

Neural Network-II

Different ANNs architectures

Feedforward Neural Networks (FFNN)

Convolutional Neural Networks (CNN)

Recurrent Neural Networks (RNN)

Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs)

What are Generative Adversarial Networks (GANs)

Generator

Discriminator

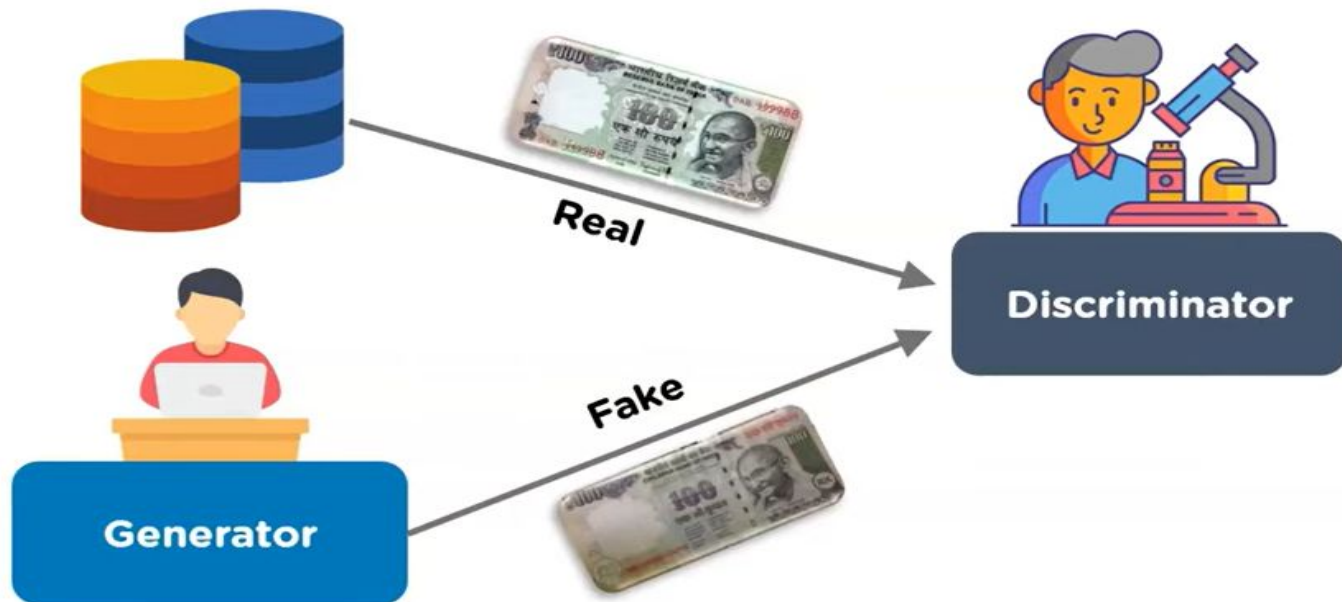
How GANs works

Types of GANs

Application of GANs

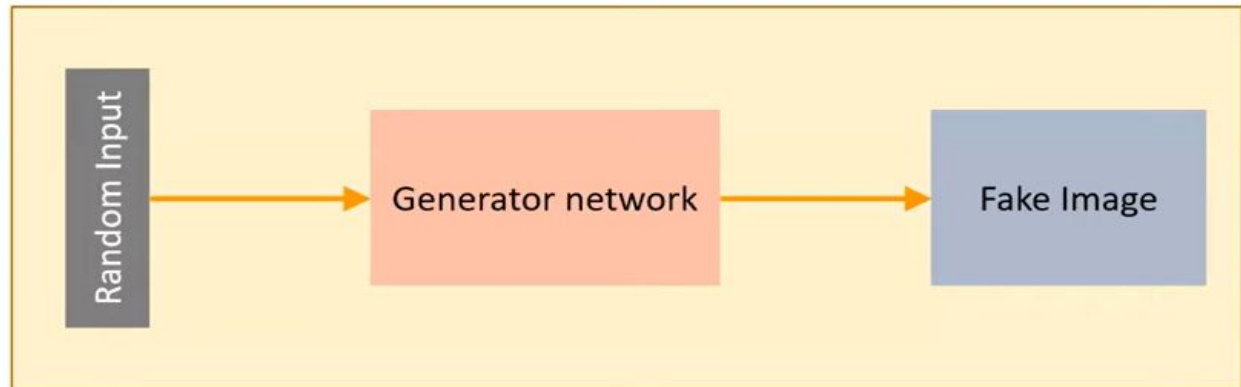
What are Generative Adversarial Networks?

Generative Adversarial Networks consist of two models that compete with each other to analyze, capture and copy the variations within a dataset

