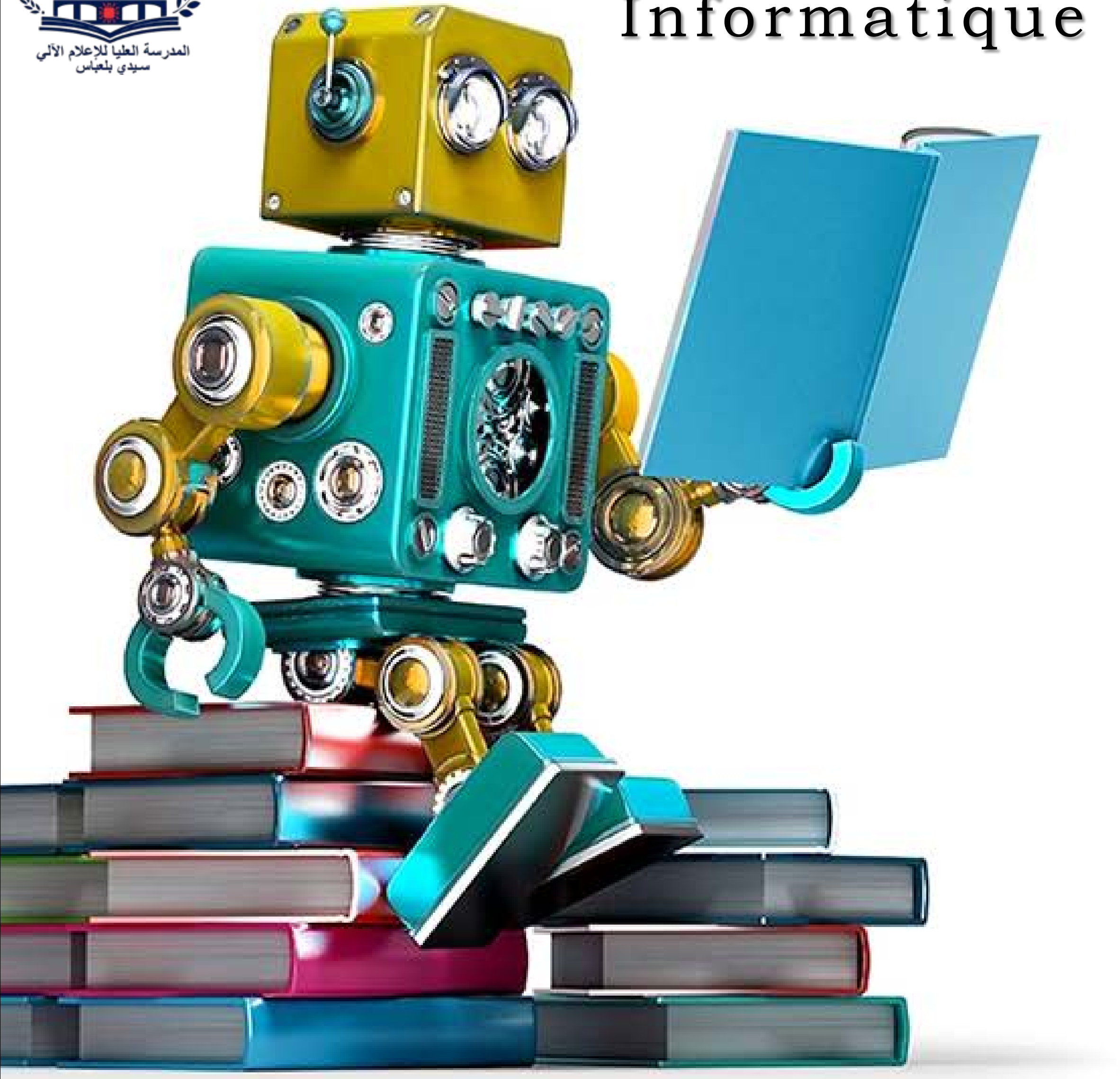
ESI-SBA - École Supérieure en Informatique 08-MAI-1945



