

(DEEP LEARNING)

IV B.TECH I SEM

(PVP-20)



DEEP
LEARNING

Topics :

- Introduction.
- History of Deep Learning
- Differences b/w Machine Learning and Deep Learning
- Learning Resources for this Course
- Why we use Deep Learning?
- How Deep Learning Works?
- Example of Deep Learning.

Topics :

- Types of Deep Learning.
- Applications of Deep Learning
- Deep Learning Frameworks
- Learning Beyond Curriculum

INTRODUCTION TO DEEP LEARNING

- Deep learning is a subset of Machine Learning that uses multi-layered neural networks, called **deep neural networks**.
- Deep learning is a **type of artificial intelligence** that can “**duplicate**” the human brain's functioning.
- The term “**deep**” usually refers to the **number of hidden layers in the neural network**.
- Models are trained by using a **large set of labeled data** and neural network architectures that **contain many layers**.

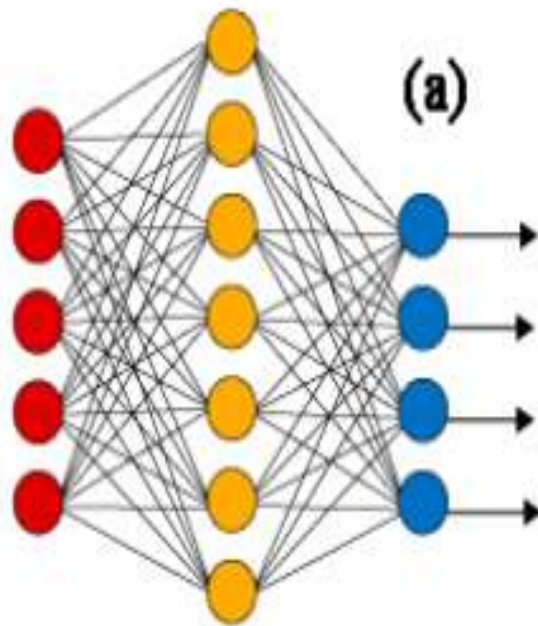


- Deep learning models can recognize complex patterns in pictures, text, sounds, and other data to produce accurate insights and prediction. It improves the ability to classify, recognize, detect and describe using data.
- Neural networks are made up of layers of interconnected nodes, and each node is responsible for learning a specific feature of the data.

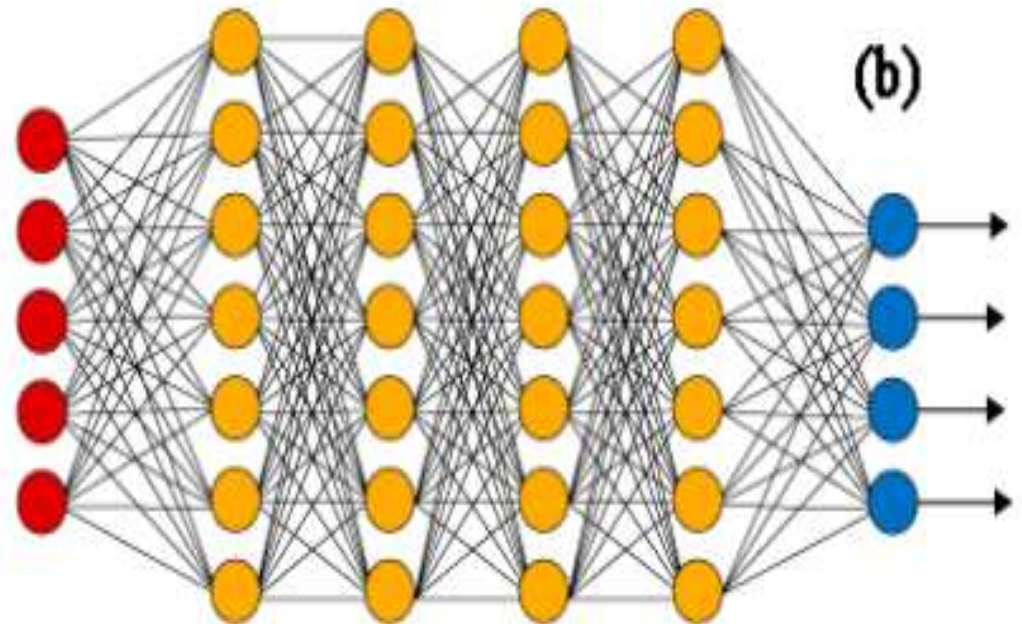


- For Example, we have to find out the sentences- the **first layer of nodes** might learn to **identify the letters**, the **second layer** might learn to **identify the words**, and the **third layer** might learn to **identify sentences** etc. Like this we **may increase the hidden layers** in the Artificial Neural Network.





(a)



(b)

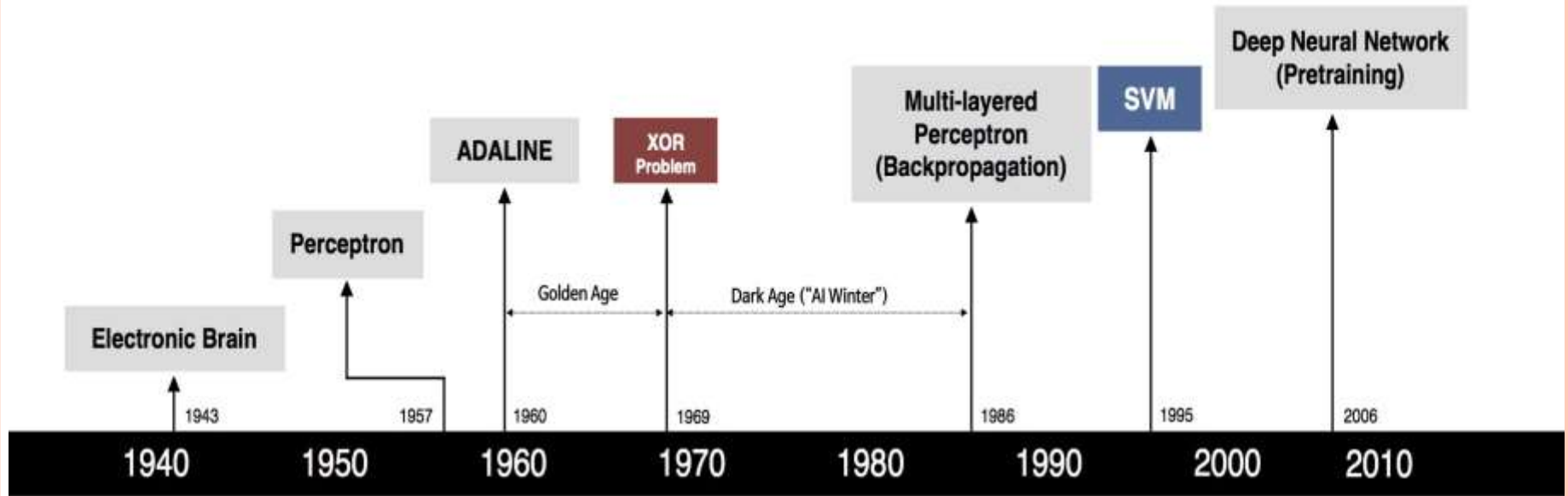
● Input Layer

● Hidden Layer

● Output Layer

(a) Simple neural network architecture; (b) Simple architecture of deep neural network (DNN)[49]

HISTORY OF DEEP LEARNING



S. McCulloch - W. Pitts



F. Rosenblatt



B. Widrow - M. Hoff



M. Minsky - S. Papert



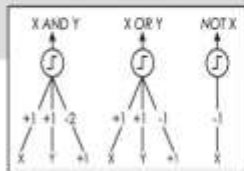
D. Rumelhart - G. Hinton - R. Williams



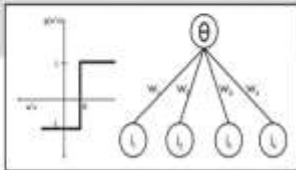
V. Vapnik – C. Cortes



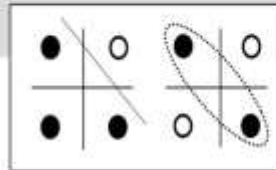
G. Hinton - S. Ruslan



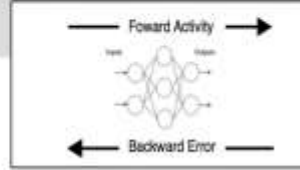
- Adjustable Weights
- Weights are not Learned



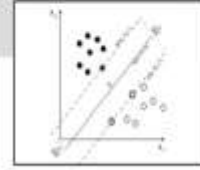
- Learnable Weights and Threshold



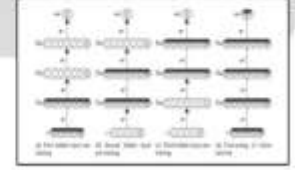
- XOR Problem



- Solution to nonlinearly separable problems
- Big computation, local optima and overfitting

