Deep Learning Practical 10 (**PSIT4P3a**)

Aim: Denoising of images using autoencoders.

Durgesh Vishwakarma

MSc IT Sem4 2021-22

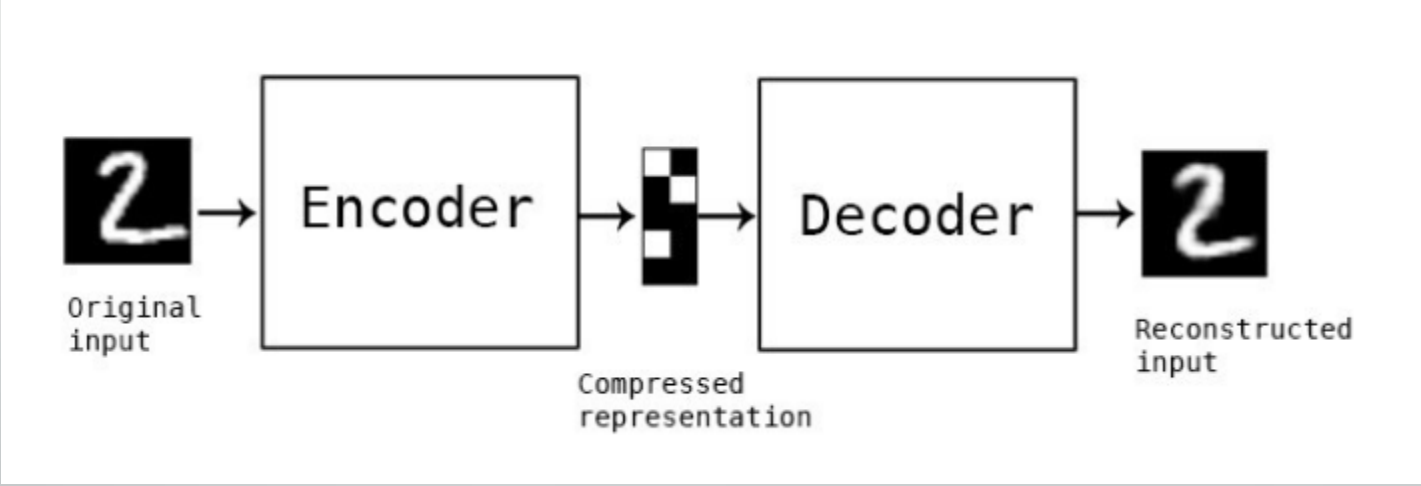
College: B. N. Bandodkar College of Science, Thane.

PRN: 2015430016

Table of Contents

What is an Autoencoder?

- Autoencoder is an **unsupervised artificial neural network** that is trained to copy its input to output. Let's consider that we are given an image, an autoencoder will first encode the image into a lower-dimensional representation, then decodes the representation back to the image.