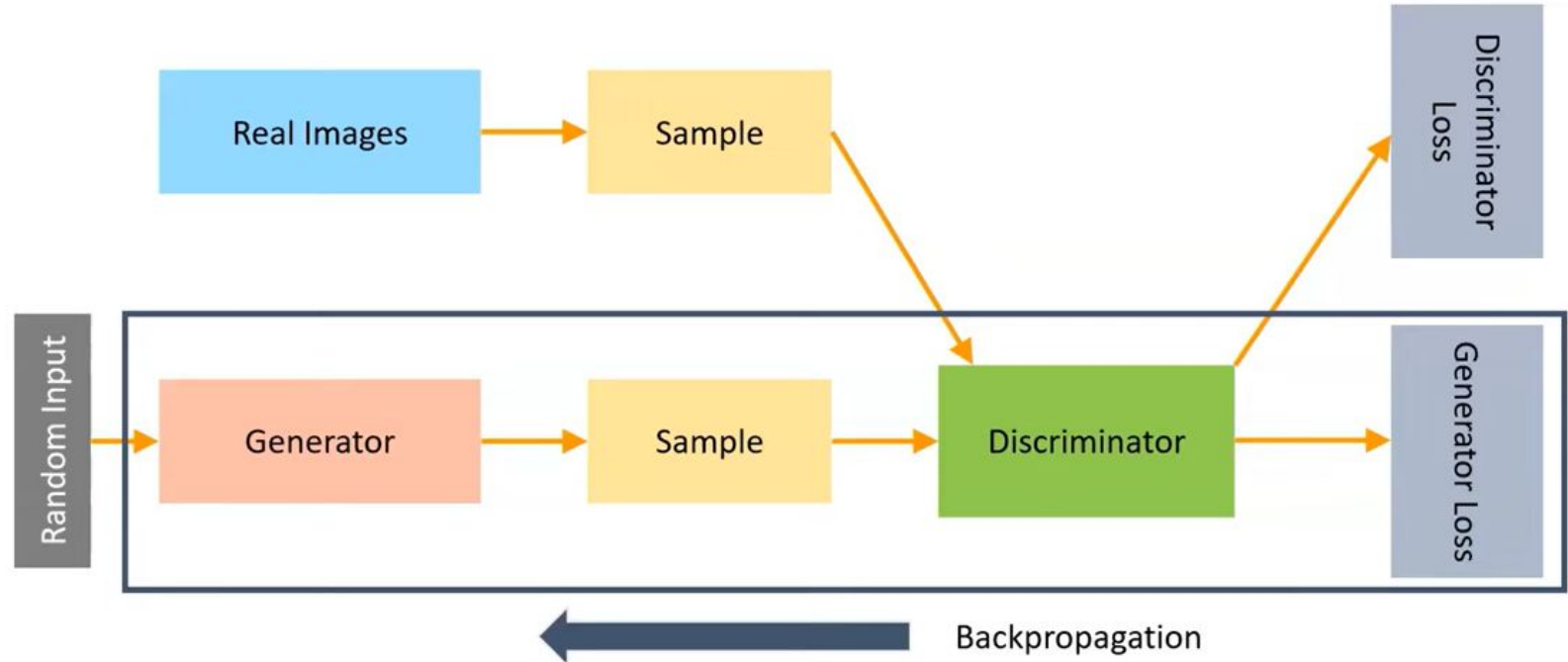


Generator

The Generator in GAN learns to create fake data by incorporating feedback from the discriminator

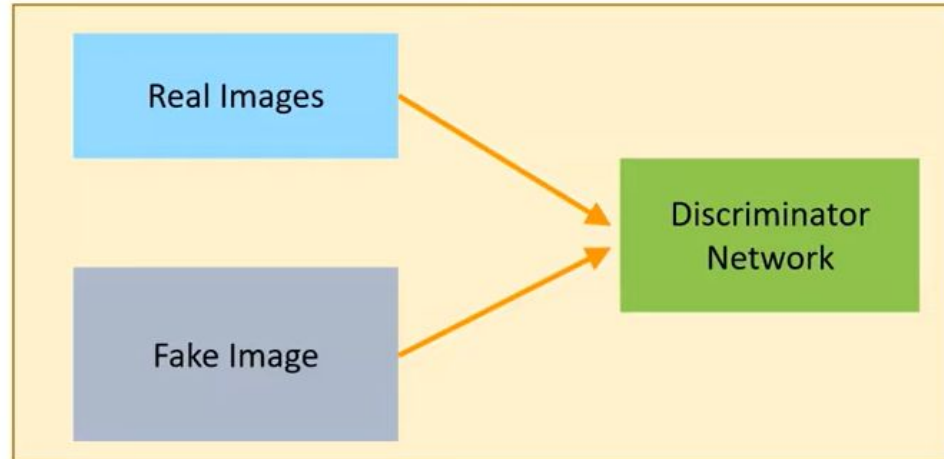


Generator Training

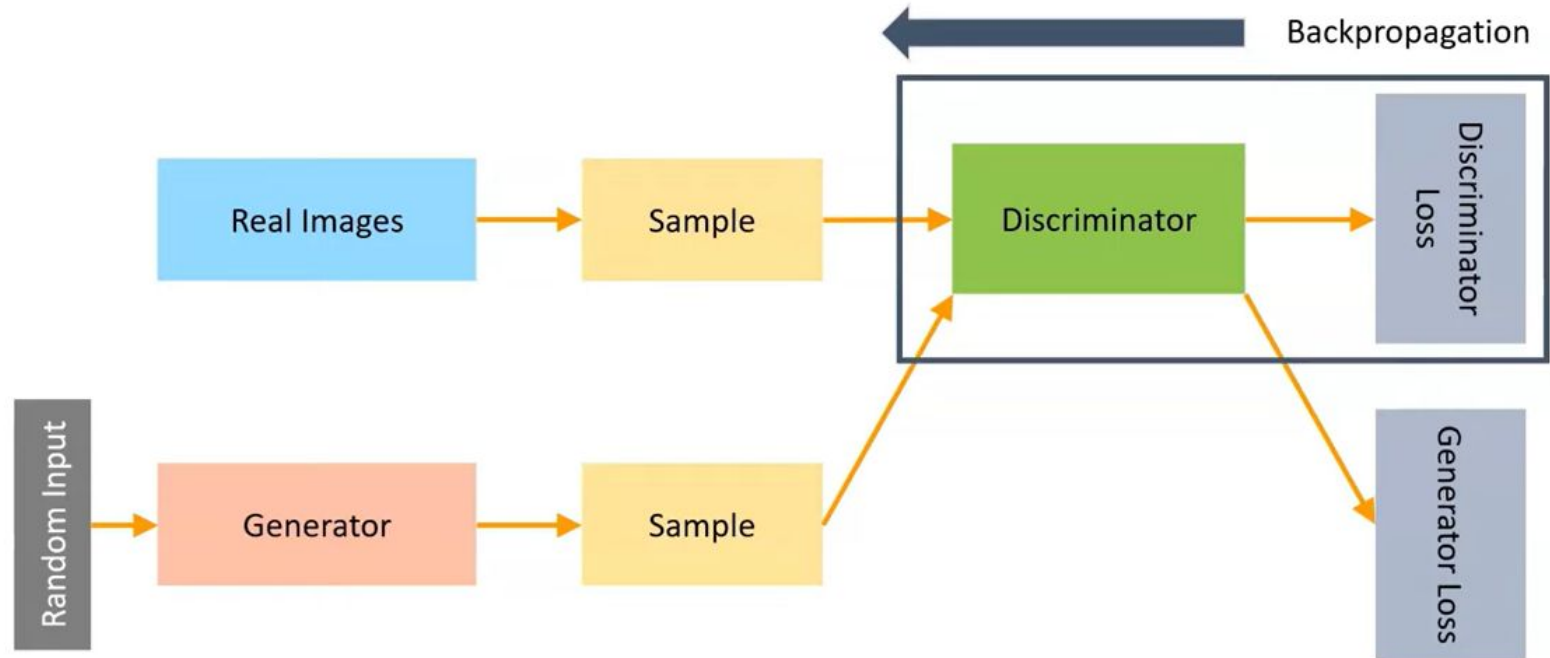


Discriminator

The Discriminator in GAN is a classifier that identifies real data from the fake data created by the Generator