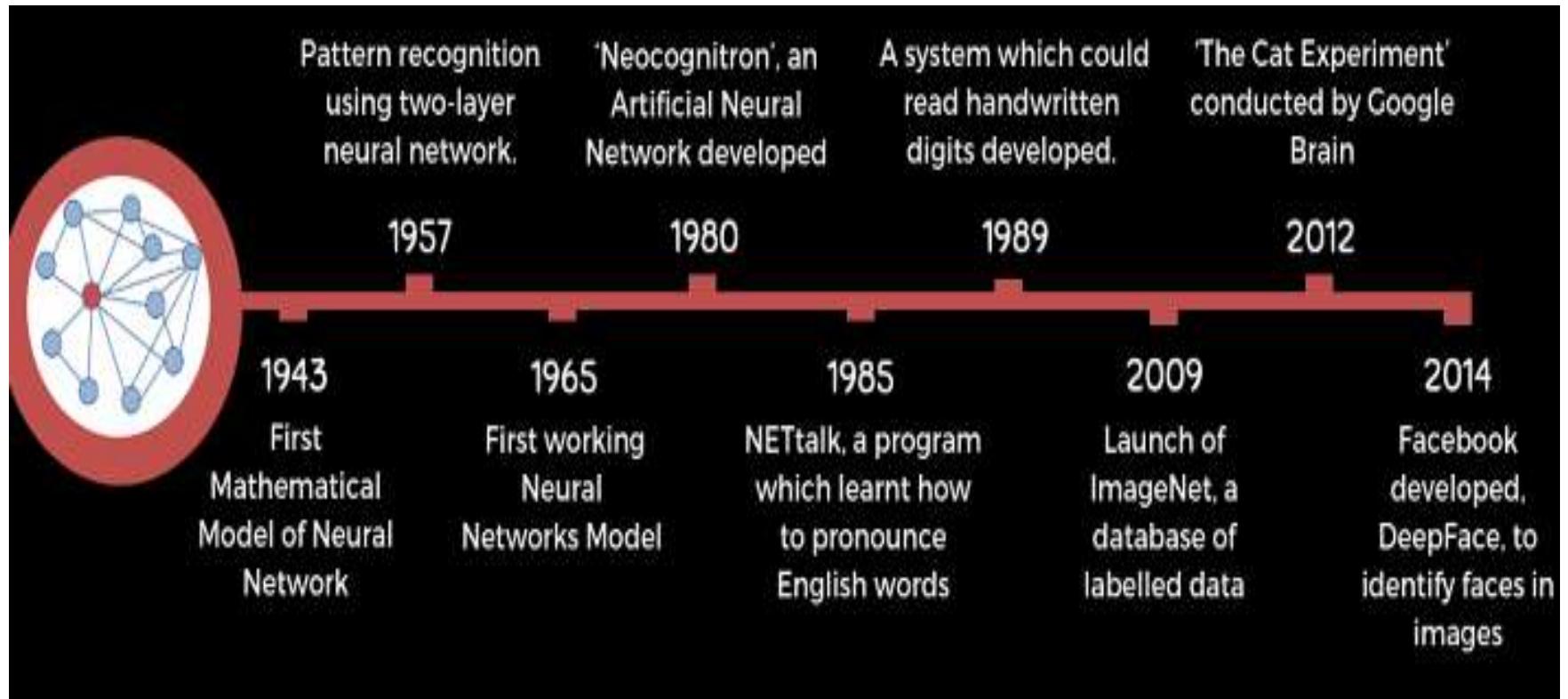


- Limitations of learning prior knowledge
- Kernel function: Human Intervention



- Hierarchical feature Learning

HISTORY OF DEEP LEARNING



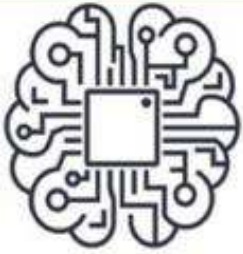
- **1957** - **Frank Rosenblatt** submitted a paper titled 'The Perceptron: A Perceiving and Recognizing Automaton', which consisted of an algorithm or a method for pattern recognition using a **two-layer neural network**.
- **1965** - **Alexey Ivakhnenko** and **V.G. Lapa** developed the first working neural network and **Alexey Ivakhnenko** created an **8-layer deep neural network** in **1971** which was demonstrated in the **computer identification system, Alpha**. This was the **actual introduction to deep learning**.



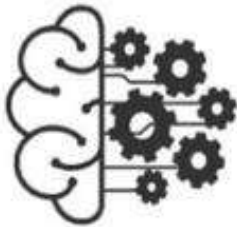
- **1980** - **Kunihiko Fukushima** developed the '**Neocognitron**', an **Artificial deep neural network** with **multiple and convolutional layers** to recognize visual patterns.
- **1985** - **Terry Sejnowski** created **NETtalk**, a **program which learnt how to pronounce English words**.
- **1989** - **Yann LeCun**, using **convolution deep neural network**, developed a system which could **read handwritten digits**.
- **Mid-2000s**: The term **“deep learning”** begins to gain popularity after a paper by **Geoffrey Hinton and Ruslan Salakhutdinov** showed **how a many-layered neural network could be pre-trained one layer at a time**.

- **2009** - As deep learning models require a **tremendous amount of labelled data** to train themselves in supervised learning, **Fei-Fei Li** launched **ImageNet**, which is a **large database of labelled images**.
- **2012** - The results of '**The Cat Experiment**' conducted by **Google Brain** were released. This experiment was based on **unsupervised learning** in which the **deep neural network** worked with **unlabelled data to recognize patterns and features in the images of cats**. However, it could only recognize 15% of images correctly.
- **2014** - Facebook puts **deep learning technology** – called **DeepFace** – into operations to **automatically tag and identify Facebook users in photographs**.

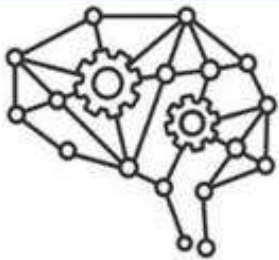
DIFFERENTIATE B/W AI, ML, DL



Artificial Intelligence: any technique that allows computer systems to mimic the human intelligence and behaviour



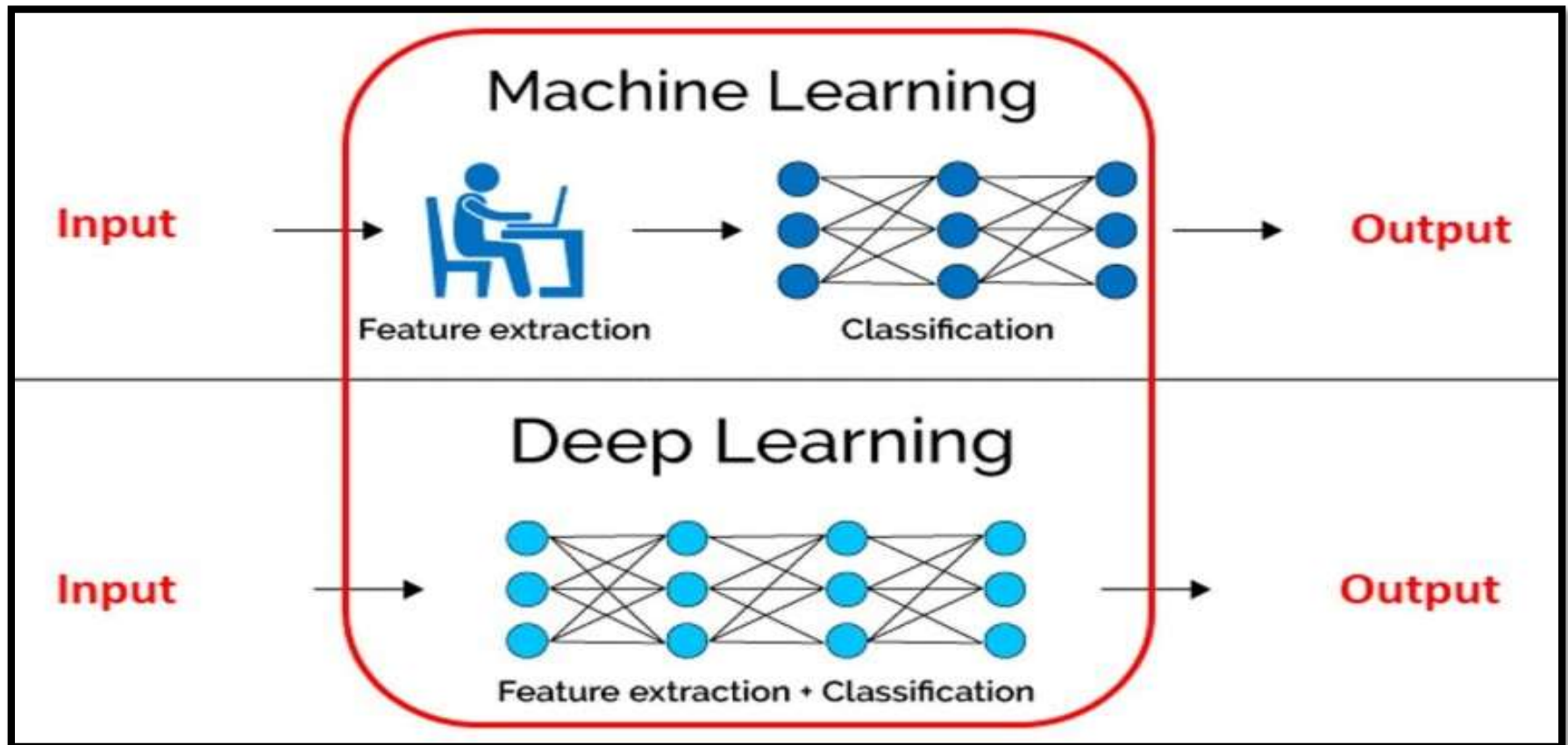
Machine Learning: subset of AI, that uses statistical techniques to enable machines to learn from data and improve with experience

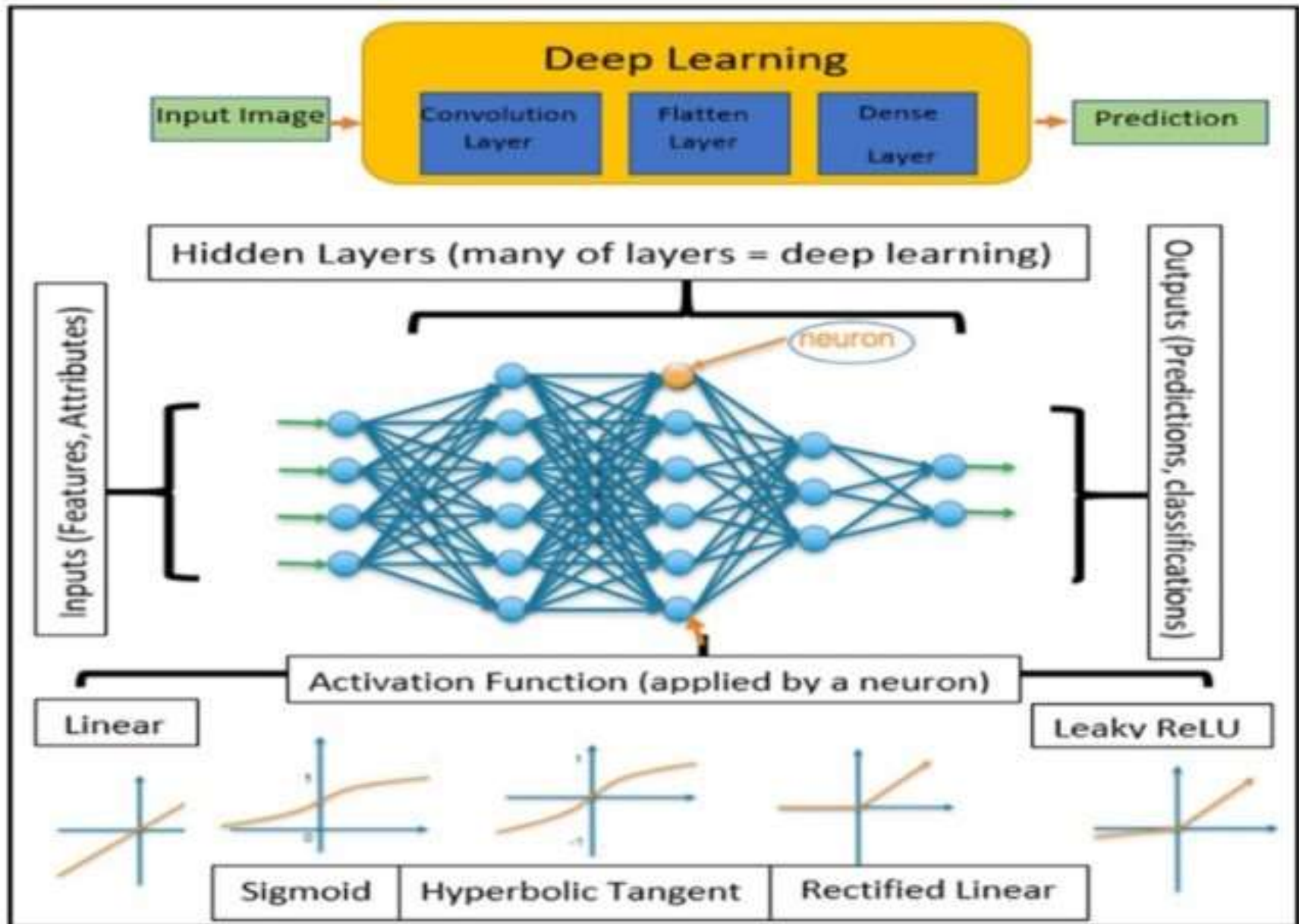


Deep Learning: subset of ML, in which multilayered neural networks learn from vast amounts of data

- The **Main Difference between ML and DL** is

Deep learning **eliminates some of data pre-processing** that is typically involved with machine learning.