DEEP LEARNING

DR N. DIF

PROGRAM

Deep learning

1. **Foundations for DL**
2. **Optimizing Deep Neural Networks**
3. Recurrent Neural Networks
4. Convolution Neural Networks
5. Transfert Learning
6. Dimensionality Reduction with Deep Learning

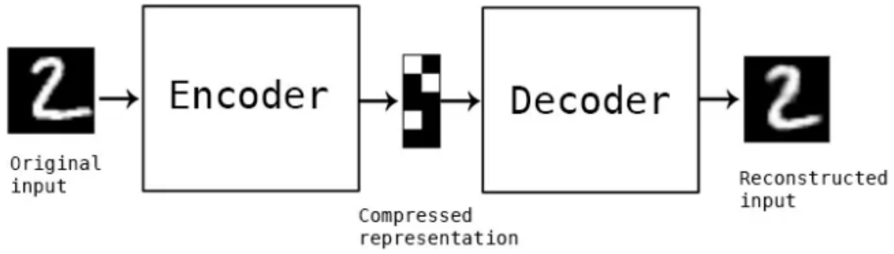
1. AUTO-ENCODERS AND UNSUPERVISED LEARNING

An autoencoder is a type of artificial neural network used to learn **data encodings** in an unsupervised manner.

Neural networks are used for the task of representation learning.

The aim of an autoencoder is to learn a **lower-dimensional representation** (encoding) for **a higher-dimensional data.**

The autoencoder tries to learn a function $h_{w,b} \approx x$. In other words, it is trying to learn an approximation to the identity function, so as to output $\hat{x}$ that is similar to x

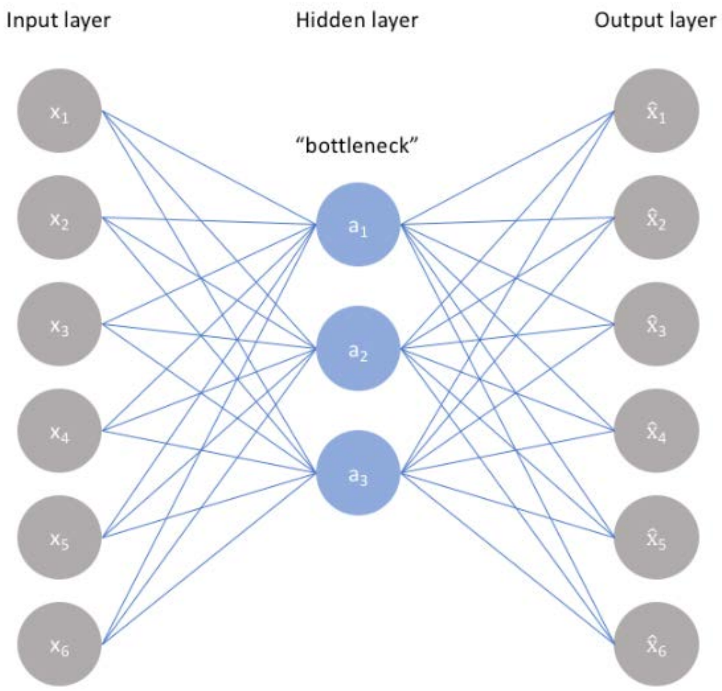


1. AUTO-ENCODERS AND UNSUPERVISED LEARNING

1.1. Structure

Autoencoders consist of 3 parts:

1. Encoder : The encoder **compress** the input to the model into a compact section called the bottleneck. The network is forced to learn a "compressed" representation of the input.
2. Bottleneck : A module that contains the compressed knowledge representations and is therefore the most important part of the network.
3. Decoder : Decompress" the knowledge representations and reconstructs the data back from its encoded form.



