 1] then we typically use crossentropy, otherwise we use the mean squared error.

Algorithm:

1) Download f`ashion_mnist.load_data() from keras`
2) Splitt data in training and testing part.
3) Plotting Noisy Training Images
4) Add noise to testing images.
5) Train autoencoder model
6) Find binary cross entropy loss. set learning rate and find accuracy.
7) Printing the Denoised Images from Model (Model Output)

Program:

```
import tensorflow as tf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import random
(x_train,y_train),(x_test,y_test)= tf.keras.datasets.fashion_mnist.load_data()
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
labels={0:"T-Shirt",1:"Trousers",2:"Pullover",3:"Dress",4:"Coat",5:"Sandal",6:"T
-Shirt",7:"Sneakers",8:"Bag",9:"Ankle Boot"}
a=np.random.randint(low=0,high=1000,size=100)
fig=plt.figure(figsize=(15,18))
c=1
for i in a:
   fig.add_subplot(10,10,c)
   plt.xticks([])
   plt.yticks([])
   plt.imshow(x_train[i],cmap='gray')
   plt.title(labels[y_train[i]],color="green",fontsize=12)
   c+=1
```

#Data Preprocessing

```
x_train=x_train/255.
x_test=x_test/255.
noise_factor=0.3
noise_train=[]

for img in x_train:
   noisy_img = img + noise_factor* np.random.randn(*img.shape)
   noisy_img = np.clip(noisy_img , 0, 1)
   noise_train.append(noisy_img)
```

#Plotting Noisy Training Images

```
a=np.random.randint(low=0,high=1000,size=50)
fig=plt.figure(figsize=(15,8))
c=1
for i in a:
   fig.add_subplot(5,10,c)
```

```
    plt.xticks([])
    plt.yticks([])
    plt.imshow(noise_train[i],cmap='gray')
    plt.title(labels[y_train[i]],color="green",fontsize=12)
    c+=1
```
#ADD NOISE TO TESTING IMAGES
```
noise_factor=0.3
noise_test=[]

for img in x_test:
    noisy_img = img + noise_factor* np.random.randn(*img.shape)
    noisy_img = np.clip(noisy_img , 0, 1)
    noise_test.append(noisy_img)

a=np.random.randint(low=0,high=1000,size=25)
fig=plt.figure(figsize=(7,8))
c=1
for i in a:
    fig.add_subplot(5,5,c)
    plt.xticks([])
    plt.yticks([])
    plt.imshow(noise_test[i],cmap='gray')
    plt.title(labels[y_test[i]],color="green",fontsize=12)
    c+=1
noise_test=np.array(noise_test)
noise_train=np.array(noise_train)
```
#AUTO-ENCODER MODEL
```
autoencoder= tf.keras.Sequential([

tf.keras.layers.Conv2D(32,(3,3),strides=2,padding='same',input_shape=(28,28,1)),
    tf.keras.layers.Conv2D(16,(3,3),strides=2,padding='same'),
    tf.keras.layers.Conv2D(16,(3,3),strides=1,padding='same'),
    tf.keras.layers.Conv2DTranspose(32,(3,3),strides=2,padding='same'),

tf.keras.layers.Conv2DTranspose(1,(3,3),strides=2,padding='same',activation='sig
moid'),
])
autoencoder.summary()
autoencoder.compile(loss='binary_crossentropy',optimizer=tf.keras.optimizers.Ada
m(learning_rate=0.001),metrics=['accuracy'])
valid_noise=noise_test[:5000,:]
valid_y=x_test[:5000,:]
noise_test = noise_test[5000:,:]
x_test = x_test[5000:,:]
```

```
print(valid_noise.shape,valid_y.shape,noise_test.shape,x_test.shape)
autoencoder.fit(noise_train.reshape(-1,28,28,1),
               x_train.reshape(-1,28,28,1),
               epochs=15,
               batch_size=64,

validation_data=(valid_noise.reshape(-1,28,28,1),valid_y.reshape(-1,28,28,1)))

metric =
autoencoder.evaluate(noise_test.reshape(-1,28,28,1),x_test.reshape(-1,28,28,1))
print("Test Accuracy is {:.2f} and loss is
{:.3f}".format(metric[1]*100,metric[0]))
```

#Printing the Denoised Images from Model (Model Output)

```
a=np.random.randint(low=0,high=1000,size=100)
fig=plt.figure(figsize=(15,18))
c=1
for i in a:
   pred=autoencoder.predict(noise_test[i].reshape(1,28,28,1))
   fig.add_subplot(10,10,c)
   plt.xticks([])
   plt.yticks([])
   plt.imshow(pred.reshape(28,28),cmap='gray')
   plt.title(labels[y_test[5000:][i]],color="green",fontsize=12)
   c+=1
```

#Printing the Original Images (Denoised)

```
fig=plt.figure(figsize=(15,18))
c=1
for i in a:
   fig.add_subplot(10,10,c)
   plt.xticks([])
   plt.yticks([])
   plt.imshow(x_test[i].reshape(28,28),cmap='gray')
   plt.title(labels[y_test[5000:][i]],color="green",fontsize=12)
   c+=1
```

Output:

Sneakers Trousers Ankle Boot T-Shirt T-Shirt T-Shirt Sneakers Sandal Sneakers Ankle Boot
Sneakers Sandal T-Shirt Trousers Pullover Coat Bag Dress Sneakers T-Shirt
T-Shirt Sneakers Dress Ankle Boot Bag Bag Ankle Boot Pullover Bag Ankle Boot
Sneakers Coat T-Shirt Coat Pullover Coat Sandal T-Shirt T-Shirt Ankle Boot
Sneakers Sneakers Trousers T-Shirt T-Shirt Bag Sandal Pullover T-Shirt T-Shirt
Ankle Boot Ankle Boot Sneakers Dress Trousers Sneakers Sneakers Coat Sandal Trousers
Bag Sandal Pullover Sneakers Dress Ankle Boot Trousers T-Shirt Coat T-Shirt
T-Shirt T-Shirt Pullover Trousers Ankle Boot Pullover Sandal T-Shirt Bag Dress
Sandal Dress Dress Bag Ankle Boot Pullover T-Shirt T-Shirt Dress T-Shirt
Sneakers Bag Sandal Coat Trousers Bag Pullover Pullover Sneakers T-Shirt

Conclusion/Outcome:

Thus we have implemented Auto encoder for Image denoising.

Marks & Signature:

R1 (4 Marks)	R2 (4 Marks)	R3 (4 Marks)	R4 (3 Mark)	Total (15 Marks)	Signature