CONTD..

S. No	Machine Learning	Deep Learning
1	ML is invented in the Year 1950	ML is invented in the Year 1970
2	Apply statistical algorithms to learn the hidden patterns and relationships in the dataset.	Uses artificial neural network architecture to learn the hidden patterns and relationships in the dataset.
3	Can work on the smaller amount of dataset	Requires the larger volume of dataset compared to machine learning
4	Better for the low-label task.	Better for complex task like image processing, natural language processing, etc.
5	Takes less time to train the model.	Takes more time to train the model.
6	Automatically Extracted Features from data using Algorithms	Need not having separate feature extraction. Here given input to Networks
7	Machines can take decisions on their own, based on Past Data	Machines can take decisions with the help of Artificial Neural Networks.
8	Less complex and easy to interpret the result.	More complex, it works like the black box interpretations of the result are not easy.
9	It can work on the CPU or requires less computing power as compared to deep learning.	It requires a high-performance computer with GPU.
10	Data is depends on Labeled and Unlabeled Data for Training and Testing	Requires Large amount of Labeled Data to train complex models

PREREQUISITES

- To effectively learn **Deep Learning**, which is a **subset of machine learning** based on **Neural Networks** with **multiple layers**, it is beneficial to have a understanding of the following **fundamental concepts and topics**:
 - **1. Mathematics**:
 - **Linear Algebra**: Vectors, matrices, matrix operations, Eigen Values, Eigen Vectors, etc.
 - **Calculus**: Derivatives, chain rule, gradients, partial derivatives, etc.
 - **Probability and Statistics**: Probability distributions, Bayes' theorem, mean, variance, etc.
- 

○ 2. Programming Skills:

- **Python:** Deep learning frameworks such as **TensorFlow** and **PyTorch** are commonly used with Python.
- **NumPy:** For **numerical computing** in Python.
- **Pandas:** For **data manipulation and analysis**.
- **Jupyter Notebooks:** For **interactive coding** and **experimentation**.



○ 3. Machine Learning Concepts:

- **Supervised Learning:** Understanding of concepts like regression and classification.
- **Unsupervised Learning:** Clustering, Dimensionality reduction, etc.
- **Model Evaluation:** Cross-validation, overfitting, underfitting, etc.

○ 4. Data Preprocessing:

- Cleaning, normalization, feature scaling, handling missing data etc.



○ 5. Basic Understanding of Neural Networks:

- **Perceptron:** The basic building block of neural networks.
- **Activation Functions:** Sigmoid, ReLU, tanh, etc.
- **Back Propagation:** The algorithm used to train neural networks.

○ 6. Deep Learning Frameworks:

- Familiarity with popular deep learning libraries such as **TensorFlow or Keras** is essential.



Course Outcomes

Upon successful completion of the course, the student will be able to

CO1	Understand the fundamental concepts of Deep learning.
CO2	Apply concepts of deep networks to analyze various architectures.
CO3	Apply deep learning models to build applications in various domains.
CO4	Analyze the given problem and apply suitable deep learning algorithm.



SYLLABUS

SYLLABUS		
UNIT NO.	CONTENTS	MAPPED CO
I	Fundamentals of Deep Networks – Defining Deep Learning, What Is Deep Learning? Common Architectural Principles of Deep Networks: Parameters, Layers, Activation Functions, Loss Functions, Hyper parameters.	CO1
II	Building Blocks of Deep Networks – Restricted Boltzmann Machine, Autoencoders, Variational Autoencoders. Major Architectures of Deep Networks: Unsupervised pretrained networks, Deep Belief Networks, Generative Adversarial Networks.	CO1, CO2, CO4
III	Convolutional Neural Networks (CNNs) – The Convolution Operation, Motivation, Pooling, Variants of the Basic Convolution Function, Structured Outputs, Data Types, Efficient Convolution Algorithms, Random or Unsupervised Features, The Neuro-scientific Basis for Convolutional Networks, Applications.	CO1, CO3, CO4
IV	Sequence Modeling – Recurrent and Recursive Nets – Unfolding Computational Graphs, Recurrent Neural Networks, Encoder-Decoder Sequence-to-Sequence Architectures, Deep Recurrent Networks, Recursive Neural Networks, The Long Short-Term Memory and Other Gated RNNs, Applications.	CO1, CO3, CO4
V	Deep Learning applications – Computer Vision, Speech Recognition, Natural Language Processing, Other Applications.	CO1, CO3, CO4

LEARNING RESOURCES :

❖ Text Books:

1. Deep learning: A practitioner's approach, Josh Patterson and Adam Gibson, –O'Reilly Media, First Edition, 2017.
2. Deep Learning , Ian Goodfellow, Yoshua Bengio, Aaron Courville, —, MIT Press, 2016.
3. Deep learning, Amit Kumar Das, Saptarsi Goswami, Pabitra Mitra, Amlan Chakrabarti, Pearson Education, First Edition, 2021.

LEARNING RESOURCES :

❖ Reference Books:

1. Fundamentals of Deep Learning, Designing next-generation machine intelligence algorithms, Nikhil Buduma, O_Reilly, Shroff Publishers, 2019.
2. Deep learning Cook Book, Practical recipes to get started Quickly, Douwe Osinga, O_Reilly, Shroff Publishers, 2019.
3. Deep learning Illustrated A Visual Interactive Guide to Artificial Intelligence, Jon Krohn, Grant Beyleveld, Aglae Bassens, Pearson Education, First Edition, 2020,

LEARNING RESOURCES :

e- Resources and other Digital Material:

1. <https://www.deeplearningbook.org/S>
2. https://onlinecourses.nptel.ac.in/noc20_cs62/preview
3. <https://www.udemy.com/share/101X6W/>

(or)

- <https://www.udemy.com/course/deep-learningadvanced-nlp/>
4. https://www.youtube.com/watch?v=5tvmMX8r_OM&list=PLtBw6njQRUrwp5_7C0oIVt26ZgjG9NI

REASONS TO USE DEEP LEARNING

1. Analyzing Unstructured Data: Deep learning algorithms can be trained to look at text data by analyzing social media posts, news, and surveys to provide valuable business and customer insights.
2. Feature engineering: A deep learning algorithm can save time because it does not require humans to extract features manually from raw data.
3. Efficiency: When a deep learning Algorithm is properly trained, it can perform thousands of tasks over and over again, faster than humans.

4. Training: The **Neural Networks** used in deep learning have the ability to be applied to **many different data types and applications**. Additionally, a deep learning model can **adapt by retraining it with new data**.

DL is usually a **more complex and high-performance GPU** to **analyze all images**.

5. Handling large and complex data: **Deep Learning algorithms** can deal **with large and complex datasets** that would be challenging for **traditional Machine Learning algorithms** to handle. This makes it **useful for finding insights from big data**, such as **posts on social media, webpages, videos, audio files, and sensor data**.

6. Handling Non-Linear Relationships: This allows it to model complex phenomena and capture higher-level abstractions. Some of the examples of handling non-linear relationships by Deep Learning algorithms are:

- a) **Physical Systems:** A Deep Potential Molecular Dynamics (DPMD) achieved an accuracy of 98.9% on predicting the potential energy surface of water molecules, which is a highly non-linear function of atomic positions.
- b) **Biological Systems:** A Deep Learning model called DeepChrome achieved an accuracy of 89.9% on predicting the chromatin state of genes.

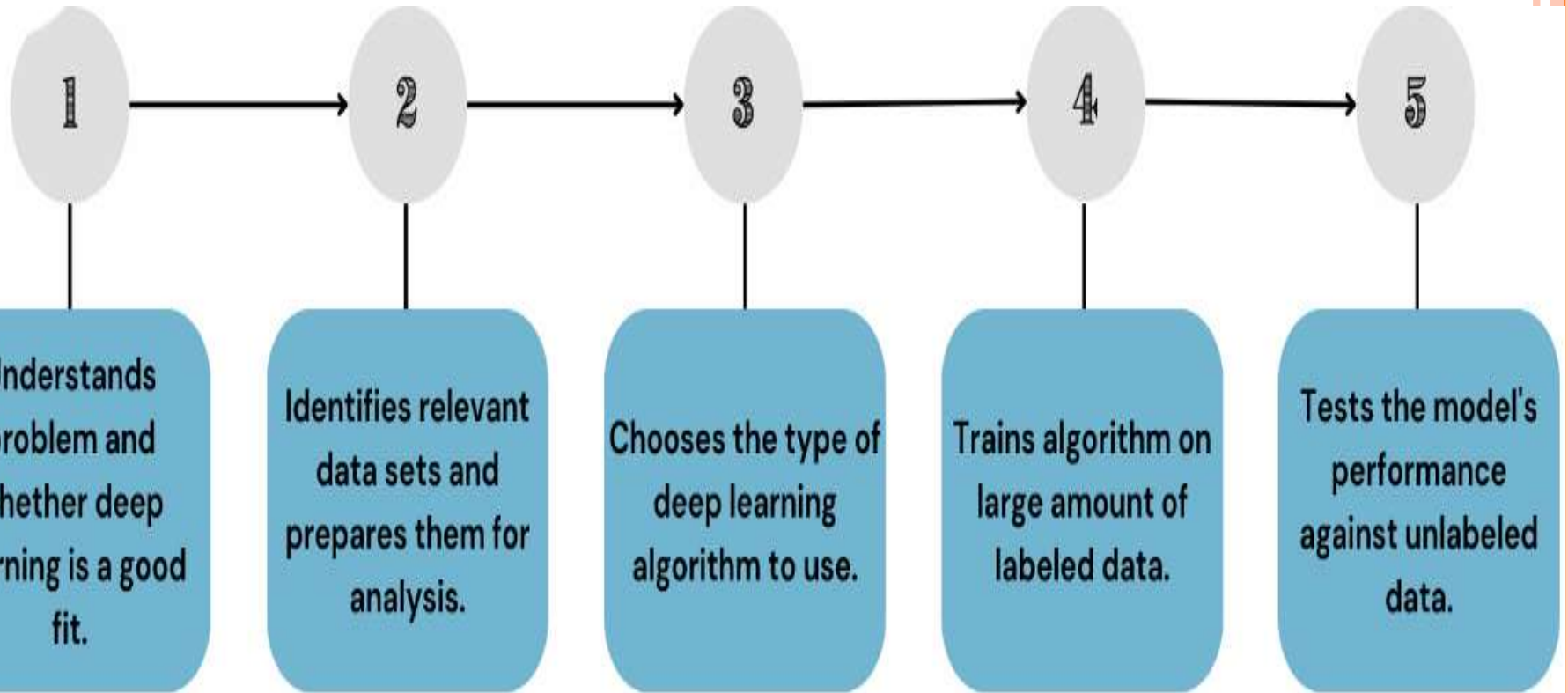
How DEEP LEARNING WORKS

- Deep learning work on Neural Network Architectures
- Deep learning is implemented by the help of **deep networks**, which are nothing but **neural networks** with **multiple hidden layers**.
- It **came in the 1980s**, but due to a **lack of labeled data and computing power**, it was not popular at that time.
- Today, we have large amounts of labeled data. For example, a **driverless car** requires millions of images and thousands of hours of video to train with **high accuracy**.