1. AUTO-ENCODERS AND UNSUPERVISED LEARNING

1.2. Hyperparameters to train autoencoders?

Size of the bottleneck : decides how much the data has to be compressed.
Number of layers : The depth of the encoder and the decoder.
Number of nodes per layer.
Reconstruction Loss : It depends on the input data and the output. In case of images, the most popular loss functions for reconstruction are MSE Loss and L1 Loss.

1. AUTO-ENCODERS AND UNSUPERVISED LEARNING

1.3. Types of autoencoders

- The first applications date to the 1980s. Initially used for dimensionality reduction and feature learning.

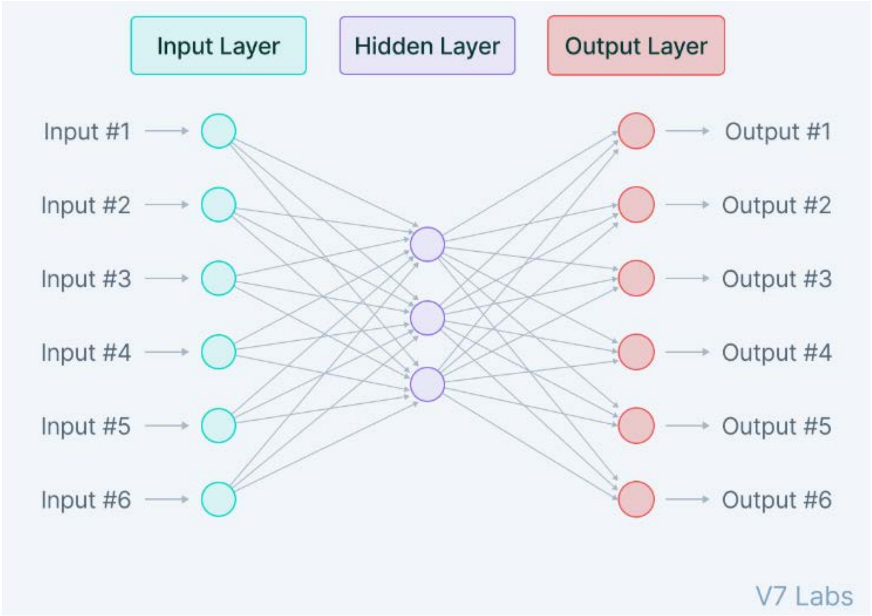
- There are five popular autoencoders :

    1. Undercomplete autoencoders
    2. Sparse autoencoders
    3. Contractive autoencoders
    4. Denoising autoencoders
    5. Variational Autoencoders (for generative modelling)
    6. Stacked Autoencoders.

1. AUTO-ENCODERS AND UNSUPERVISED LEARNING

1.3. Types of autoencoders : Undercomplete autoencoders

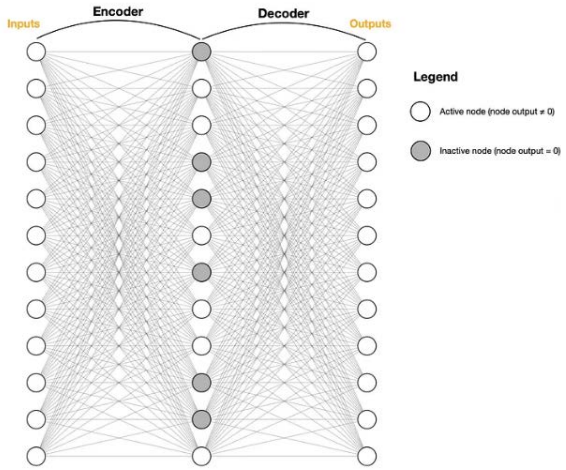
- Undercomplete autoencoder takes in an image and tries to predict the same image as output.
- Truly unsupervised.
- The primary is the generation of the **bottleneck**, so it can be modeled as a form of dimensionality reduction.
- Perform better than PCA because it can learn non-linear relationships.
- It uses called reconstruction loss function as a loss function that check how well the input image have been reconstructed, such as L1 loss $L = \left|X - \hat{X}\right|$.