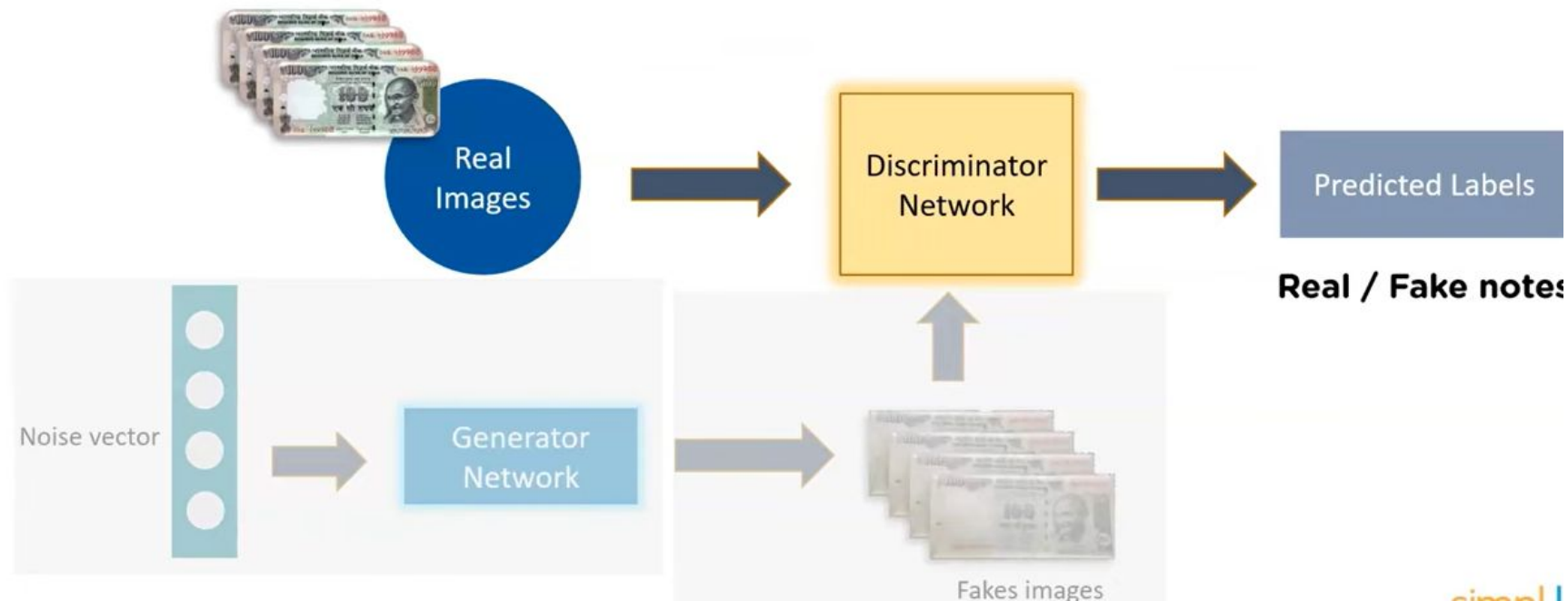


Discriminator Training



How GANs Work?

The Discriminator estimates the probability that the sample it got is from the training data and not from the Generator



How GANs Work?

The mathematical formula for working on GANs can be represented as:

$$V(D, G) = E_{x \sim P_{data}(x)} [\log D(x)] + E_{z \sim p(z)} [\log(1 - D(G(z)))]$$

Where,

G = Generator

D = Discriminator

Pdata(x) = distribution of real data

p(z) = distribution of generator

x = sample from Pdata(x)

z = sample from P(z)

D(x) = Discriminator network

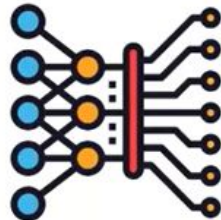
G(z) = Generator network

How GANs Work?

Steps for training GAN

- **Define the problem**
- **Choose the architecture of GAN**
- **Train Discriminator on real data**
- **Generate fake data for Generator**
- **Train Discriminator on fake data**
- **Train Generator with the output of Discriminator**

Types of GANs



Vanilla GANs

Simplest type of GAN where the Generator and Discriminator are simple multi-layer perceptrons

Deep Convolutional GANs (DCGANs)

DCGANs comprise of ConvNets and are more stable and generate higher quality images

Prediction of Next Frame In A Video



Text To Image Generation

A Flower With **Red** Petals And **Green** Leaves



Image To Image Translation



Real



Generated

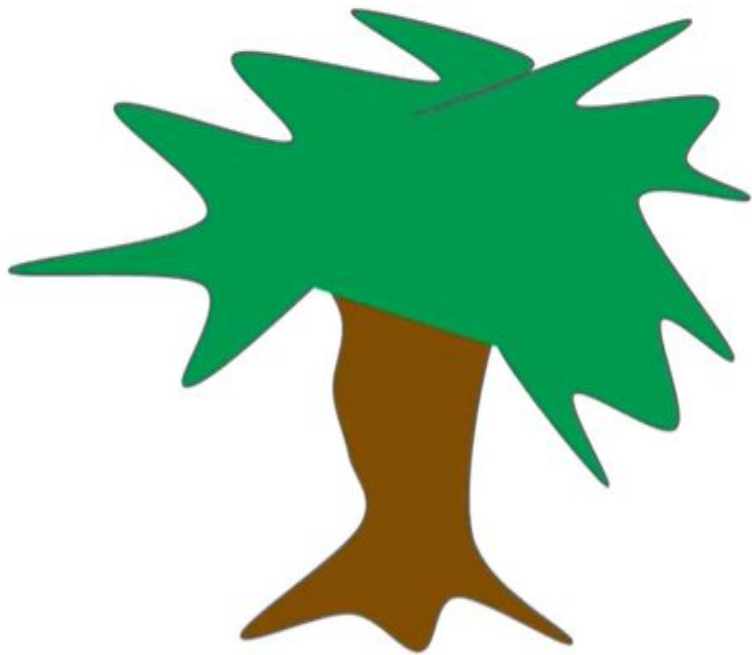


Reconstructed

Enhancing the Resolution Of An Image



Interactive image generation



Feedforward Neural Networks (FFNN)

(FFNN) is a type of artificial neural network where the **information flows in one direction** only, from the input layer through one or more hidden layers to the output layer.

The output of each layer is connected to the input of the next layer, and the weights and biases of the connections are learned during the training process.

FFNNs are commonly used for tasks such as **classification, Control systems such as robotics , and pattern recognition.**

They can be **trained using supervised learning**, where the training data consists of input-output pairs, and the network learns to map inputs to outputs.

The weights and biases of the network are updated during the training process using **backpropagation**, which is an algorithm that computes the gradients of the loss function with respect to the weights and biases.