How DEEP LEARNING WORKS

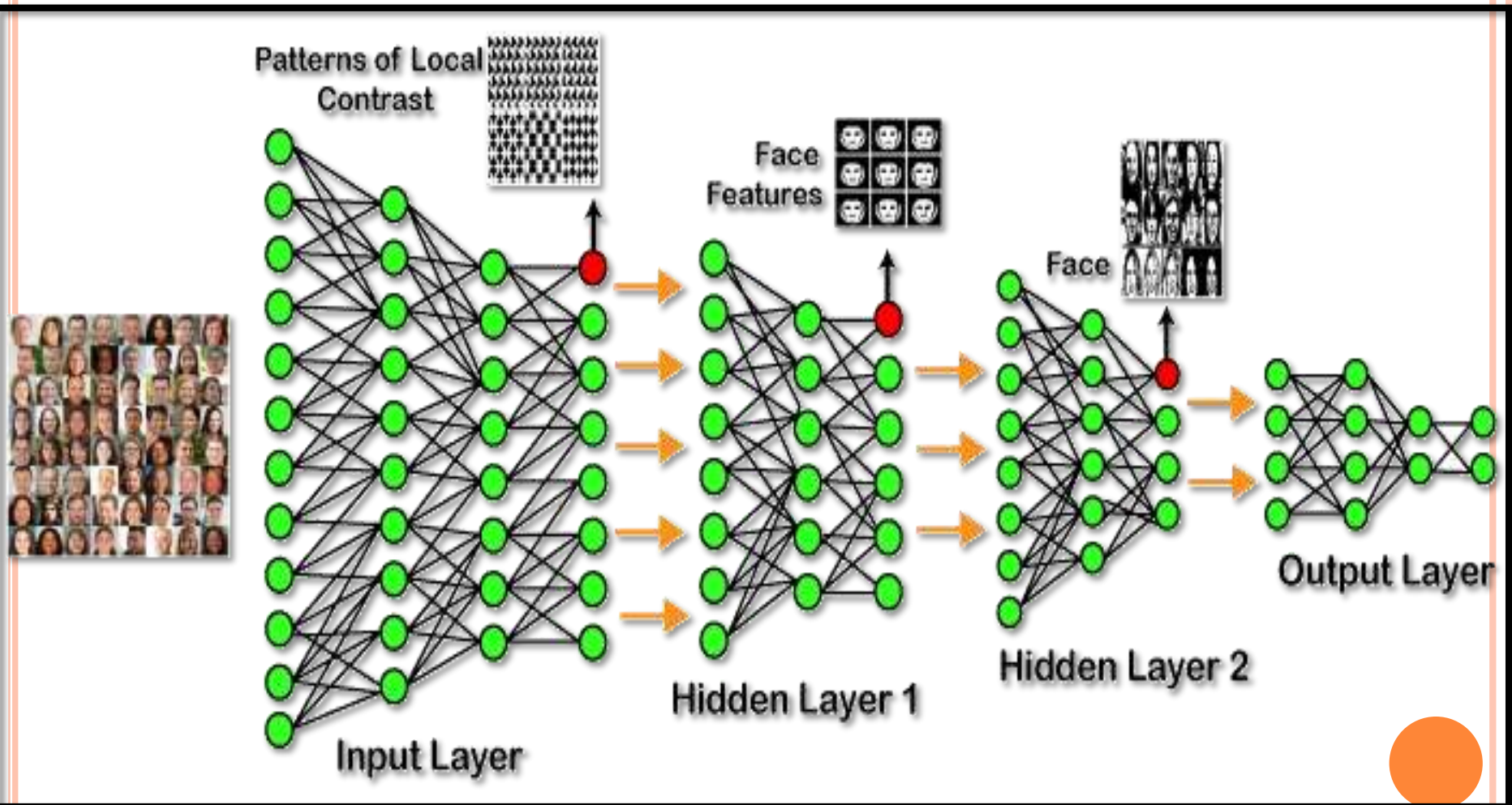
- The **number of hidden layers** in the **neural network** usually refers to **“deep”**. Hidden layers in **deep neural networks** can have **as many as 150**.
- These **models** are trained by using **large sets of labeled data** & **neural networks** learn features directly from the data.
- For **computing power**, we have **high-performance GPUs**, **cloud computing** that is **efficient** for deep learning.

How DEEP LEARNING WORKS

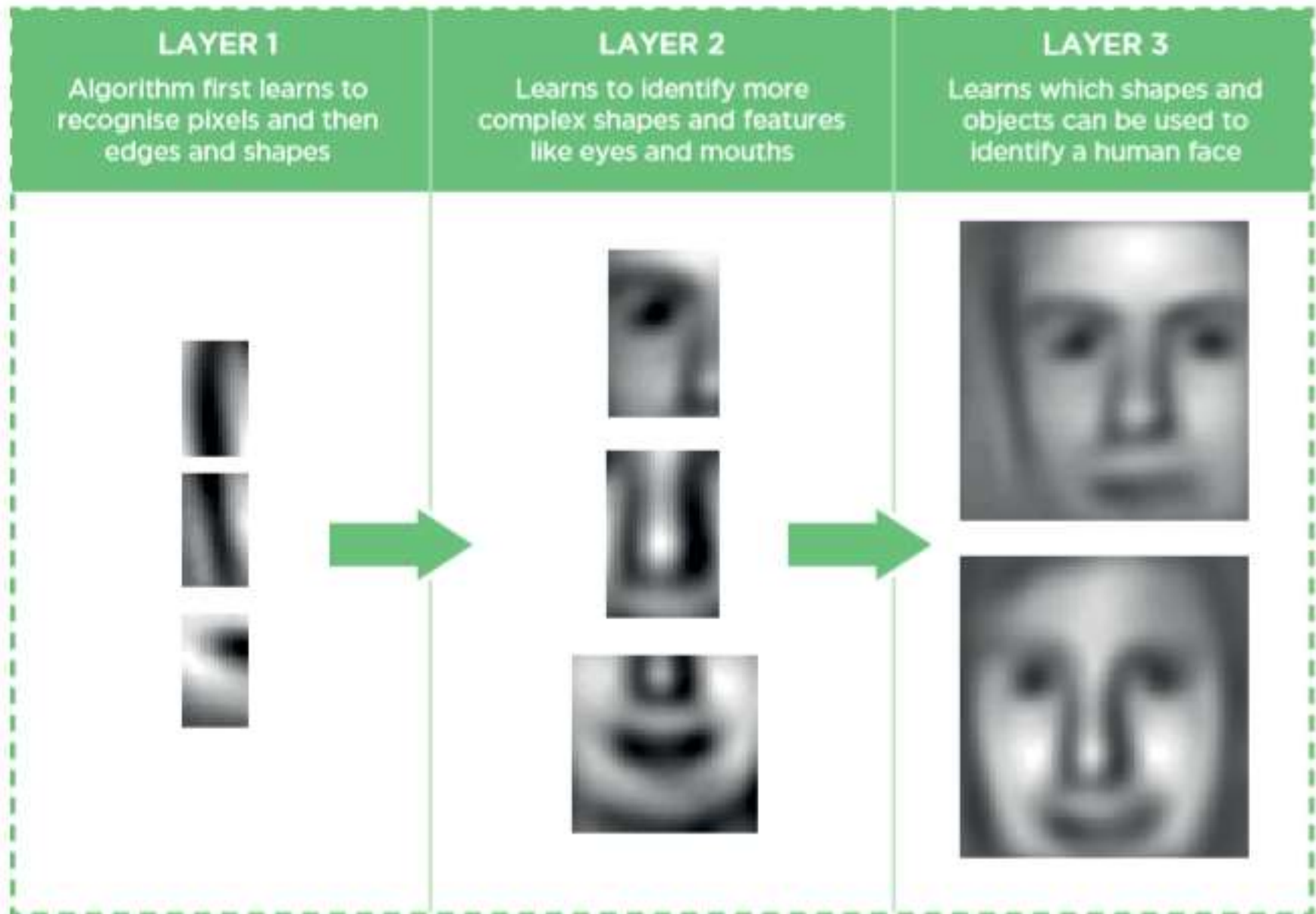


EXAMPLE OF DEEP LEARNING

➤ Image Recognition using Deep Learning:

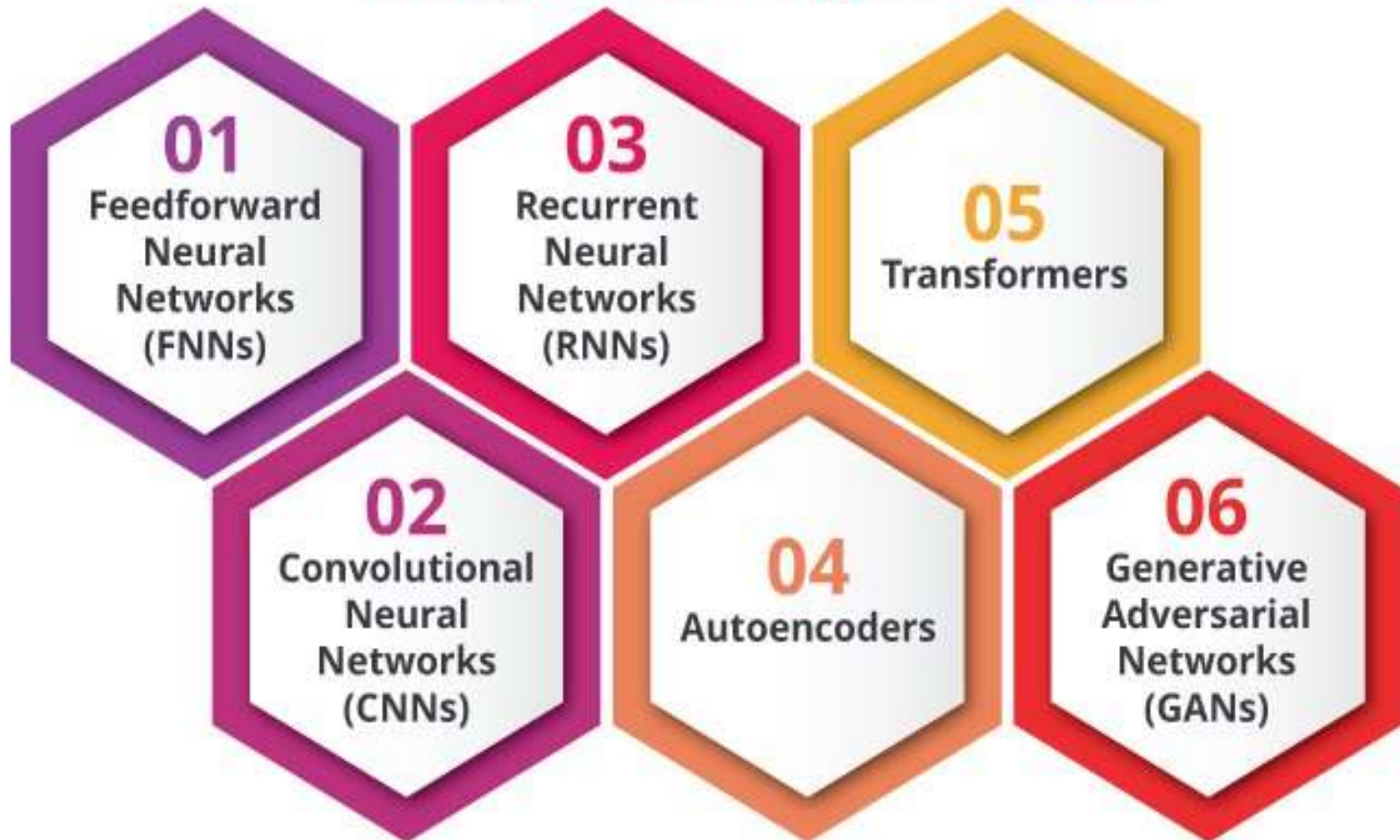


- 1 Here, we are passing the high dimensional data to the input layer. To match the dimensionality of the input data, the input layer will contain multiple sub-layers of perceptrons so that it can consume the entire input.
- 2 The output received from the input layer will contain patterns and will only be able to identify the edges of the images based on the contrast levels.
- 3 This output will be fed to the Hidden layer 1 where it will be able to identify various face features like eyes, nose, ears etc.
- 4 Now, this will be fed to the hidden layer 2 where it will be able to form the entire faces. Then, the output of layer 2 is sent to the output layer.
- 5 Finally, the output layer performs classification based on the result obtained from the previous and predicts the name.