1. AUTO-ENCODERS AND UNSUPERVISED LEARNING

1.3. Types of autoencoders : Sparse Autoencoders

- Similar to the undercomplete autoencoders, that use **regularization** to overcome **overfitting** by imposing a **sparsity** constraint on the hidden units.
- *Sparse autoencoders work by **penalizing the activation** of some neurons in hidden layers according to a specific input.*
- *This penalty, called **the sparsity function**, prevents the neural network from activating more neurons and serves as a regularizer.*



Inspiration : limit the network's capacity to memorize the input data without limiting the networks capability to extract features from the data.

NOTE : SPASITY

- $\hat{\rho}_j$ presents the average activation of hidden unit j (averaged over the training set).

- $\hat{\rho}_j = \frac{1}{m}\sum_{i=1}^{m}\left[a_j^{(2)}(x^{(i)})\right]$
- We would like to (approximately) enforce the constraint $\hat{\rho}_j = \rho$.
- ρ is a sparsity parameter with a small value close to zero. We would like the average activation of each hidden neuron j to be close to 0.05 (say). To satisfy this constraint, the hidden unit's activations must mostly be near 0.
- **Method** : add an extra penalty term to the optimization objective (cost function) that penalizes $\hat{\rho}_j$ deviating significantly from ρ. This penalty that can be defined by : *Kullback-Leibler (KL)-Divergence.*

(KL)-Divergence :
$$\begin{cases} L_{new} = L(x, \hat{x}) + \beta \sum_{j=1}^{s} KL(\rho || \hat{\rho}_j) \\ KL(\rho || \hat{\rho}_j) = \rho log \frac{\rho}{\hat{\rho}_j} + (1-\rho) log \frac{1-\rho}{1-\hat{\rho}_j} \end{cases}$$

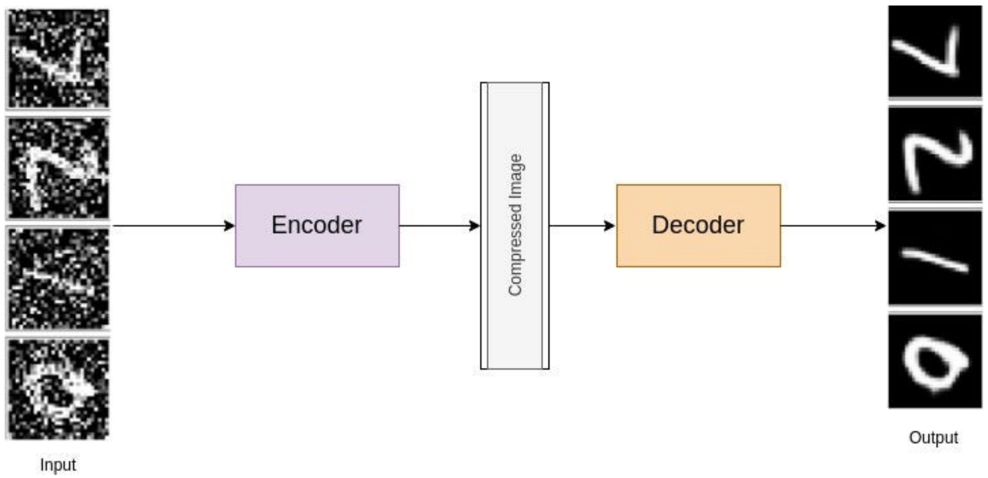
β is a regularization parameter that controls the weights of the sparsity (default : 0.001). S is the number of neurons in the hidden layer.