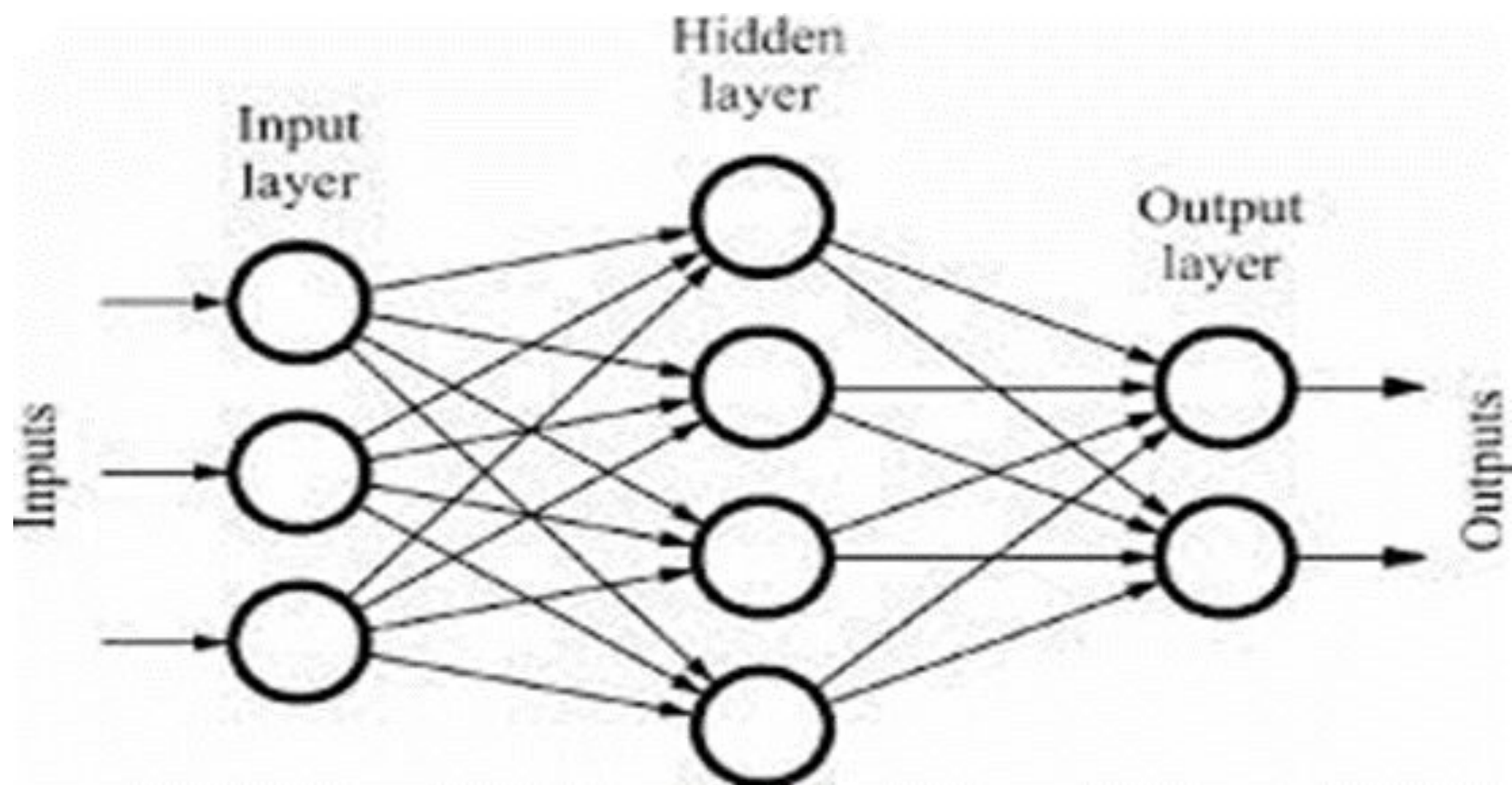
How FFNN works

The input layer of an FFNN takes in the input data, which is usually **in the form of a vector**.

Passes it through a series of **hidden layers**, each consisting of a set of neurons.

Each neuron in a hidden layer takes in the **weighted sum of the outputs from the previous layer**, **adds a bias term**, and applies an **activation function** to produce an output that is passed to the next layer.

The output layer produces the **final output of the network**, which is usually a prediction or a classification.



Convolutional Neural Networks (CNN)

A Convolutional Neural Network (CNN) is a type of Deep Learning architecture commonly used for image classification and recognition tasks.

- Before diving into the Convolution Neural Network, let us first revisit some concepts of Neural Network. In a regular Neural Network, there are three types of layers:
 - Input Layers
 - Hidden Layer
 - Output Layer
- CNN consists of multiple layers
 - Convolutional layers
 - Pooling layers
 - Fully connected layers.

9



Location shifted

1	1	1	-1	-1
1	-1	1	-1	-1
1	1	1	-1	-1
-1	-1	1	-1	-1
-1	-1	1	-1	-1
-1	1	-1	-1	-1
1	-1	-1	-1	-1



1	1	1	-1	-1
1	-1	1	-1	-1
1	1	1	-1	-1
-1	-1	1	-1	-1
-1	-1	1	-1	-1
-1	1	-1	-1	-1
1	-1	-1	-1	-1



-1	1	1	1	-1
-1	1	-1	1	-1
-1	1	1	1	-1
-1	-1	-1	1	-1
-1	-1	-1	1	-1
-1	-1	1	-1	-1
-1	1	-1	-1	-1

To handle **variety** in digits we can use simple artificial neural network (ANN)





Image size = $1920 \times 1080 \times 3$

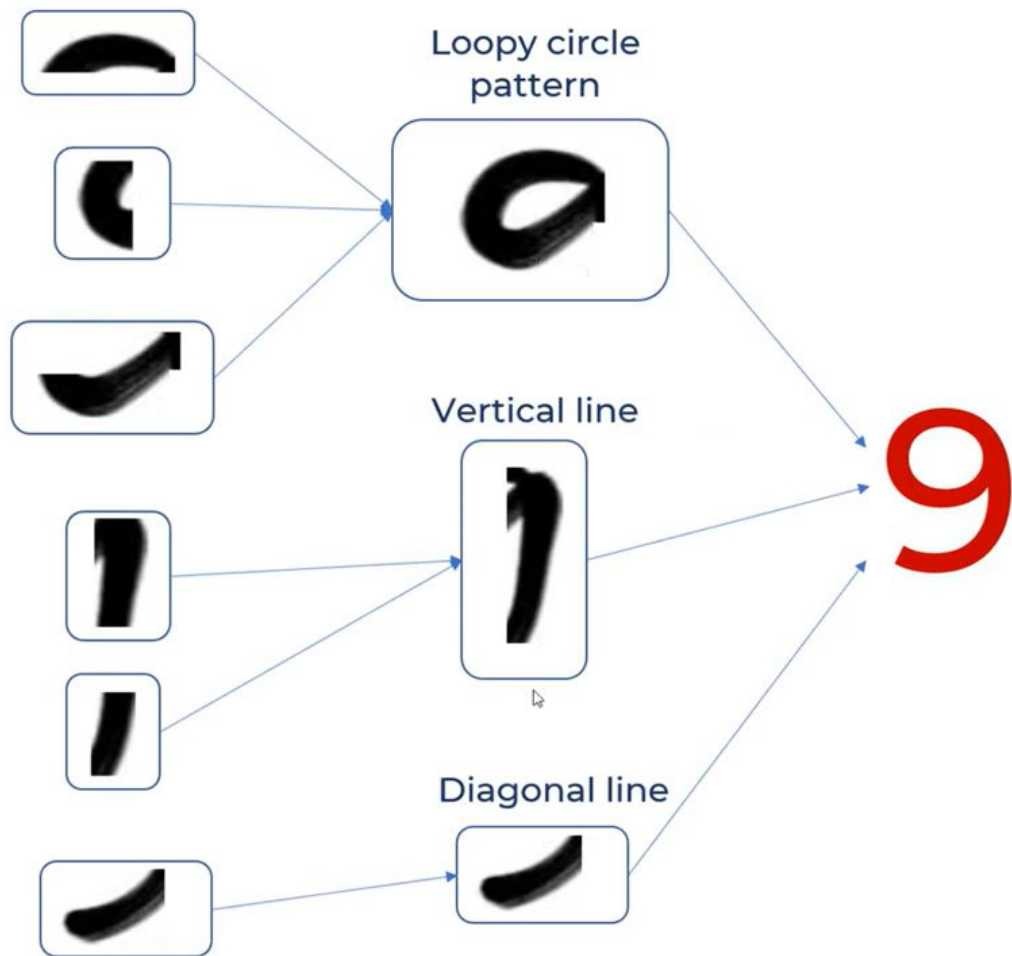
First layer neurons = $1920 \times 1080 \times 3 \sim 6 \text{ million}$