TYPES OF DEEP LEARNING

Types Of Deep Learning models



○ 1. Feed Forward Neural Networks(FNNs):

- FNNs, also known as Multilayer Perceptrons (MLPs), are the **simplest type of artificial neural network**.
- In an FNN, **information moves in only one Direction**. It can be **moved from Input Layer to Output Layer**. There are **no back-loops in the feed-forward network**.
- To minimize the prediction error, the **back propagation algorithm can be used to update the weight values**.
- FNNs are widely used for **simple classification and regression problems**.

○ Applications:

- Data Compression
- Pattern Recognition
- Computer Vision
- Sonar Target Recognition
- Speech Recognition
- Handwritten Characters Recognition

○ 2. Convolutional Neural Networks(CNNs):

- CNNs are a type of **deep learning model** that are especially effective for **processing grid-like data** such as images, making them extremely useful for image recognition, image classification, clustering of images and object recognition etc.
- An advanced example of CNN is its use in **self-driving cars**, specifically in the system **responsible for detecting objects around the vehicle**.

The **real-time application** where the **CNN doesn't just classify static images** but **continuously processes video frames** to **identify and classify** objects such as other vehicles, pedestrians, traffic signs, and lanes.

Applications:

- Identify Faces, Street Signs, Tumors.
- Image Recognition.
- Video Analysis.
- NLP.
- Anomaly Detection.
- Drug Discovery.
- Checkers Game.
- Time Series Forecasting.

○ 3. Recurrent Neural Networks(RNNs):

Recurrent neural networks get feedback from output neurons. RNNs are designed to recognize patterns in sequences of data, such as text, genomes, handwriting, or the spoken word, and are therefore very effective for natural language processing and speech recognition tasks.

Before Transformers came to age, a special variant of RNN Long Short-Term Memory Networks (LSTM) was prevalent in machine translation and text generation.

- The Recurrent neural network mainly accesses the preceding info of existing iterations.
- For example, to guess the succeeding word in any sentence, one must have knowledge about the words that were previously used. It not only processes the inputs but also shares the length as well as weights crossways time.

Applications:

- Machine Translation
- Robot Control
- Time Series Prediction
- Speech Recognition
- Speech Synthesis
- Time Series Anomaly Detection
- Rhythm Learning
- Music Composition

○ 4. Auto Encoders:

Autoencoders are a specific type of neural network architecture used for learning efficient representations of input data in an unsupervised manner.

They are composed of two parts: **An encoder**, which transforms the input data into a lower-dimensional code, and a **Decoder**, which attempts to reconstruct the original input data from this code.

Auto encoders are often employed for **anomaly detection in various types of data**, from **time-series sensor data to complex images**

Applications:

- Classification.
- Clustering.
- Feature Compression.

○ 5. Transformers:

- Transformers, specifically the “self-attention” mechanism within them, have been a **game-changer in the field of deep learning**. They are especially powerful in handling sequence data for tasks such as language understanding, translation, and text summarization.
- The best-known implementation of a transformer is **OpenAI’s GPT-3 model**, which can generate **human-like text given some input**. It can be used for tasks like **writing essays, answering questions, creating poetry, and even writing software code**.

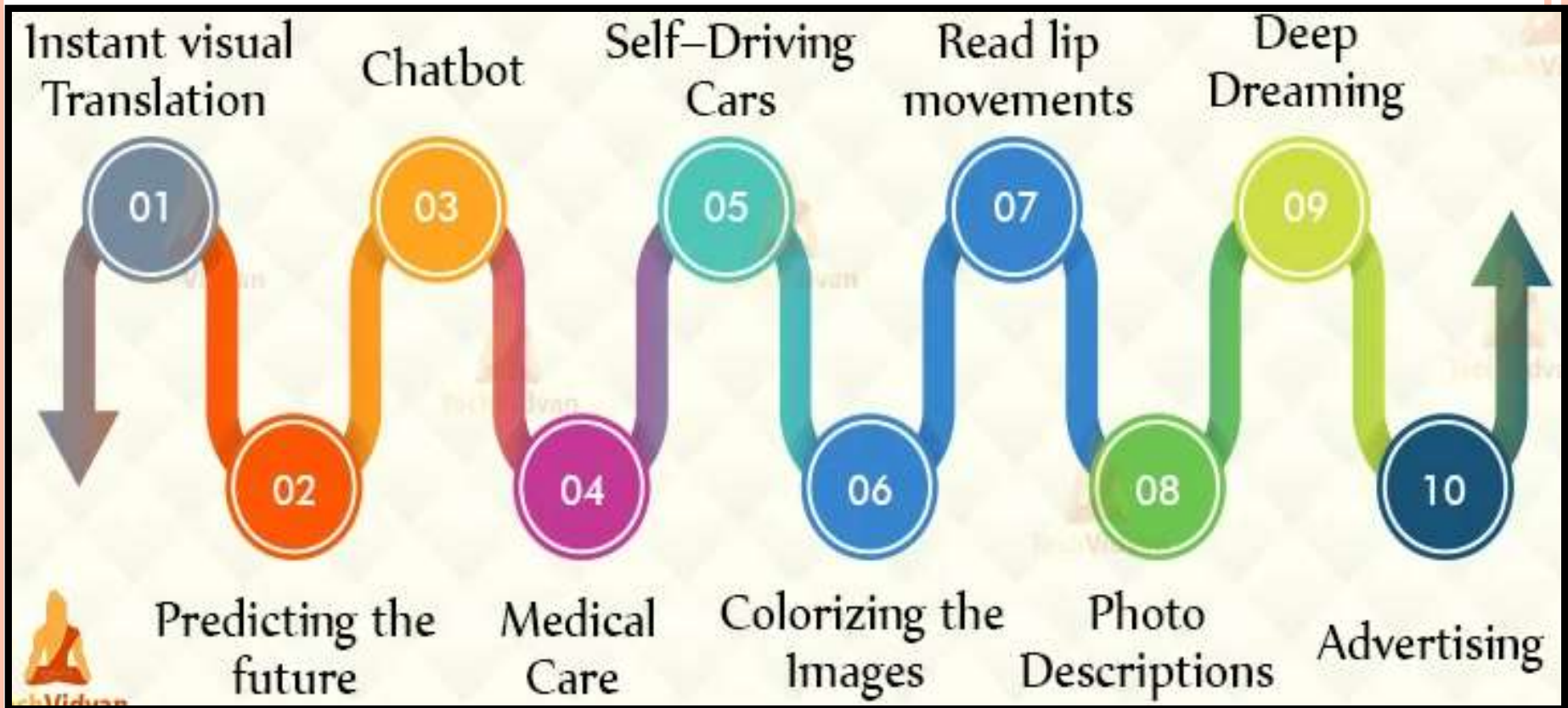
○ 6. Generative Adversarial Networks (GANs):

_____ GANs are a type of **neural network** that is used for **generating new data samples** that are **similar to a training dataset**. They consist of **two networks: a generator and a discriminator**. The **generator** tries to **generate new data samples**, while the **discriminator** tries to **distinguish between the generated samples and the training data**.

Applications:

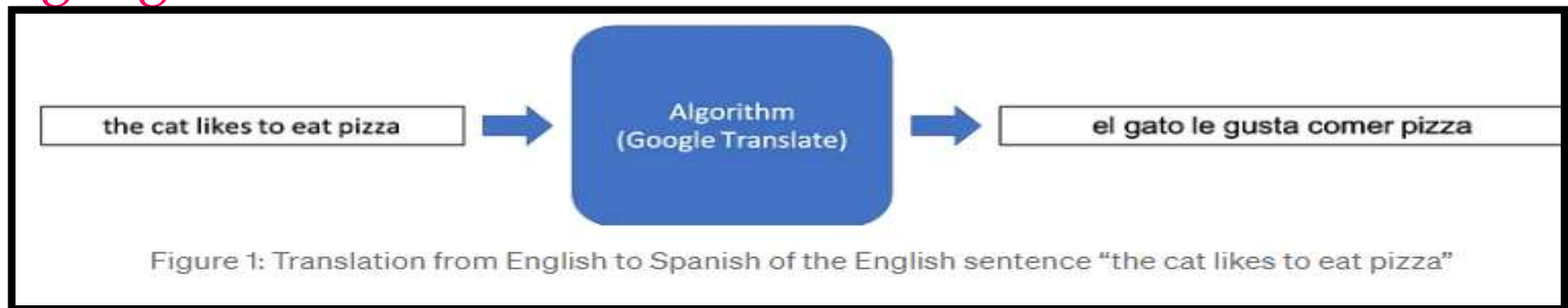
- GANs can be used to generate new examples for image datasets in various domains, such as **medical imaging, satellite imagery, and natural language processing**