- The sparsity can also be introduced by L1 regularization

1. AUTO-ENCODERS AND UNSUPERVISED LEARNING

1.3. Types of autoencoders : Denoising Autoencoders

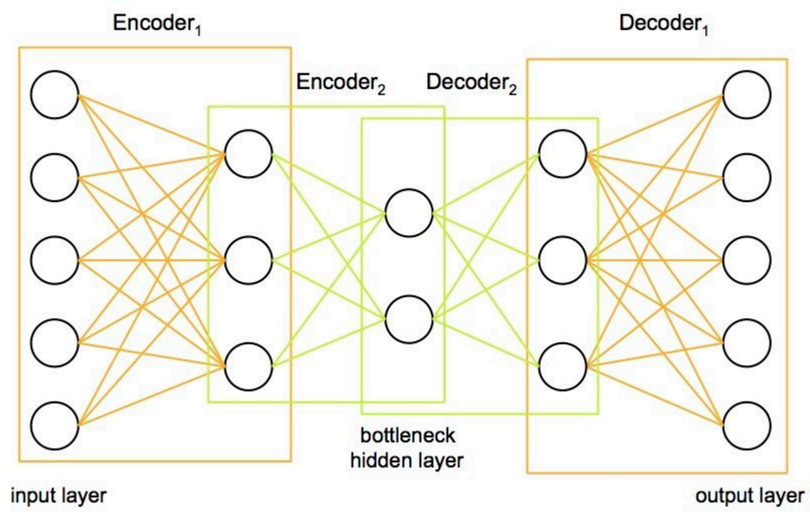
- Autoencoders that remove noise from an image.
- *The input image and its ground truth are different.*
- *It feeds a noisy version of the image, where noise has been added via digital alterations. The noisy image is fed to the encoder-decoder architecture, and the output is compared with the ground truth image.*



1. AUTO-ENCODERS AND UNSUPERVISED LEARNING

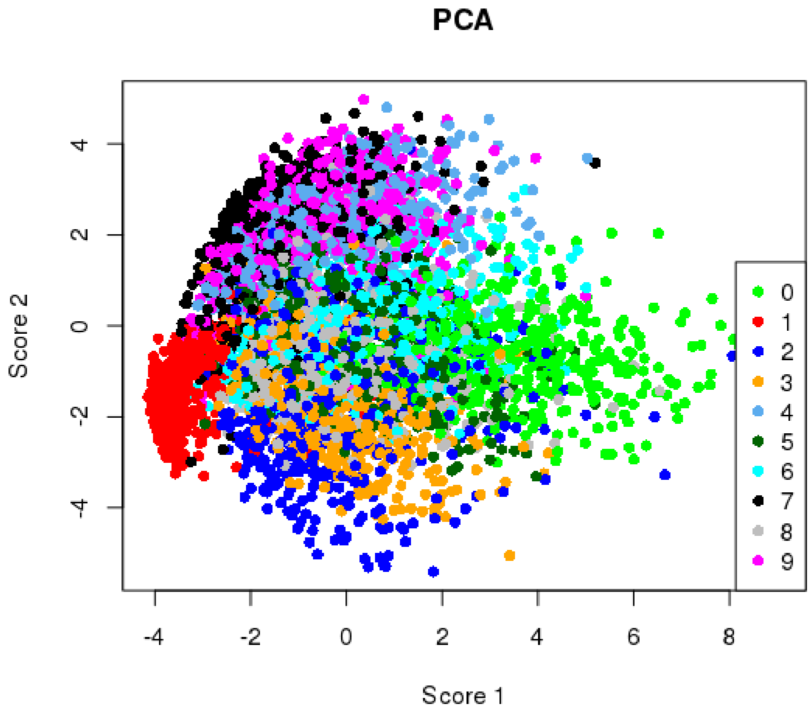
1.3. Types of autoencoders : Stacked autoencoders