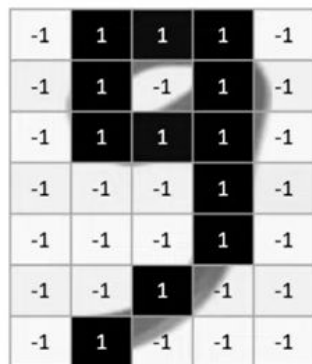
Hidden layer neurons = Let's say you keep it $\sim 4 \text{ million}$

Weights between input and hidden layer = $6 \text{ mil} * 4 \text{ mil}$
= 24 million

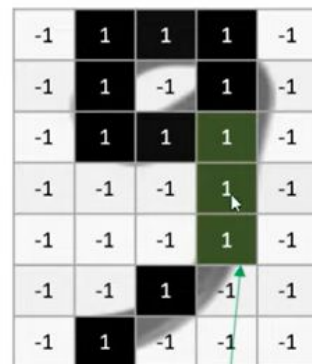
So how Human recognize
these images easily???



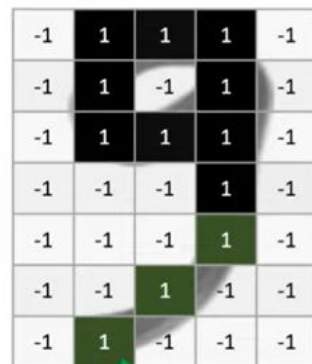
How can we make computers
recognize these tiny
features???



Loopy pattern
filter



Vertical line
filter



Diagonal line
filter

-1	1	1	1	-1
-1	1	-1	1	-1
-1	1	1	1	-1
-1	-1	-1	1	-1
-1	-1	-1	1	-1
-1	-1	1	-1	-1
-1	1	-1	-1	-1

*

1	1	1
1	-1	1
1	1	1

-0.11	1	-0.11
-0.55	0.11	-0.33
-0.33	0.33	-0.33
-0.22	-0.11	-0.22
-0.33	-0.33	-0.33

Feature Map


Loopy pattern detector

$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$

$=$

	1			

Loopy pattern detector

 *

1	1	1
1	-1	1
1	1	1

 =

	1	

Loopy pattern detector

$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$

$=$

	1	
	1	

[illegible]

Filter are nothing but feature
detectors

Loopy pattern detector

$$\begin{array}{c} \text{9} \end{array} * \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & -1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline & 1 & \\ \hline & & \\ \hline & & \\ \hline & & \\ \hline & & \\ \hline \end{array}$$

Vertical line detector

$$\begin{array}{c} \text{9} \end{array} * \begin{array}{|c|c|c|} \hline -1 & 1 & -1 \\ \hline -1 & 1 & -1 \\ \hline -1 & 1 & -1 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline & & \\ \hline & & \\ \hline & 1 & \\ \hline & & \\ \hline & & \\ \hline \end{array}$$

Diagonal line detector

$$\begin{array}{c} \text{9} \end{array} * \begin{array}{|c|c|c|} \hline -1 & -1 & 1 \\ \hline -1 & 1 & -1 \\ \hline 1 & -1 & -1 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline & & \\ \hline & & \\ \hline & & \\ \hline & 1 & \\ \hline & & \\ \hline \end{array}$$



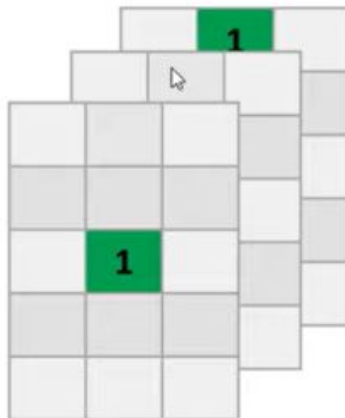
*

Filters



=

Feature Maps



-1	1	1	1	-1
-1	1	-1	1	-1
-1	1	1	1	-1
-1	-1	-1	1	-1
-1	-1	-1	1	-1
-1	-1	1	-1	-1
-1	1	-1	-1	-1

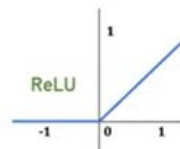
*

Loopy pattern
filter

1	1	1
1	-1	1
1	1	1



-0.11	1	-0.11
-0.55	0.11	-0.33
-0.33	0.33	-0.33
-0.22	-0.11	-0.22
-0.33	-0.33	-0.33



0	1	0
0	0.11	0
0	0.33	0
0	0	0
0	0	0

But still the issue of too much computation is not resolved...

5	1	3	4
8	2	9	2
1	3	0	1
2	2	2	0

8	9
3	2

2 by 2 filter with stride = 2

Shifted 9 at
different position

1	1	1	-1	-1
1	-1	1	-1	-1
1	1	1	-1	-1
-1	-1	1	-1	-1
-1	-1	1	-1	-1
-1	1	-1	-1	-1
1	-1	-1	-1	-1

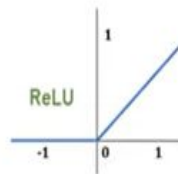
Loopy pattern
filter

*

1	1	1
1	-1	1
1	1	1

→

1	-0.11	-0.11
0.11	-0.33	0.33
0.33	-0.33	-0.33
-0.11	-0.55	-0.33
-0.55	-0.33	-0.55