DEEP LEARNING APPLICATIONS



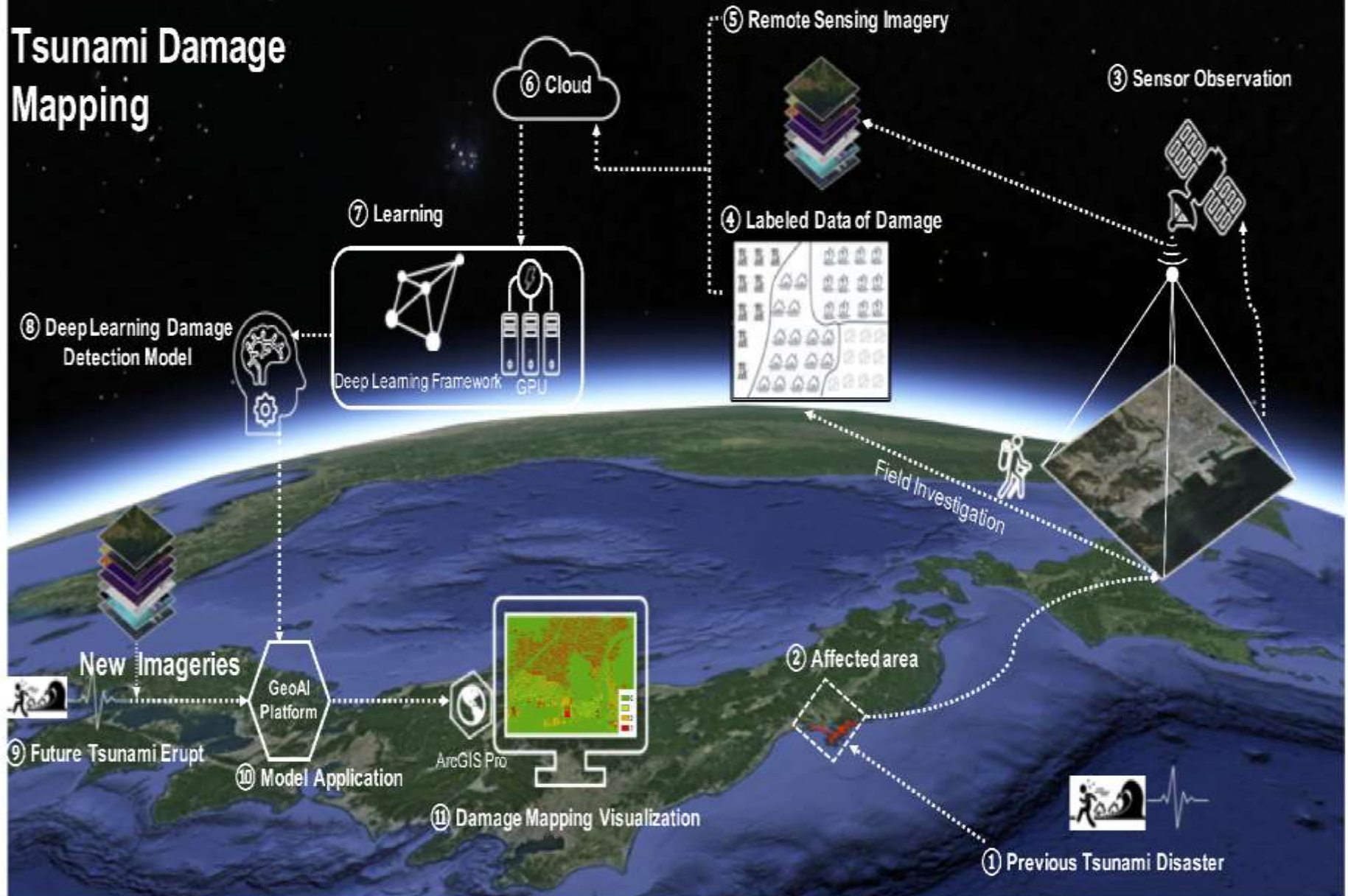
- 1. Instant Visual Translation: With deep learning, identification of text on the images is possible. Once identification completes, it translates the text immediately and recreates the image with translated text.

For example, Suppose you visit an unknown country whose local language is not known to you. An app like google translator converts the text of an image in an understandable language.



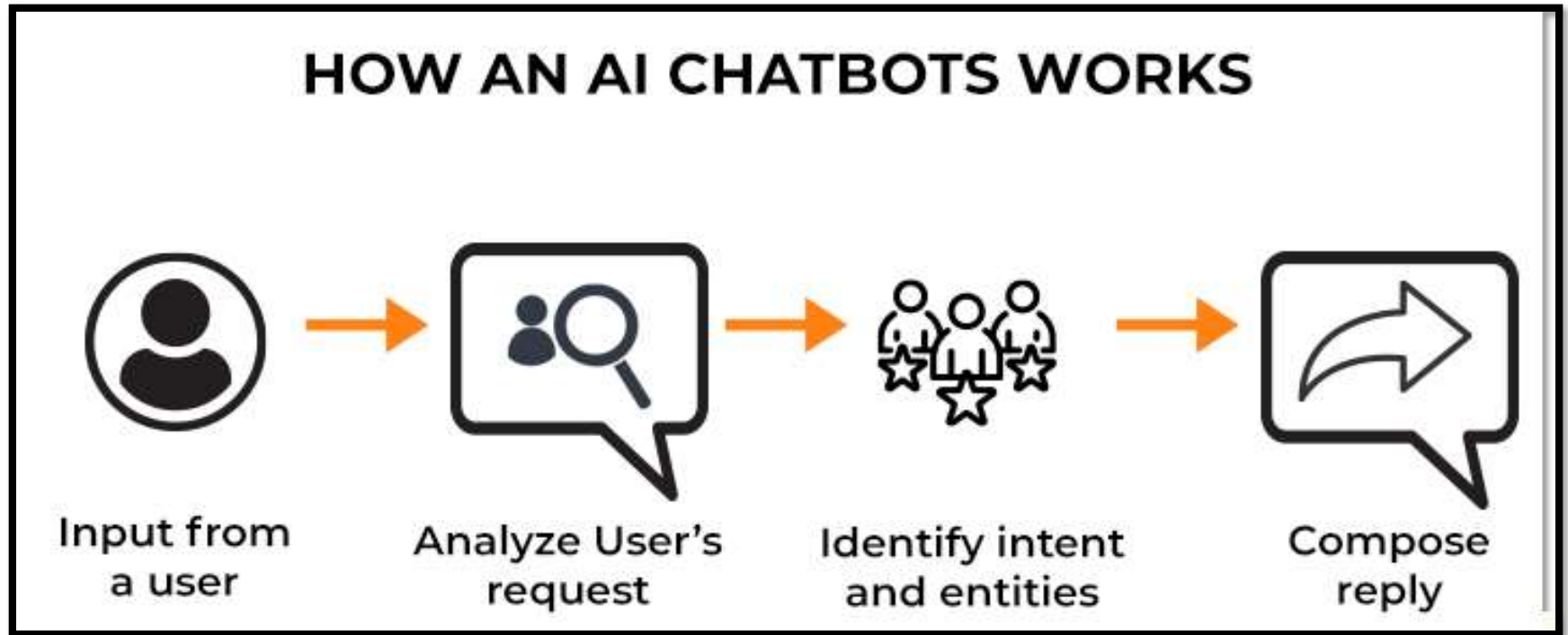
- With the power of deep learning, **Neural Machine Translation (NMT)** has arisen as the **most powerful algorithm** to perform this task.
- This state-of-the-art algorithm is an application of deep learning in which **datasets of translated sentences** are used to **train a model capable of translating between any two languages**.
- **This conversion is composed by using Two Recurrent Neural Networks(RNN)**

- 2. Predicting the Future: Deep neural network helps in the prediction of earthquakes, tsunamis, cyclones, etc. So that preventive measures can be taken to save lives from falling into the grasp of natural calamities.
 - This can be done by using Deep Neural Networks and Conversions can be done by Auto Encoders.

