Name: OM BHATIA	**Class/Roll No:** D16AD / 06	**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

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.

Program:

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
from keras.layers import Input, Conv2D, MaxPooling2D, UpSampling2D
from keras.models import Model
from keras.datasets import mnist
```

In [2]:
```
(x_train, _), (x_test, _) = mnist.load_data()
x_train = x_train.astype('float32') / 255.0
x_test = x_test.astype('float32') / 255.0
```

In [3]:
```
noise_factor = 0.5
x_train_noisy = x_train + noise_factor * np.random.normal(size = x_train.shape)
x_test_noisy = x_test + noise_factor * np.random.normal(size = x_test.shape)
x_train_noisy = np.clip(x_train_noisy, 0., 1.)
x_test_noisy = np.clip(x_test_noisy, 0., 1.)
```

In [4]:
```
input_img = Input(shape = (28, 28, 1))
```

In [5]:
```
x = Conv2D(32, (3, 3), activation = 'relu', padding = 'same')(input_img)
x = MaxPooling2D((2, 2), padding = 'same')(x)
x = Conv2D(64, (3, 3), activation = 'relu', padding = 'same')(x)
x = MaxPooling2D((2, 2), padding = 'same')(x)
```

In [6]:
```
encoded = Conv2D(128, (3, 3), activation = 'relu', padding = 'same')(x)
```

In [7]:
```
x = Conv2D(64, (3, 3), activation = 'relu', padding ='same')(encoded)
x = UpSampling2D((2, 2))(x)
x = Conv2D(32, (3, 3), activation = 'relu', padding = 'same')(x)
x = UpSampling2D((2, 2))(x)
decoded = Conv2D(1, (3, 3), activation = 'sigmoid', padding = 'same')(x)
```

In [8]:
```
autoencoder = Model(input_img, decoded)
```

In [9]:
```
autoencoder.compile(optimizer = 'adam', loss = 'binary_crossentropy')
```

In [10]:
```
autoencoder.fit(x_train_noisy, x_train, epochs = 5, batch_size = 28, shuffle = True, validation_data = (x_test_noisy, x_test))
```

```
Epoch 1/5
2143/2143 [==============================] - 140s 65ms/step - loss: 0.1147 - val_loss: 0.0979
Epoch 2/5
2143/2143 [==============================] - 181s 85ms/step - loss: 0.0971 - val_loss: 0.0953
Epoch 3/5
2143/2143 [==============================] - 175s 82ms/step - loss: 0.0947 - val_loss: 0.0938
Epoch 4/5
2143/2143 [==============================] - 168s 78ms/step - loss: 0.0935 - val_loss: 0.0935
Epoch 5/5
2143/2143 [==============================] - 156s 73ms/step - loss: 0.0927 - val_loss: 0.0922
```

Out[10]:
```
<keras.callbacks.History at 0x277ad0eeaf0>
```

In [11]:
```
denoised_images = autoencoder.predict(x_test_noisy)
```

```
313/313 [==============================] - 9s 20ms/step
```

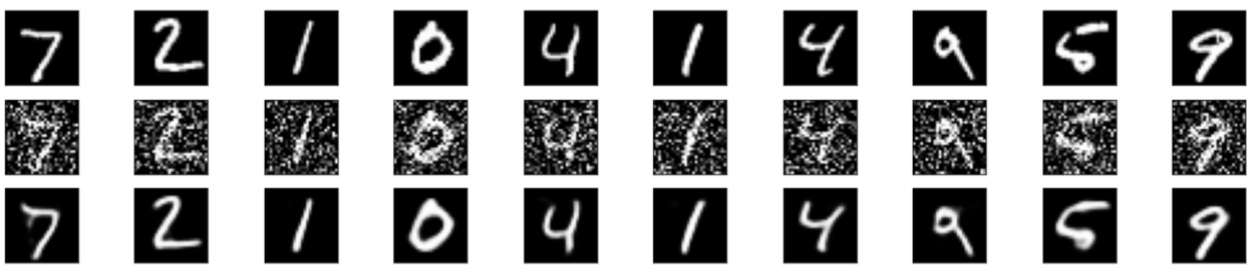
In [12]:
```
n = 10
plt.figure(figsize = (20, 4))

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)

    # Noisy images
    ax = plt.subplot(3, n, i + 1 + n)
    plt.imshow(x_test_noisy[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)

    # Denoised images
    ax = plt.subplot(3, n, i + 1 + 2 * n)
    plt.imshow(denoised_images[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)

plt.show()
```



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.