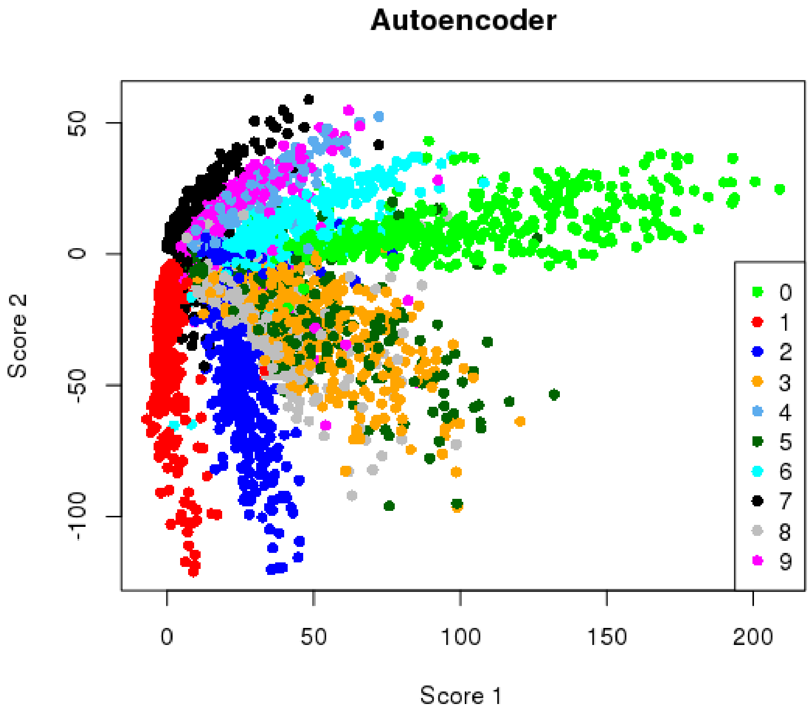
- The stacked autoencoders are, as the name suggests, multiple encoders stacked on top of one another.
- Used for dataset that have a complex relationship within the features.