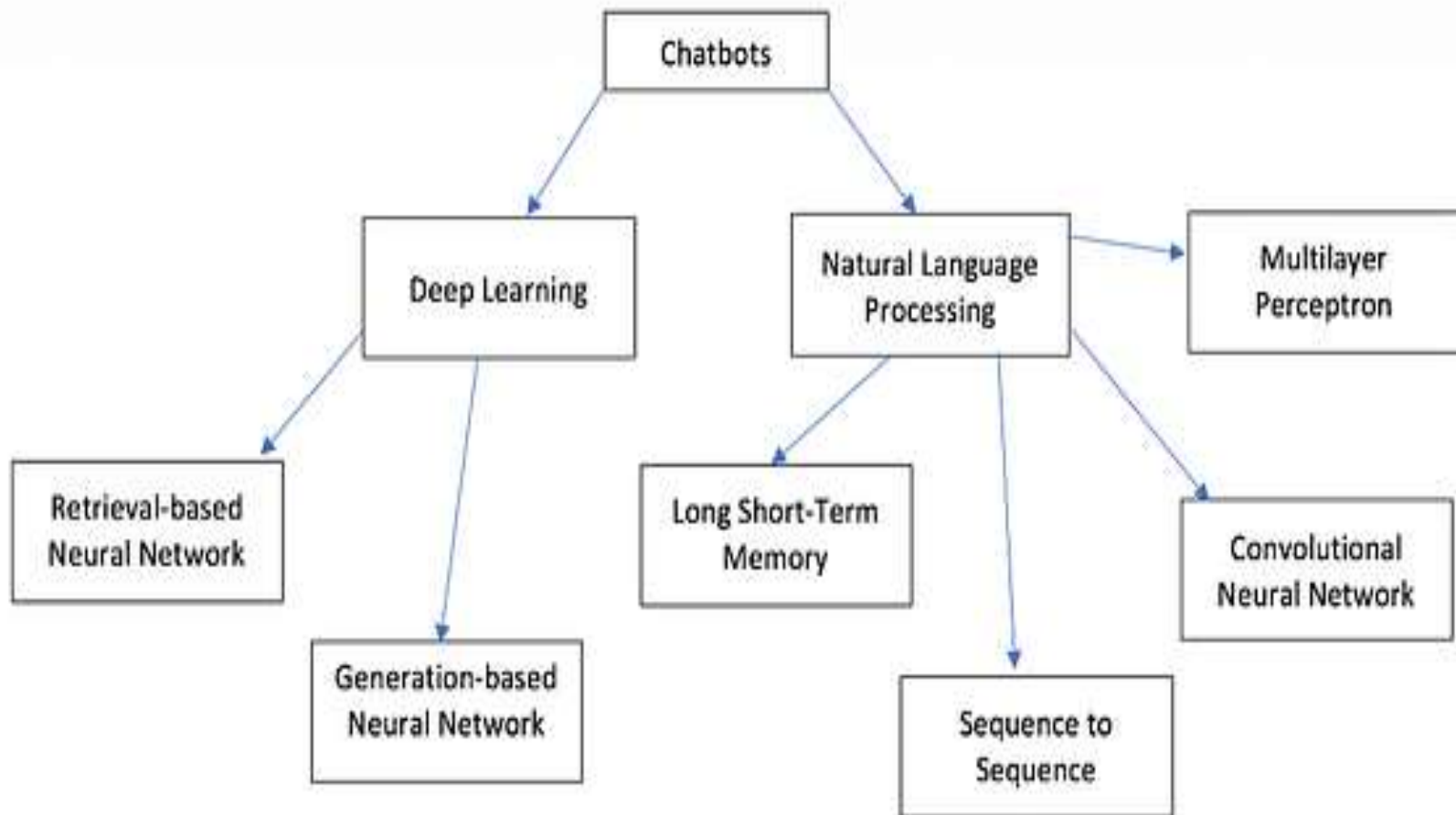


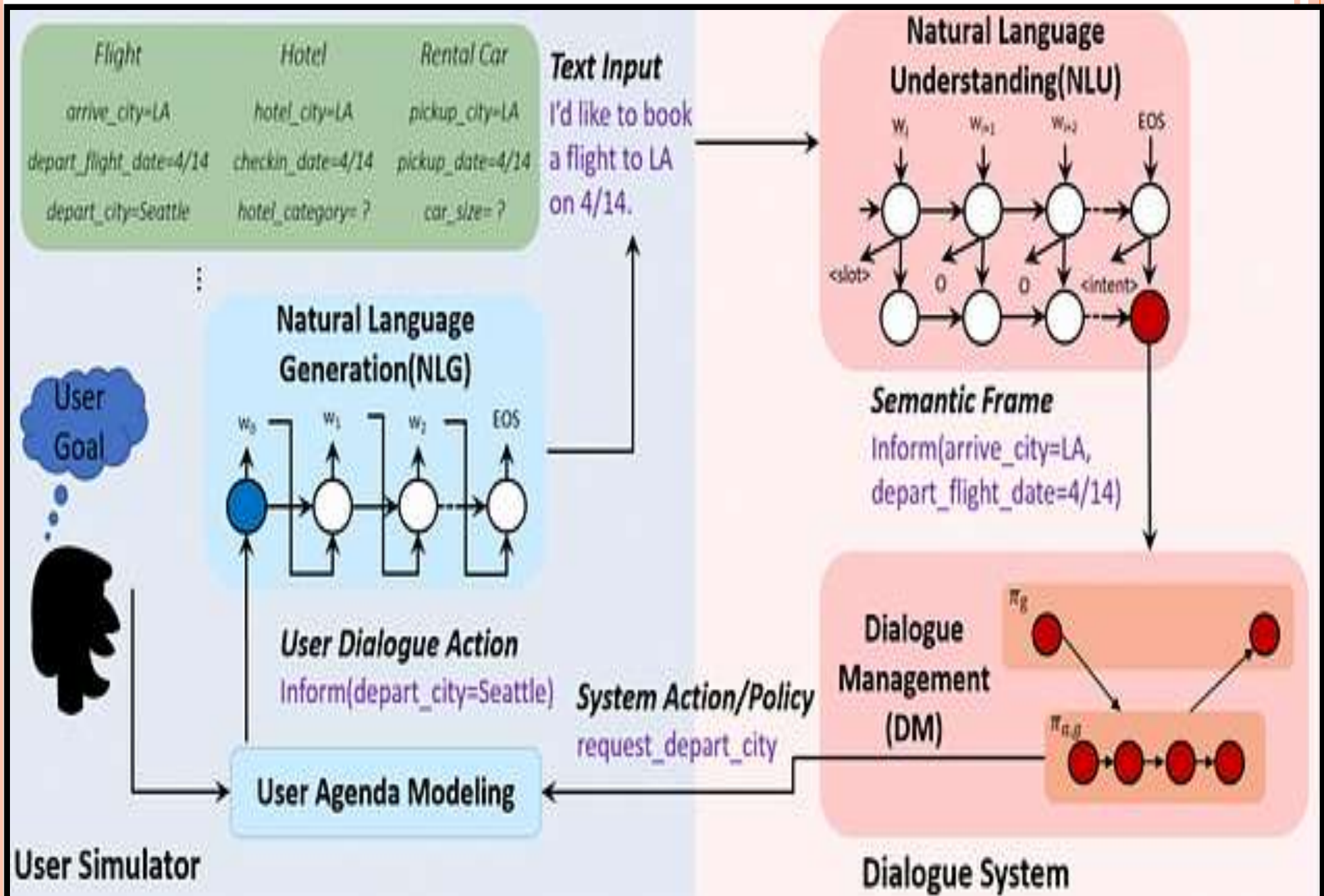
- 3. Chatbot: With the help of **deep learning Applications**, **chatbots** are becoming smarter day by day.
- Amazon, Flipkart and many e-commerce websites are using chatbots for customer services.
- **Transportation apps** like Ola and Uber are also implementing chatbots to provide personalized aid.
- Siri is also a good example of a chatbot.
- Chatbots work by adopting mainly 3 classification Methods: Pattern Matching, Recurrent Neural Network, Artificial Neural Networks

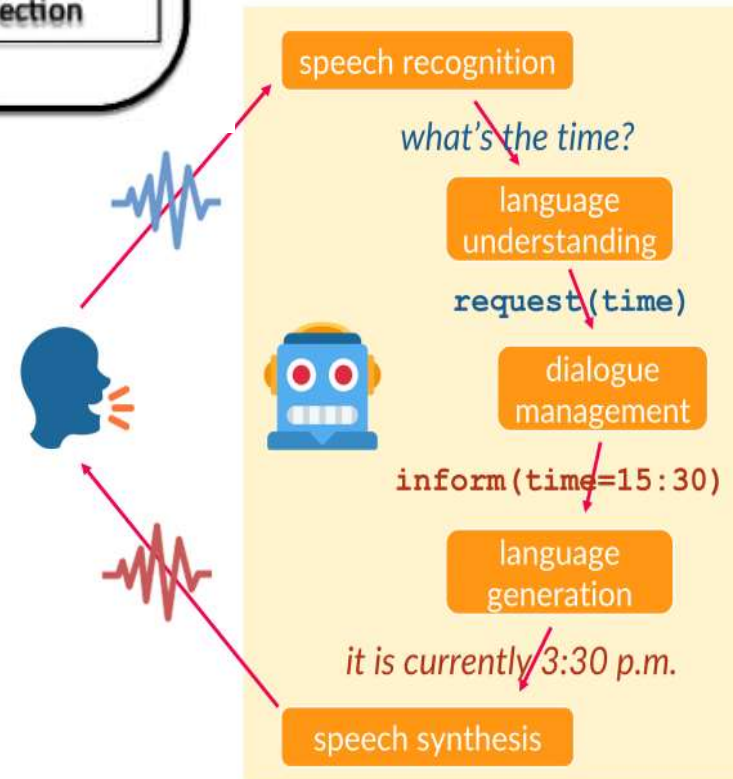
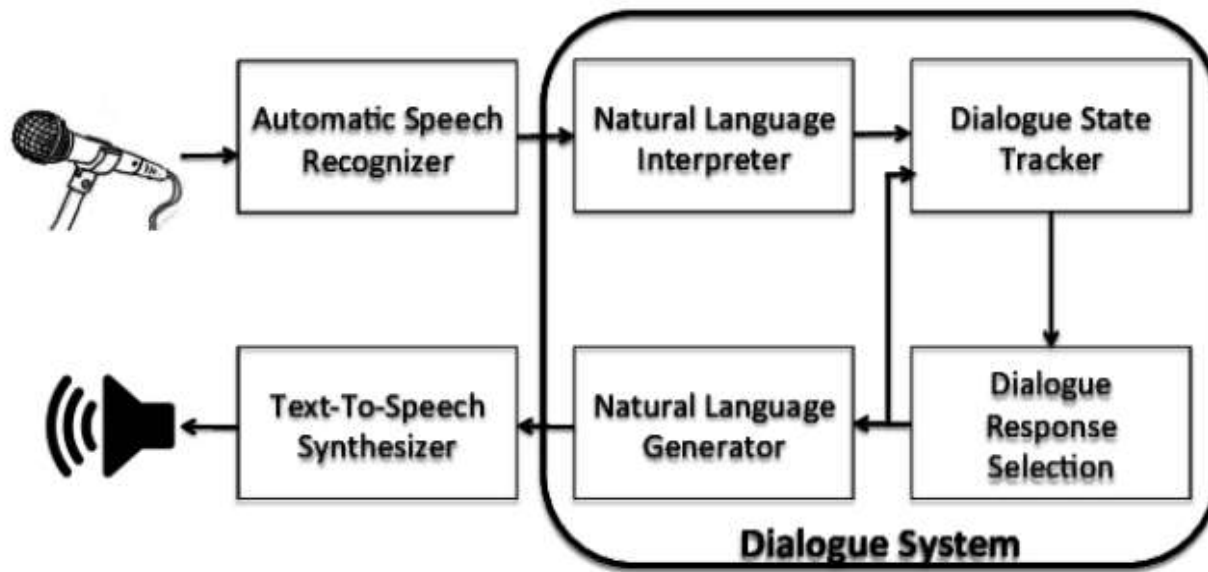
How AI Chatbot Works



The below picture illustrate the **conceptual map of Chatbot using Deep learning**





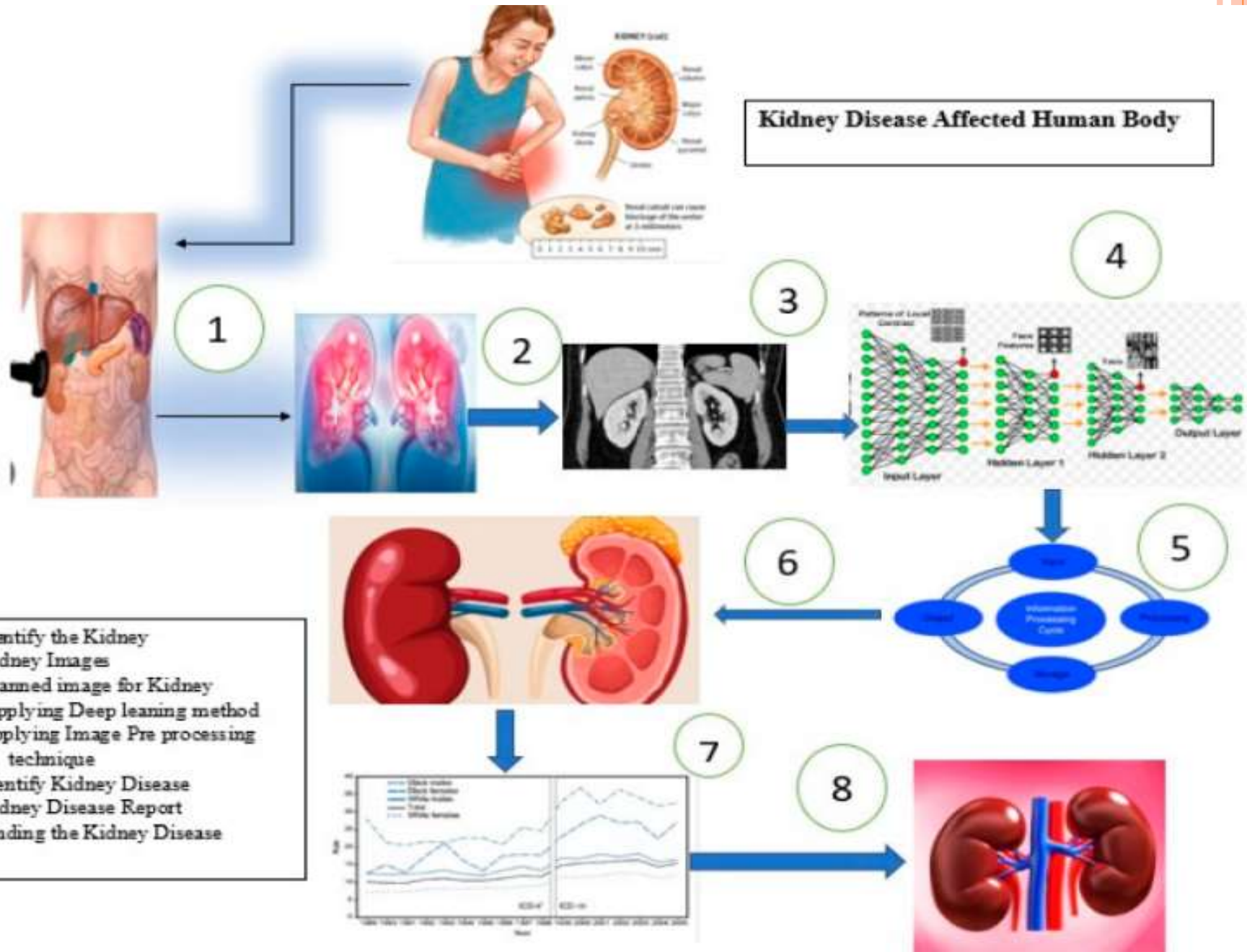


○ 4. Medical Care:

