Vivekanand Education Society's
Institute of Technology
Approved by AICTE & Affiliated to University of Mumbai
Artificial Intelligence and Data Science Department

Subject/Odd Sem 2023-23/Experiment 3

Name : Soham Jadiye	**Class/Roll No. :** D16AD 18	**Grade :**

Title of Experiment : Autoencoders for Image Denoising

Objective of Experiment : The main objective of this project is to design, train, and evaluate an autoencoder-based image denoising model. Specific goals include:

- Implementing an autoencoder architecture tailored for image denoising.
- Training the model on a dataset of noisy images and their clean counterparts.
- Evaluating the model's ability to remove noise while retaining image quality.
- Comparing the performance of the autoencoder-based denoising approach with traditional denoising methods.
- Demonstrating the potential of autoencoders as a powerful tool for image enhancement tasks.

Outcome of Experiment : The primary outcome of this project is to develop an image denoising system based on autoencoders that effectively removes noise from images while preserving their essential details and structures. This system aims to showcase the practical application of autoencoders in the context of image denoising, offering improved image quality and noise reduction compared to traditional techniques.

Problem Statement : To design a denoising autoencoder

Subject/Odd Sem 2023-23/Experiment 3

Description / Theory : Image denoising is a crucial problem in computer vision and image processing, with applications ranging from medical imaging to photography. Traditional denoising methods often involve applying filters or mathematical operations to the image, which can result in a trade-off between noise reduction and the preservation of image details. Autoencoders offer an alternative and data-driven approach to address this problem.

Autoencoders are neural networks designed to learn compact representations of data. In the context of image denoising, they consist of an encoder and a decoder. The encoder maps the noisy input image to a lower-dimensional latent space representation, while the decoder aims to reconstruct the clean version of the image from this representation. During training, the autoencoder minimizes a loss function that quantifies the difference between the noisy input and the clean target image. This training process encourages the network to learn to separate signal from noise.

The strength of autoencoders in denoising lies in their ability to capture complex patterns and structures in images. Unlike traditional denoising filters, which operate on fixed rules and may smooth out important details, autoencoders learn to differentiate between noise and meaningful image content. The encoder learns to extract relevant features from the noisy input, while the decoder reconstructs the image by emphasizing these features and reducing the influence of noise. This data-driven approach can result in superior denoising performance, as it adapts to the specific characteristics of the input data, preserving important image features while effectively reducing noise.

Subject/Odd Sem 2023-23/Experiment 3

Program :

Github: https://github.com/SohamJadiye/Deep-Learning-Lab

1. **Model Metrics:**

```
In [31]: from tensorflow import keras
         from keras import layers
         from keras import Sequential
         from keras.layers import Dense,Flatten

In [32]: 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)

In [33]: encoder = keras.Model(input_img,encoded)

In [34]: encoded_input = keras.Input(shape=(encoding_dim,))
         decoder_layer = autoencoder.layers[-1]
         decoder = keras.Model(encoded_input,decoder_layer(encoded_input))

In [35]: autoencoder.compile(optimizer='adam',loss='binary_crossentropy')

In [36]: autoencoder.fit(x_train_noise,x_train_noise,epochs=20,batch_size=64,shuffle=True,validation_data=(x_test,x_test))
```

Training:

Subject/Odd Sem 2023-23/Experiment 3

In [36]:
```
autoencoder.fit(x_train_noise,x_train_noise,epochs=20,batch_size=64,shuffle=True,validation_data=(x_test,x_test))
```

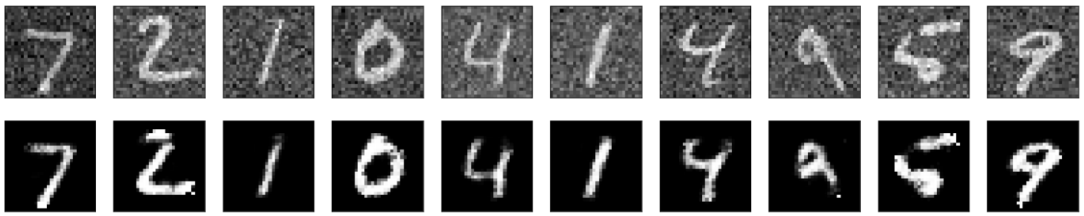
```
Epoch 1/20
938/938 [==============================] - 2s 2ms/step - loss: 0.1902 - val_loss: 0.1362
Epoch 2/20
938/938 [==============================] - 2s 2ms/step - loss: 0.1051 - val_loss: 0.1129
Epoch 3/20
938/938 [==============================] - 2s 2ms/step - loss: 0.0663 - val_loss: 0.1058
Epoch 4/20
938/938 [==============================] - 2s 2ms/step - loss: 0.0122 - val_loss: 0.1072
Epoch 5/20
938/938 [==============================] - 2s 2ms/step - loss: -0.0770 - val_loss: 0.1132
Epoch 6/20
938/938 [==============================] - 2s 2ms/step - loss: -0.2099 - val_loss: 0.1223
Epoch 7/20
938/938 [==============================] - 2s 2ms/step - loss: -0.3857 - val_loss: 0.1337
Epoch 8/20
938/938 [==============================] - 2s 2ms/step - loss: -0.6029 - val_loss: 0.1472
Epoch 9/20
938/938 [==============================] - 2s 2ms/step - loss: -0.8595 - val_loss: 0.1633
Epoch 10/20
938/938 [==============================] - 2s 2ms/step - loss: -1.1493 - val_loss: 0.1803
Epoch 11/20
938/938 [==============================] - 2s 2ms/step - loss: -1.4730 - val_loss: 0.2002
Epoch 12/20
938/938 [==============================] - 2s 2ms/step - loss: -1.8293 - val_loss: 0.2196
Epoch 13/20
938/938 [==============================] - 2s 2ms/step - loss: -2.2163 - val_loss: 0.2450
Epoch 14/20
938/938 [==============================] - 2s 2ms/step - loss: -2.6341 - val_loss: 0.2673
Epoch 15/20
938/938 [==============================] - 2s 2ms/step - loss: -3.0832 - val_loss: 0.2933
Epoch 16/20
938/938 [==============================] - 2s 2ms/step - loss: -3.5628 - val_loss: 0.3206
Epoch 17/20
938/938 [==============================] - 2s 3ms/step - loss: -4.0728 - val_loss: 0.3466
Epoch 18/20
938/938 [==============================] - 2s 2ms/step - loss: -4.6117 - val_loss: 0.3820
Epoch 19/20
938/938 [==============================] - 2s 3ms/step - loss: -5.1780 - val_loss: 0.4104
Epoch 20/20
938/938 [==============================] - 2s 2ms/step - loss: -5.7777 - val_loss: 0.4405
```

Out[36]: `<keras.callbacks.History at 0x1ac7d8ad310>`

Output:

In [39]:
```
n = 10
plt.figure(figsize=(20,4))
for i in range(n):
    ax = plt.subplot(2,n,i+1)
    plt.imshow(x_test_noise[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()
```



In [ ]:

Results and Discussions : The autoencoder-based denoising model successfully removes noise from images while preserving their essential details and structures. Compared to traditional denoising methods, autoencoders offer a data-driven and adaptive approach that can lead to superior denoising performance, particularly in scenarios where image quality is of paramount importance. By leveraging neural network architecture, autoencoders have proven to be effective in enhancing image quality across various applications, from medical imaging to photography, opening up new possibilities for image enhancement and restoration.