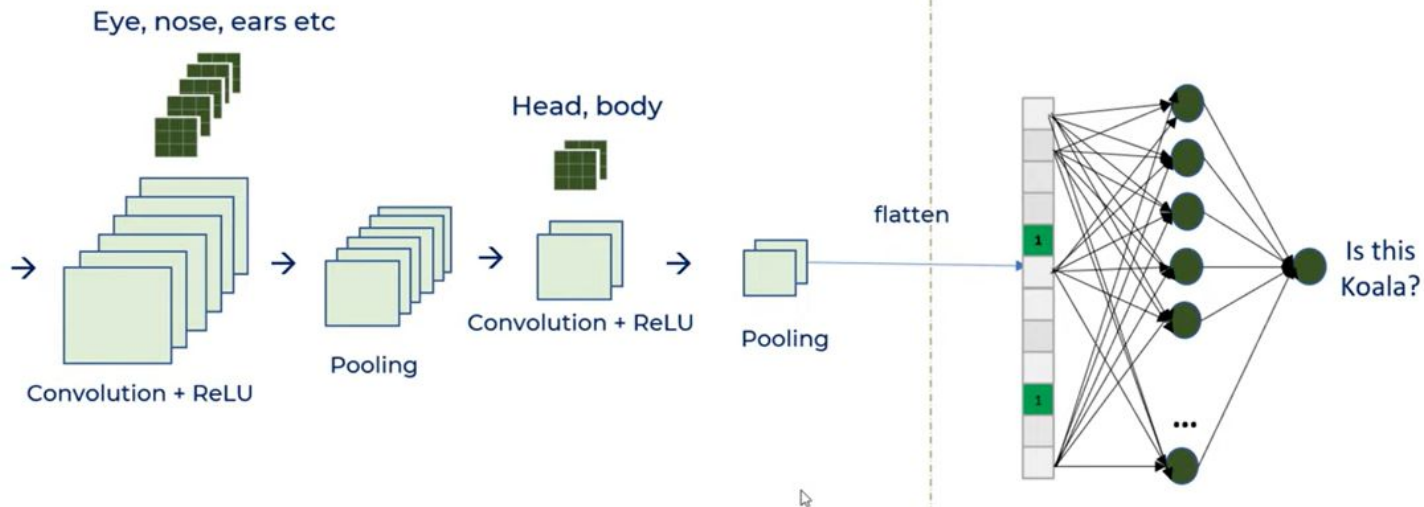
→

1	0	0
0.11	0	0.33
0.33	0	0
0	0	0
0	0	0

Max
pooling

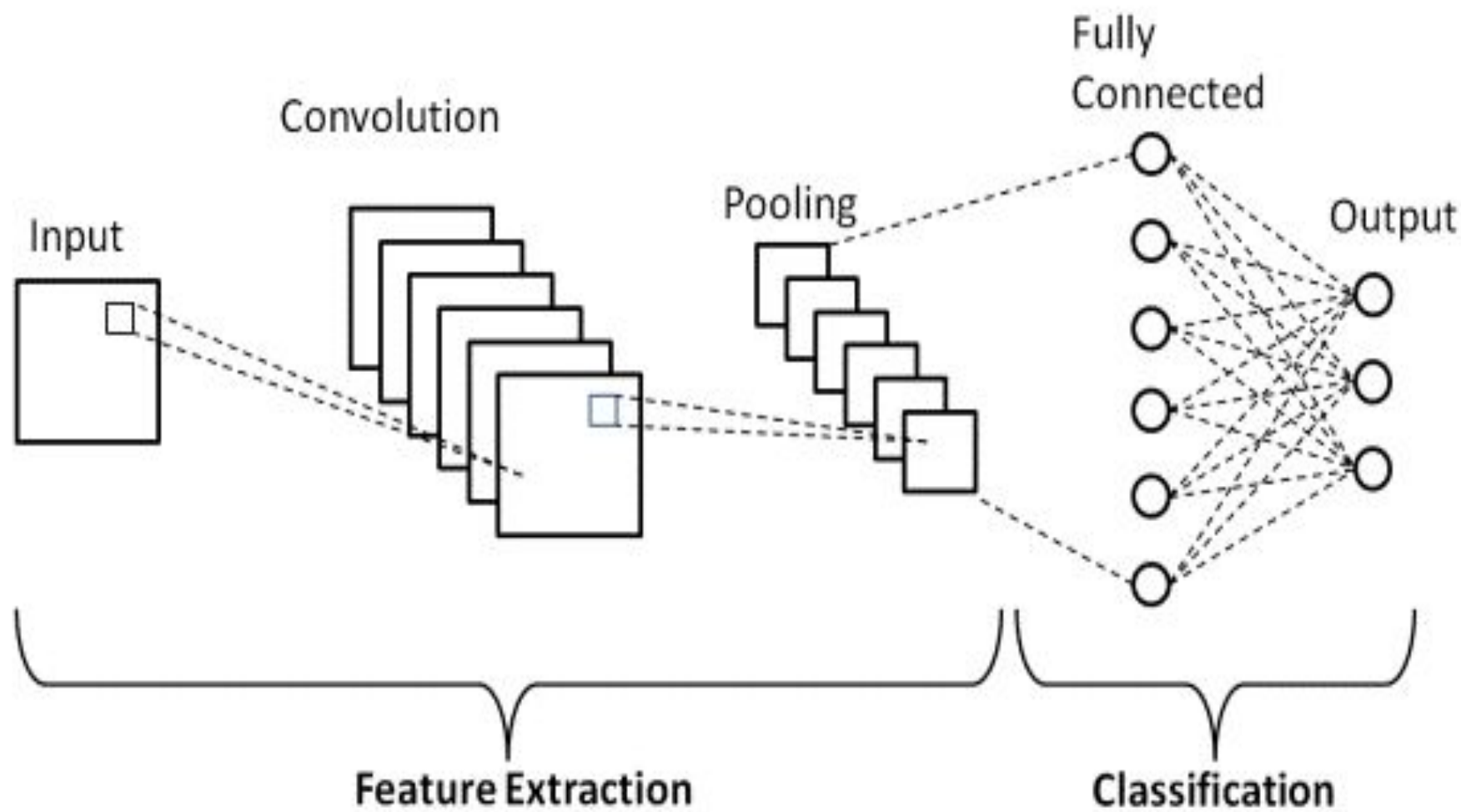
→

1	0.33
0.33	0.33
0.33	0
0	0



Feature Extraction

Classification



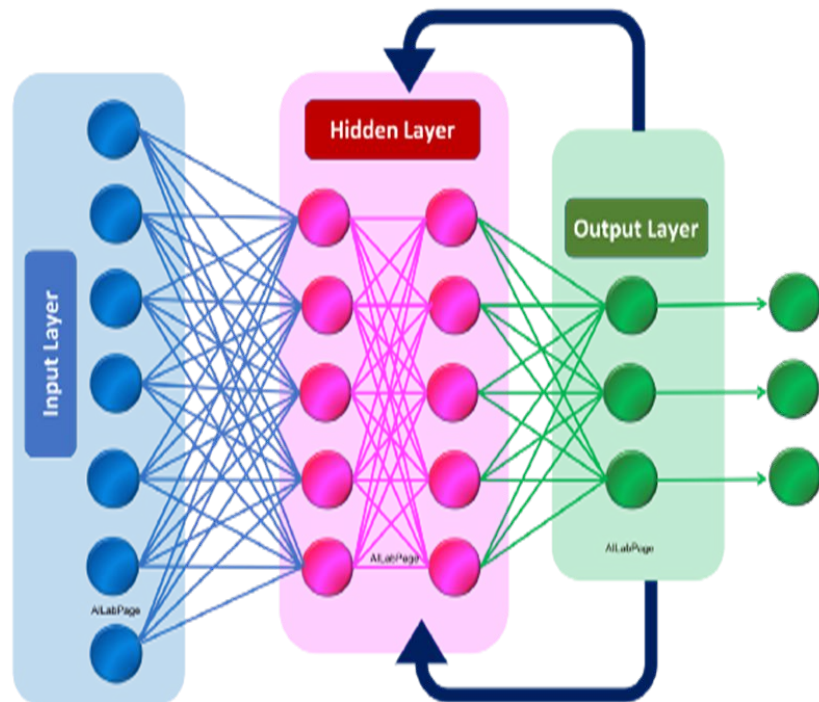
RNN – Recurrent Neural Network

A recurrent neural network (RNN) is a type of neural network that is particularly well-suited for **processing sequential data**, such as time-series data or natural language text.