1. AUTO-ENCODERS AND UNSUPERVISED LEARNING

1.4. Applications : Dimensionality reduction



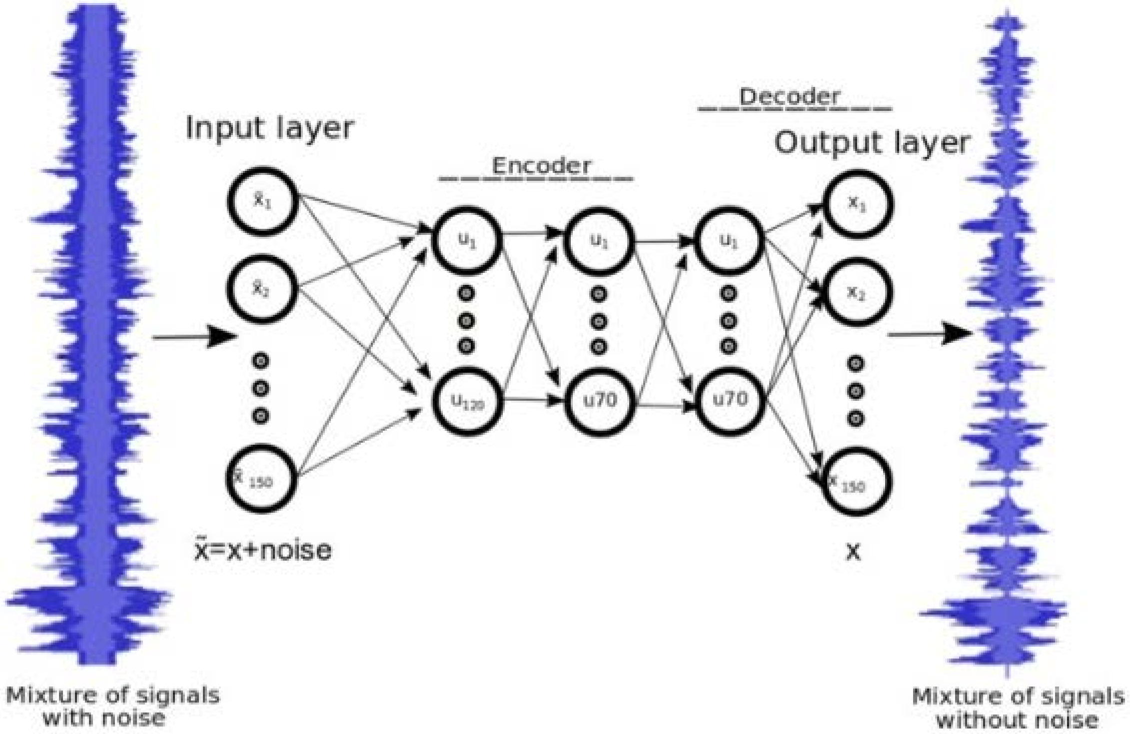


Source : https://iq.opengenus.org/dimensional-reduction-using-autoencoder/

1. AUTO-ENCODERS AND UNSUPERVISED LEARNING

1.4. Applications : Noise reduction

Signal Speech Reconstruction and Noise removal using Convolutional Denoising Autoencoders with Neural Deep Learning



2. GENERATIVE ADVERSARIAL NETWORKS (GANS)

2.1. Definition

GAN is basically an approach to generative modeling that generates a new set of data based on training data that look like training data. GANs have two main blocks(two neural networks) which compete with each other and are able to capture, copy, and analyze the **variations in a dataset**

Generative

Creates fake data

Adversarial

Generator and discriminator, each competing to win. Generator trying to generate fake data and Discriminator, not to be fooled.

Network

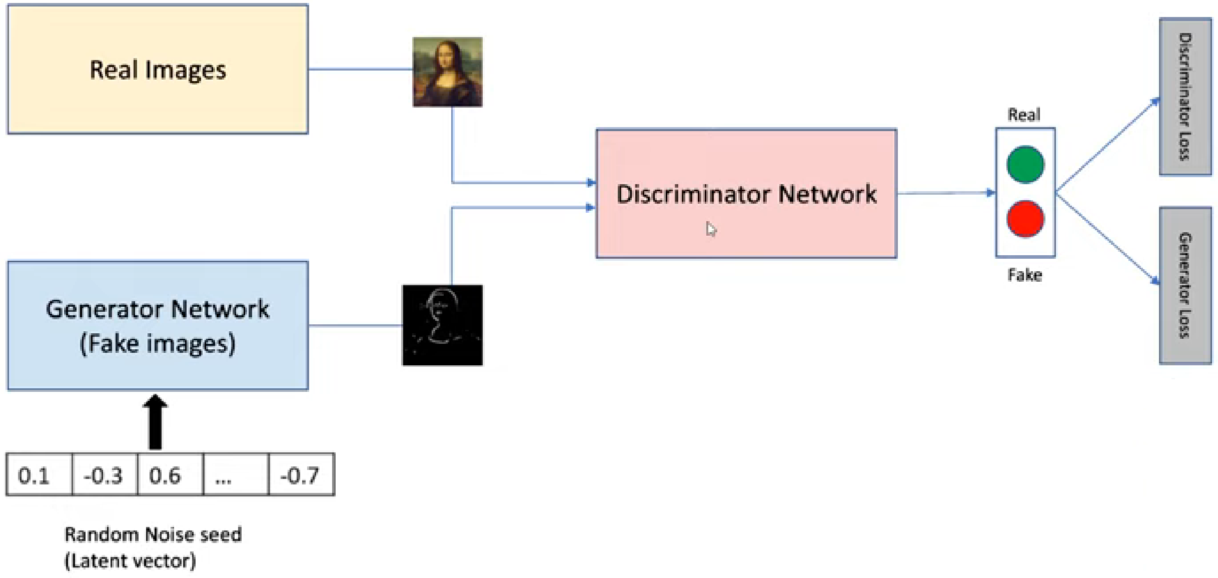
Use deep neural networks as artificial intelligence (AI) algorithms for training purposes

2. GENERATIVE ADVERSARIAL NETWORKS (GANS)

2.2. Architecture

Generator : It is an unsupervised learning approach. It will generate data that is fake data based on original(real) data. It generates the fake image based on feedback of the discriminator. When the discriminator is made a fool by the generator, the training stops.