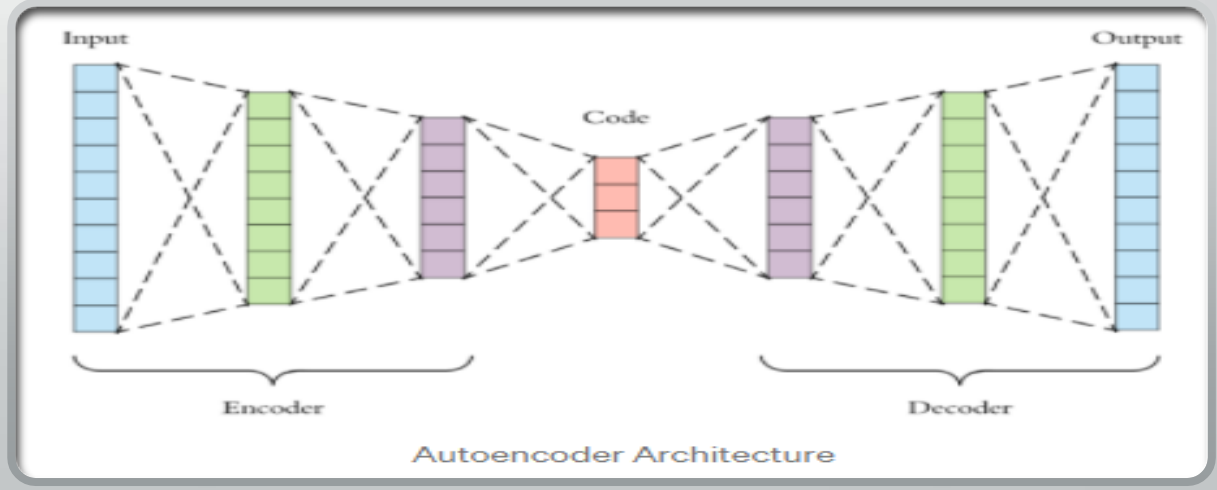


Architecture of Autoencoder

There are mainly 3 parts in autoencoders

1. **Encoder**: In this part of the architecture the model compresses the input data to represent the compressed data in a reduced dimension.
2. **Code**: Also known as Bottleneck this part of the architecture represents the compressed data that is going to be fed to the decoder.
3. **Decoder**: This part reconstructs the encoded data as close to the input data as possible. The output from the decoder is a lossy reconstruction of the original data.

- The goal of an autoencoder is to get an output that is identical to the input. The dimensionality of the input and output is similar as obviously the goal is to get the output as identical to the input we can get.
- They are trained similarly to ANNs via backpropagation



Autoencoder Architecture

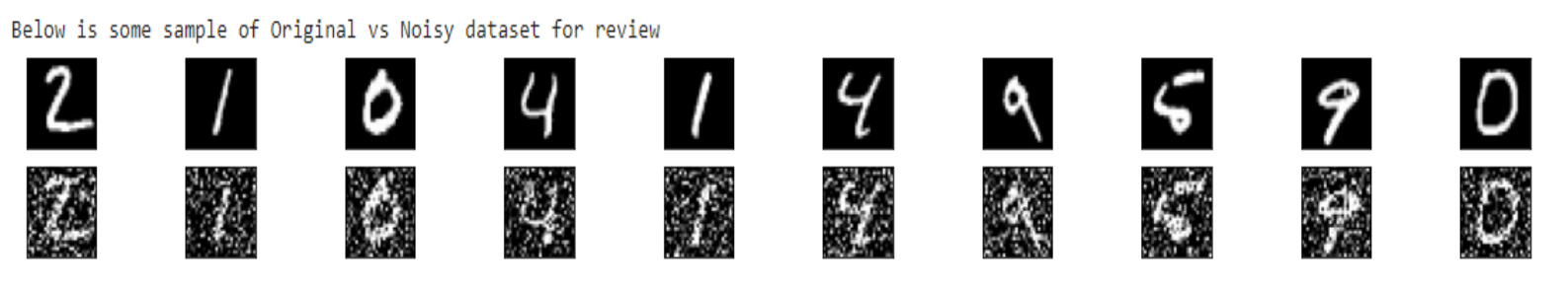
1. Importing libraries and dataset

- In this example we will use some python libraries such as Keras, TensorFlow, NumPy and Matplotlib.
- And for dataset we will use MNIST dataset from TensorFlow.

MNIST is a dataset of black and white handwritten images of size 28x28.

2. Adding Noise to MNIST Image dataset

- In this stage we will add some noise to original MNIST dataset images and our Noisy dataset will look like below after transformation.
- Later we will send this Noisy dataset to autoencoder as input.



3. Building Autoencoder model using Keras

- Here we will build an Autoencoder model using Keras and train it to 100 epochs for better output.
- It may take few minutes to execute and produce output.

