Laboratory Assignment No 4

Aim: Use Autoencoder to implement anomaly detection. Build the model by using:

a. Import required libraries

b. Upload / access the dataset

c. Encoder converts it into latent representation

d. Decoder networks convert it back to the original input

e. Compile the models with Optimizer, Loss, and Evaluation Metrics

Objective: To learn about Autoencoders and developing autoencoder for anomaly detection

Infrastructure: Computer/ Laptop/ Virtual Machine

Software used: Jupyter Notebook/Google Colab, TensorFlow, Keras

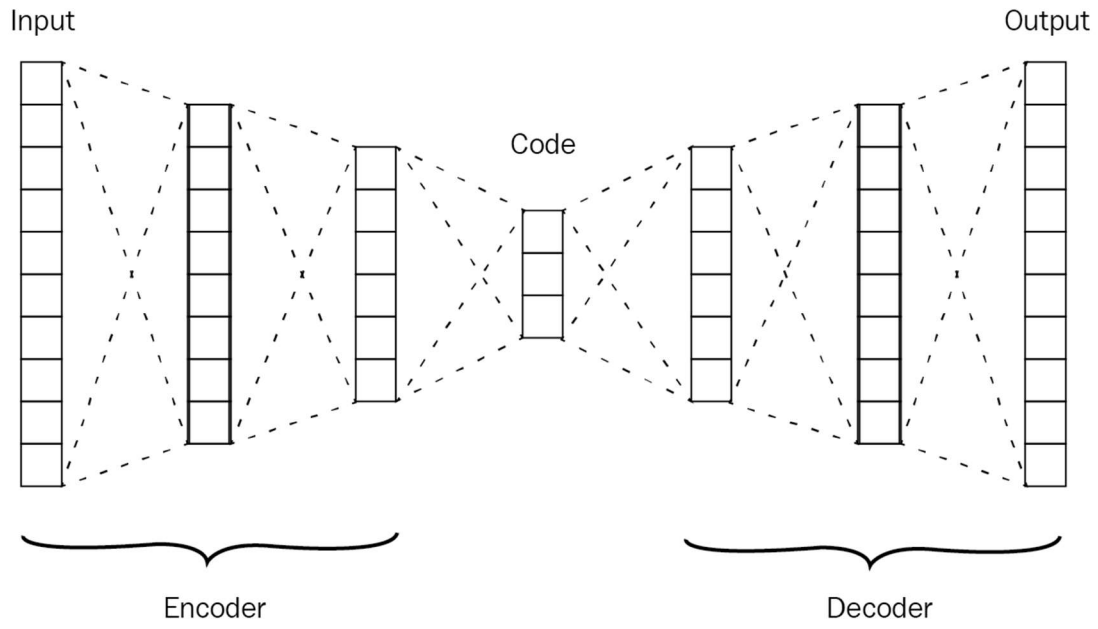
Theory:

Autoencoder

Autoencoder are artificial neural network capable of learning efficient representation of the input data, called coding's, without any supervision. These coding's have typically a much lower dimensionality than the input data, making autoencoders useful for dimensionality reduction and compression. These coding's, the code is a compact "summary" or "compression" of the input.

Autoencoders act as powerful feature detectors and can be used for unsupervised pre-training of deep neural networks. Similarly, they are capable of randomly generating new data that looks very similar to the training data. For example, you can train an autoencoder on picture of faces and it would then be able to generate new faces.

Architecture of Autoencode



An autoencoder consists of 3 components

1. Encoder: - It compresses the input into a latent space representation. The encoder layer encodes the input image as a compressed representation in a reduced dimension. The compressed image is the distorted version of the original image.
2. Code: - This is the compressed input (from encoder) which is fed to decoder for reconstructing the original input later.
3. Decoder: - It decodes the encoded output in form of code, back to the original input. The decoded output is a lossy reconstruction of the original input. The goal is to get an output as identical as was the input.

The layer between the encoder and decoder i.e. the code is also known as Bottleneck. This is a well-designed approach to decide which aspects of the observed data are relevant information and what aspects can be discarded.

Parameters used for training an autoencoder

Four Parameters are: -

1. Code size: - It is number of nodes in the middle(bottleneck) layer. Smaller size results in more compression and it may be difficult to make the size smaller beyond a certain limit to get satisfactorily results.
2. Number of Layers: - The autoencoder can be as deep as you like. They are very similar to an ANN; you only need to decide how many layers autoencoder should have.

3. Number of nodes per layer: Number of nodes per layer decreases with each subsequent layer of the encoder and increases back in the decoder. Also, the decoder is usually symmetric to the encoder in terms of layer structure.
4. Loss Function: - We use either MSE (Mean Squared Error) or Binary cross entropy as the loss function. If the input values are in the range [0,1] then you typically use cross-entropy, otherwise you use the mean squared error.

Implementation:

1. Load the necessary libraries.
2. Import the dataset from the respective library/
3. Shape the data as per your needs.
4. Encode the input data in latent representation.
5. Decode the output of the encoder to convert it back to original input.
6. Use the models with Optimizers, Loss, and Evaluation Metrics.

Conclusion:

We learnt how to detect anomaly using autoencoders.