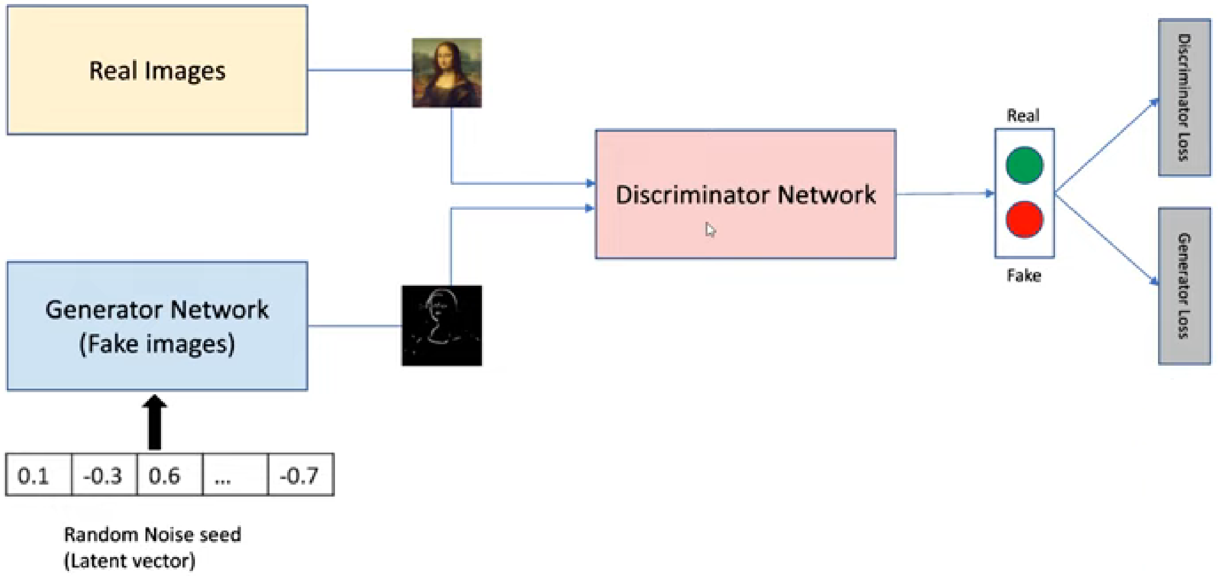
Discriminator : It is a supervised approach means It is a simple classifier that predicts data is fake or real. It is trained on real data and provides feedback to a generator.



2. GENERATIVE ADVERSARIAL NETWORKS (GANS)

2.2. Architecture

1. Define the GAN architecture.
2. Train discriminator on real dataset.
3. Generate fake inputs from the generator.
4. Train discriminator on fake data.
5. Train generator on new fake inputs.
6. Repat 1,2,3,4 seps for multiple epochs.
7. Save generator model to create new fake data



The Generator is trained while the Discriminator is freeze. After the Discriminator is trained by the generated fake data of the Generator.

2. GENERATIVE ADVERSARIAL NETWORKS (GANS)

2.3. Types of GANs

1. **Vanilla GAN:** This is the simplest type of GAN. Here, the Generator and the Discriminator are simple multi-layer perceptron.
2. **Deep Convolutional GAN (DCGAN):** DCGAN is one of the most popular and also the most successful implementations of GAN. It is composed of ConvNets in place of multi-layer perceptron.
3. **Conditional GAN (CGAN)**
4. **Laplacian Pyramid GAN (LAPGAN)**

2. GENERATIVE ADVERSARIAL NETWORKS (GANS)

2.3. Applications of GANs

1.Generate new data from available data – It means generating new samples from an available sample that is not similar to a real one.
2.Generate realistic pictures of people that have never existed.
3.Gans is not limited to Images, It can generate text, articles, songs, poems, etc.
4.Text to Image Generation (Object GAN and Object Driven GAN)
5.Interactive Image Generation – It means that GANs are capable to generate images and video footage in an art form if they are trained on the right real dataset.