Unlike feedforward neural networks, which process input data in a strictly forward direction, RNNs have a **feedback loop** that allows information to persist and be passed from one time step to the next.

This feedback loop makes RNNs particularly good at modeling sequences with long-term dependencies.

Recurrent Neural Networks

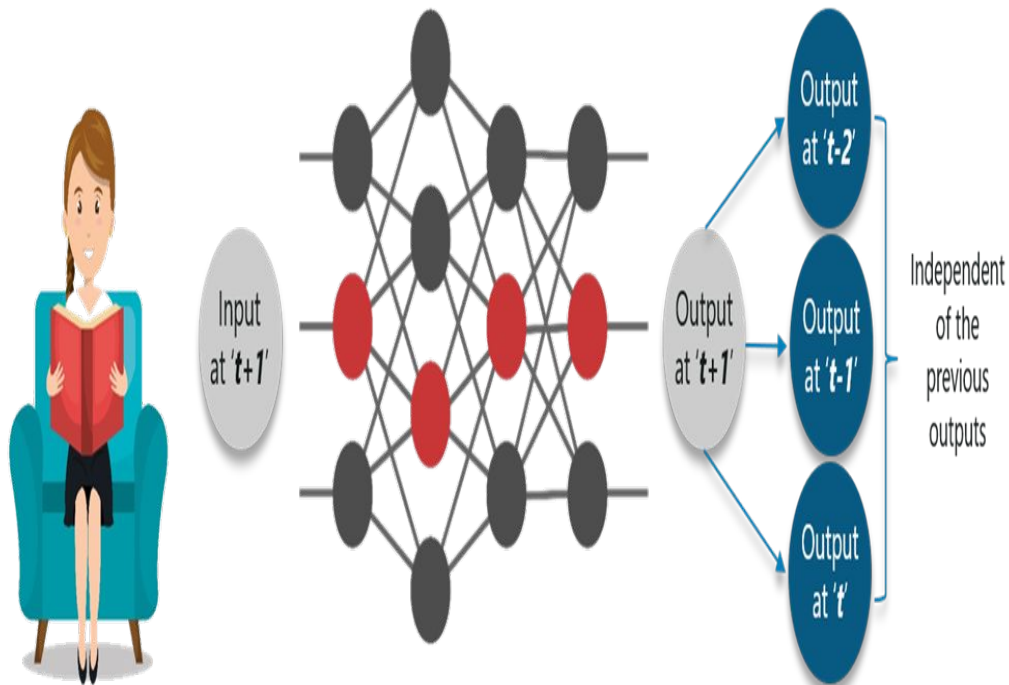


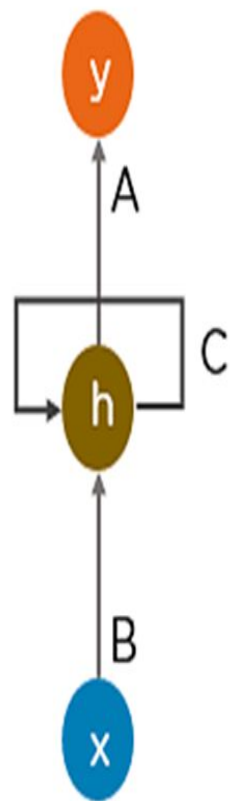
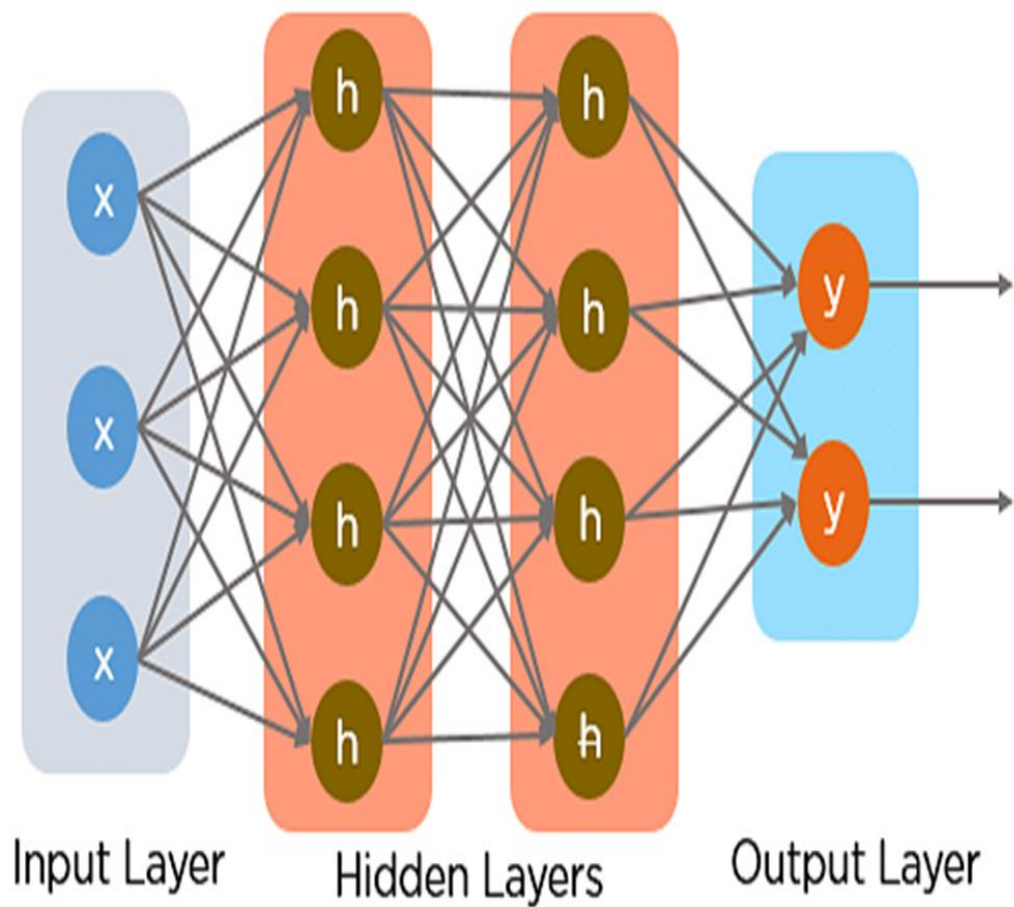
Why Not Feedforward Networks?

- Consider this **scenario** where you will require the **use** of the **previously obtained output**:

- The concept is similar to **reading a book**. With **every page** you move forward into, you need the **understanding** of the **previous pages** to make **complete** sense of the **information** ahead in most of the **cases**.

- With a **feed-forward network** the **new** output at time ' **$t+1$** ' has **no relation** with outputs at either time t , $t-1$ or $t-2$.