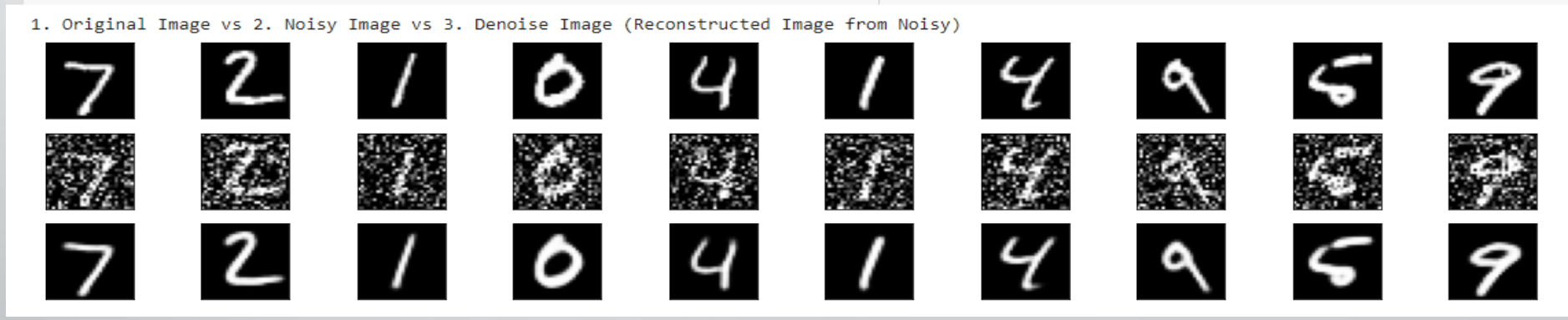
```
Epoch 1/100
469/469 [==============================] - 7s 8ms/step - loss: 0.2495 - val_loss: 0.1147
Epoch 2/100
469/469 [==============================] - 3s 7ms/step - loss: 0.1133 - val_loss: 0.1064
Epoch 3/100
469/469 [==============================] - 3s 7ms/step - loss: 0.1069 - val_loss: 0.1035
Epoch 4/100
469/469 [==============================] - 3s 7ms/step - loss: 0.1040 - val_loss: 0.1015
```

to

```
Epoch 98/100
469/469 [==============================] - 3s 7ms/step - loss: 0.0927 - val_loss: 0.0929
Epoch 99/100
469/469 [==============================] - 3s 7ms/step - loss: 0.0928 - val_loss: 0.0930
Epoch 100/100
469/469 [==============================] - 3s 7ms/step - loss: 0.0929 - val_loss: 0.0930
<tensorflow.python.keras.callbacks.History at 0x7f408e322410>
```

4. Testing Autoencoder model

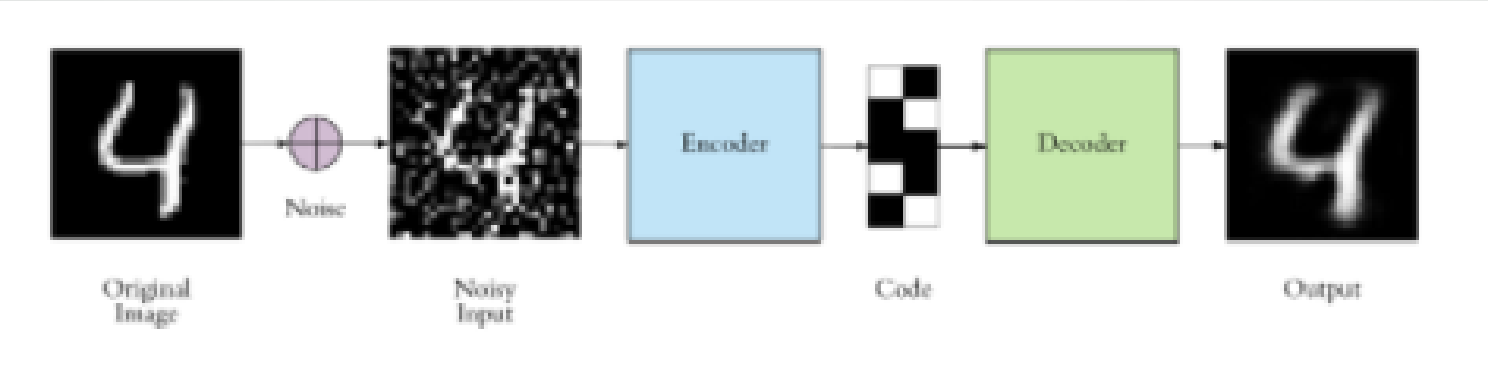
- As our Autoencoder model is ready now. We will compare below to test the model.
  1. Original Image
  2. Noisy Image
  3. Denoised Image



Conclusion

- We have successfully performed following actions in Denoising of images using autoencoders.
  - Pulled Original image dataset from TensorFlow's MNIST dataset.
  - Transformed Original Image into Noisy Image
  - Encoded Noisy Image into code using Keras's autoencoders model.
  - Decoded Code to reconstruct Original Image using Keras's autoencoders model.

- For more details, please review document file of this.



Thank You!