USN NUMBER: 1RVU22CSE128
NAME: Reethu RG Thota

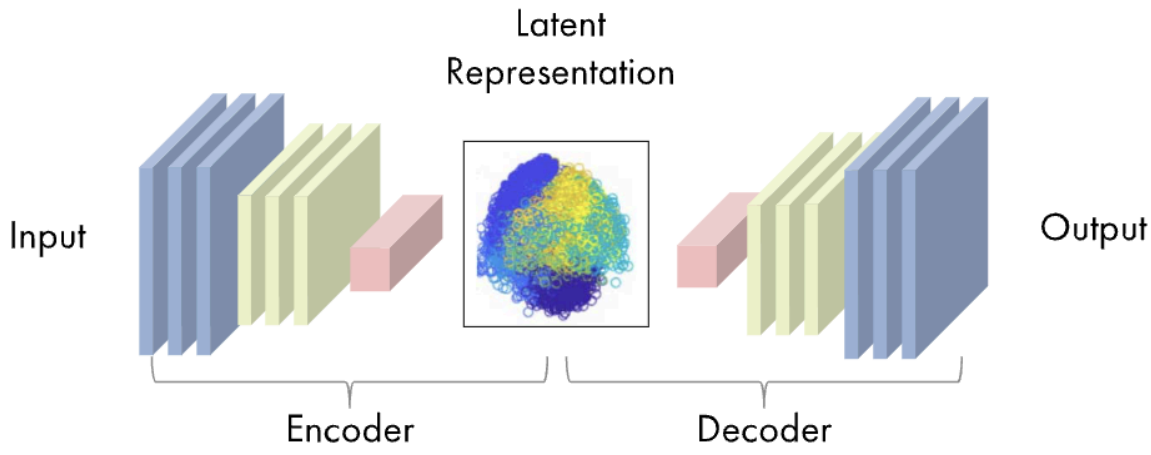
Ex No: 6 **Date:** 11/09/24	**Denoising with a CNN Autoencoder**

Objective

To train a Convolutional Neural Network (CNN) Autoencoder that reconstructs original input images from noisy versions. The model will use the Fashion MNIST dataset, and the goal is to denoise images corrupted with random noise.

Descriptions

In this experiment, we aim to build a CNN-based Autoencoder that takes noisy images as input and attempts to recreate the original images. The model will be trained on the Fashion MNIST dataset, which contains grayscale images of fashion items. The main steps involve normalizing the dataset, introducing noise to the input images, and training the model to reduce this noise.



Autoencoders are neural networks used to learn efficient codings of input data. In this task, the Autoencoder will have an encoder that compresses the input images into a latent space, and a decoder that reconstructs the images from this compressed representation. The loss function aims to minimize the difference between the original and reconstructed images.

USN NUMBER: 1RVU22CSE128
NAME: Reethu RG Thota

Model Architecture

The model consists of two main components:

- **Encoder:** The encoder downsamples the input image into a compressed representation.
- **Decoder:** The decoder reconstructs the original image from the compressed representation.

Steps for building the CNN Autoencoder

The steps involved in building the CNN with transfer learning are:

1. **Install Dependencies:** Install the necessary libraries, including TensorFlow Hub.
2. **Prepare the Dataset:** Fashion MNIST dataset is loaded and prepared by normalizing the images (values between 0 and 1) and adding random Gaussian noise. The noisy images serve as inputs, while the clean images remain the target for reconstruction.

```
def map_image_with_noise(image, label):

 '''Normalizes the images and generates noisy inputs.'''

 image = tf.cast(image, dtype=tf.float32)

 image = image / 255.0

 noise_factor = 0.5

 noise = tf.random.normal(shape=tf.shape(image), mean=0.0, stddev=1.0)

 image_noisy = image + noise_factor * noise

 image_noisy = tf.clip_by_value(image_noisy, 0.0, 1.0)



 return image_noisy, image
```

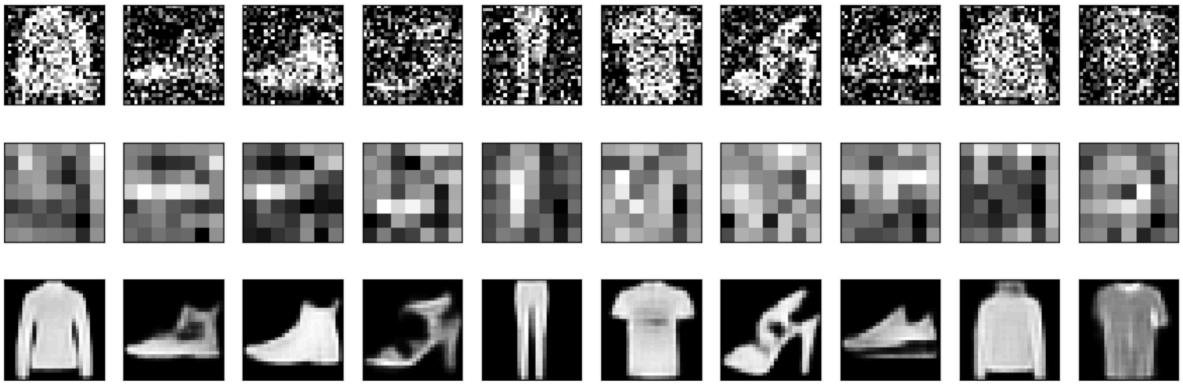
3. **Build the CNN Autoencoder:**
   The model consists of a convolutional encoder that extracts features and a decoder that reconstructs the image from these features.
4. **Train the Model:**
   The model is trained using the noisy images as input and clean images as labels. The Mean Squared Error (MSE) is used as the loss function to measure the reconstruction error.

```
model.compile(optimizer=tf.keras.optimizers.Adam(),
loss='binary_crossentropy')

model.fit(train_dataset, steps_per_epoch=train_steps,
validation_data=test_dataset, validation_steps=valid_steps, epochs=40)
```

USN NUMBER: 1RVU22CSE128
NAME: Reethu RG Thota

Result



GitHub Link

https://github.com/reethuthota/Deep_Learning/tree/main/Lab6