Recurrent Neural Network

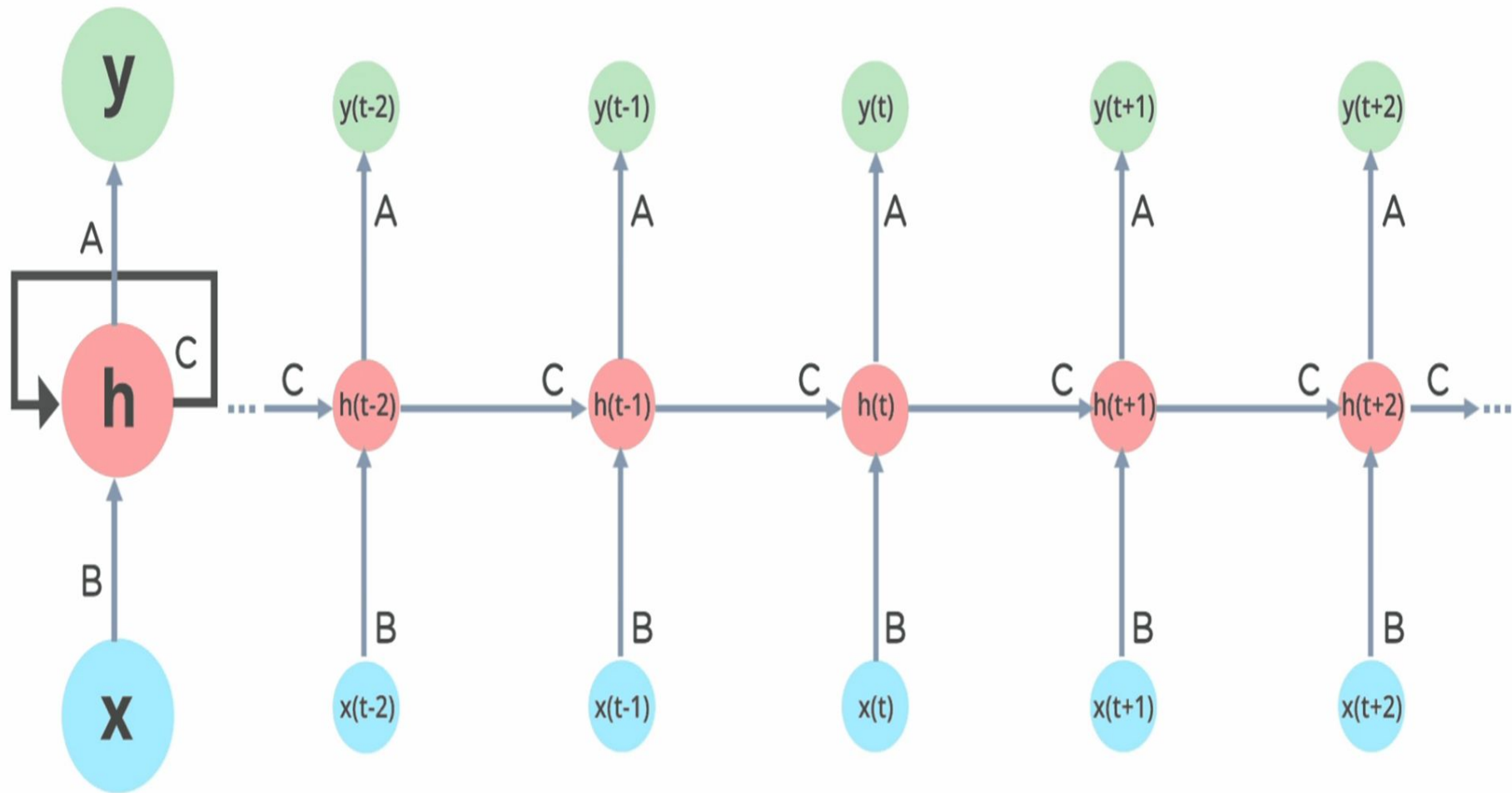


Image Captioning



"A Dog catching a ball in mid air"

Natural Language Processing



Natural Language Processing

When it rains, look for rainbows.
When it's dark, look for stars.

Positive Sentiment

Machine Translation



Here the person is speaking in English and it is getting translated into Chinese, Italian, French, German and Spanish languages

Machine Translation

Training Techniques for ANN

1. Training a network using Supervised Learning
1. Training a network using Unsupervised Learning
1. Training a network using Reinforced Learning

ANNs to solve some real-life problems

1. Image Recognition: ANNs have been used extensively for image recognition tasks, such as identifying objects in images, detecting faces, and recognizing handwriting. One notable application is Google's image recognition software, which uses a deep neural network to identify objects in images.
2. Fraud Detection: ANNs can be trained to detect fraudulent transactions in credit card data. For example, a bank might use an ANN to analyze a customer's transaction history and flag any unusual or suspicious activity.
3. Speech Recognition: ANNs are used in many speech recognition systems, such as voice assistants like Siri and Alexa. These systems use neural networks to analyze audio input and identify the words being spoken.

ANNs to solve some real-life problems

1. **Stock Market Prediction:** ANNs can be used to predict stock prices based on historical data. For example, a hedge fund might use an ANN to analyze past stock market trends and make predictions about future market movements.
2. **Medical Diagnosis:** ANNs can be trained to diagnose medical conditions based on patient data, such as symptoms, medical history, and test results. For example, an ANN might be used to diagnose cancer based on the results of a patient's biopsy.
3. **Natural Language Processing:** ANNs can be used for natural language processing tasks, such as sentiment analysis, language translation, and chatbot development. For example, Facebook's AI-powered chatbot uses a neural network to understand and respond to user messages.

Simulation of biological neurons to problem solving

By simulating the way that neurons in the brain process information, ANNs can learn to recognize patterns, make decisions, and perform complex tasks.

One example of a problem that can be solved using ANNs that simulate biological neurons is image classification.

Another example of a problem that can be solved using ANNs is speech recognition.

ANNs can be used to control the movements of a robot's limbs, by simulating the way that neurons in the brain control muscle movements.

The complexity of ANN is dependent upon _____?

Number of Neurons

Number of Nodes

Number of Anodes

Number of Layers

What is the purpose of ART?

- a) take care of approximation in a network
- b) take care of update of weights
- c) take care of pattern storage
- d) none of the mentioned

What type of inputs does ART – 1 receives?

- a) bipolar
- b) binary
- c) both bipolar and binary
- d) none of the mentioned

Can data be stored directly in associative memory?

a) yes

b) no

which of the following is false?

- a) neural networks are artificial copy of the human brain
- b) neural networks have high computational rates than conventional computers
- c) neural networks learn by examples
- d) none of the mentioned

What is the objective of associative memories?

- a) to store patterns
- b) to recall patterns
- c) to store association between patterns
- d) none of the mentioned

Image recognition is possible by using _____?

Recurrent Neural Networks

Feed-forward Neural Networks

Convolutional Neural Networks

Deconvolutional Neural Networks