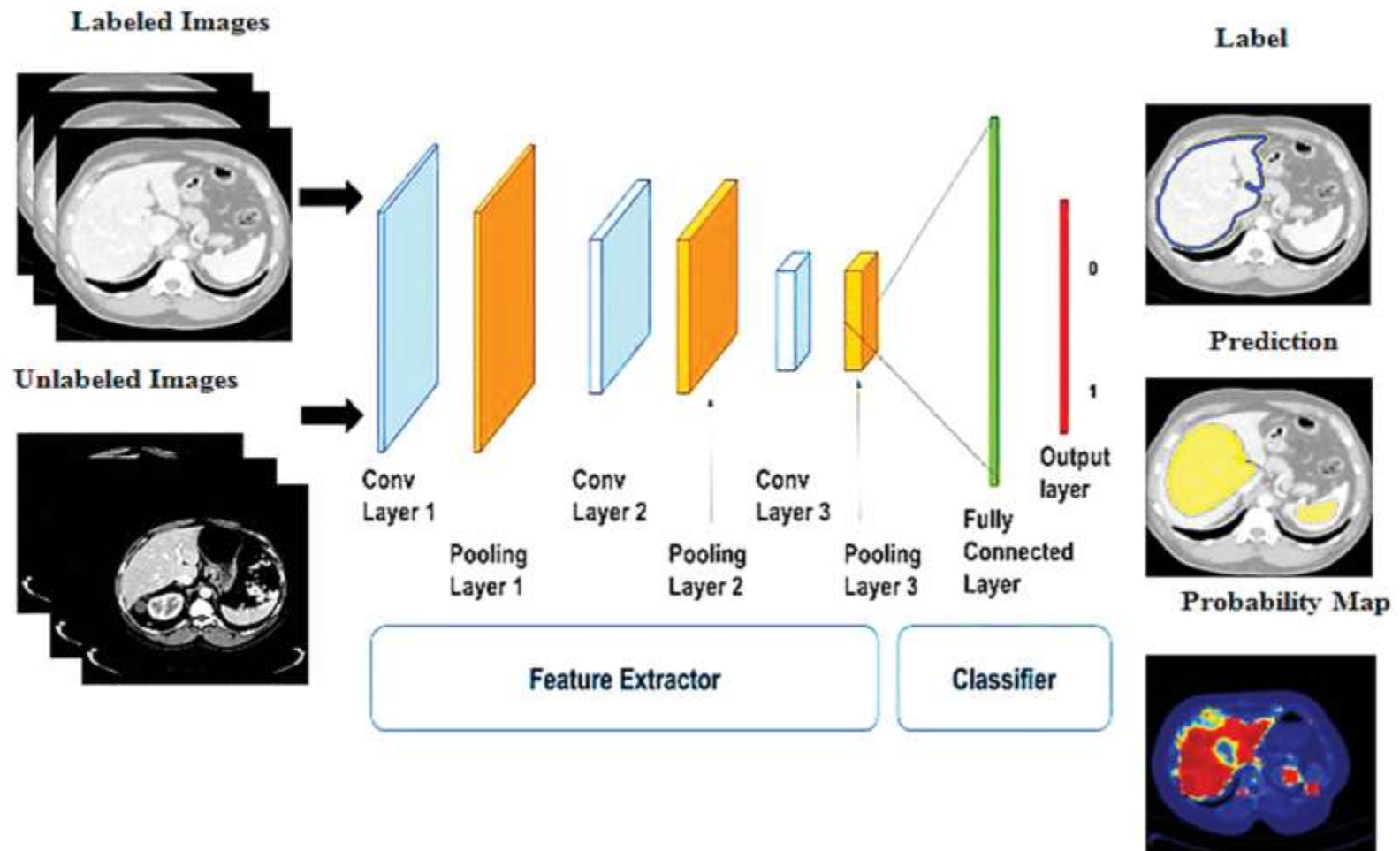
- a) Deep learning helps in detecting cancer cells and analyzing the MRI images to give elaborative results.
- b) Google has made **Google AI eye doctor software**. It examines **retina scans** and identifies **diabetic retinopathy**, which can **cause blindness**.

Suppose we have **taken Kidney Disease Prediction** it **will follow the structure**.

Kidney Disease Affected Human Body



- 1- Identify the Kidney
- 2- Kidney Images
- 3- Scanned image for Kidney
- 4- Applying Deep learning method
- 5- Applying Image Pre processing technique
- 6- Identify Kidney Disease
- 7- Kidney Disease Report
- 8- Finding the Kidney Disease



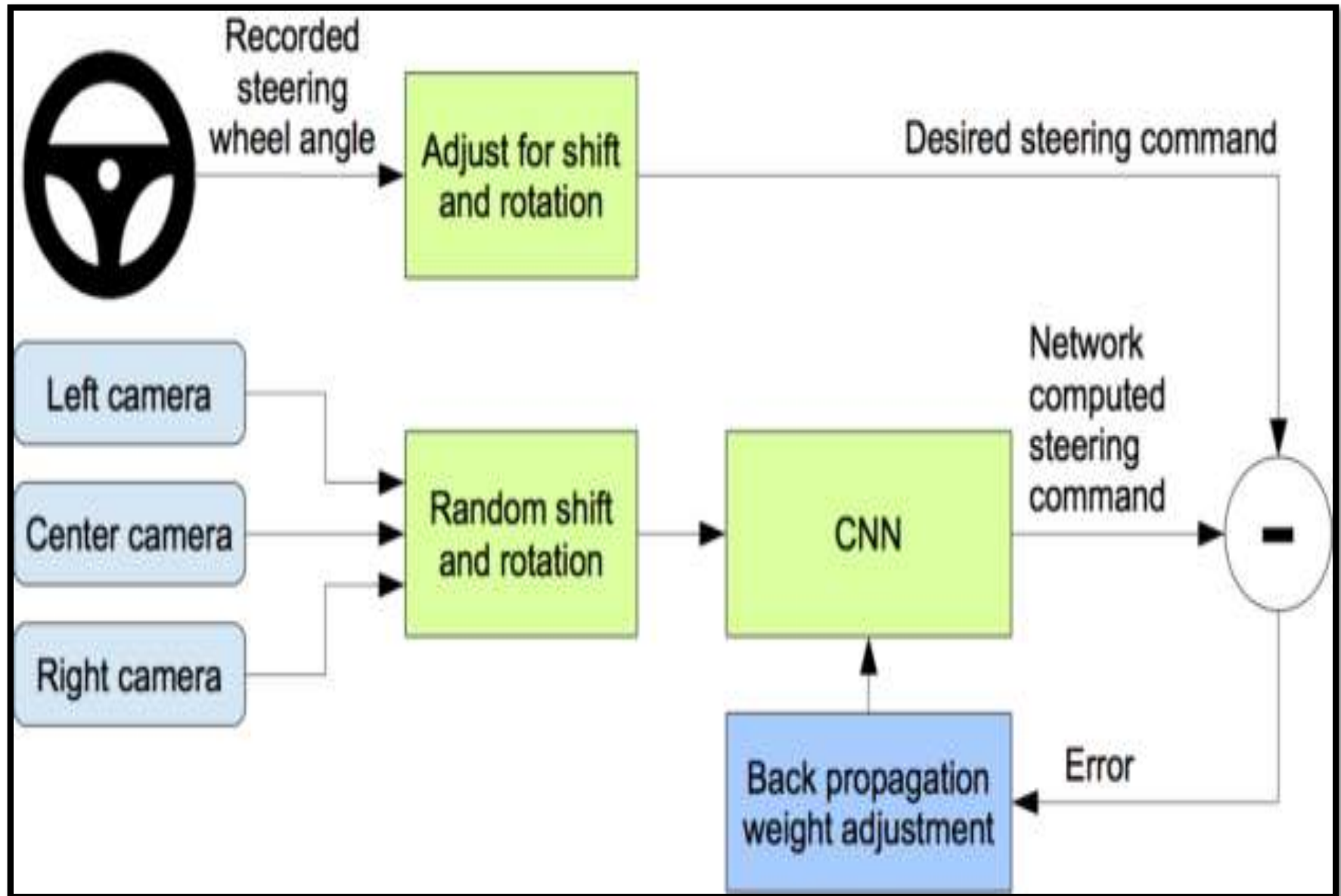
- This image , **we can take Convolution Neural Networks are used for Feature Extraction and Classification.**

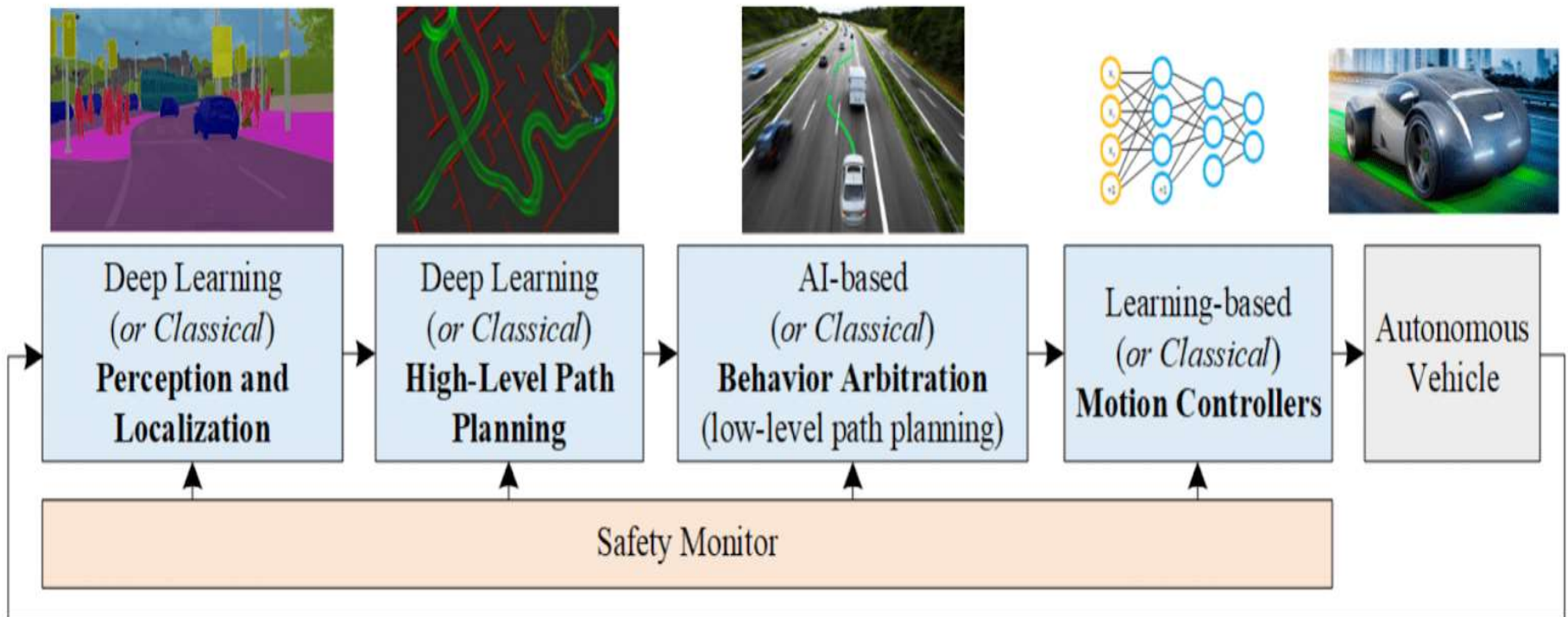
○ 5. Self Driving Cars:

Google has made an **amazing self-driven car**. This car operates on a **combination of sensors and software**.

Car can classify objects, people, traffic signs and signals. It also detects the road works. It uses **Lidar Technology**.

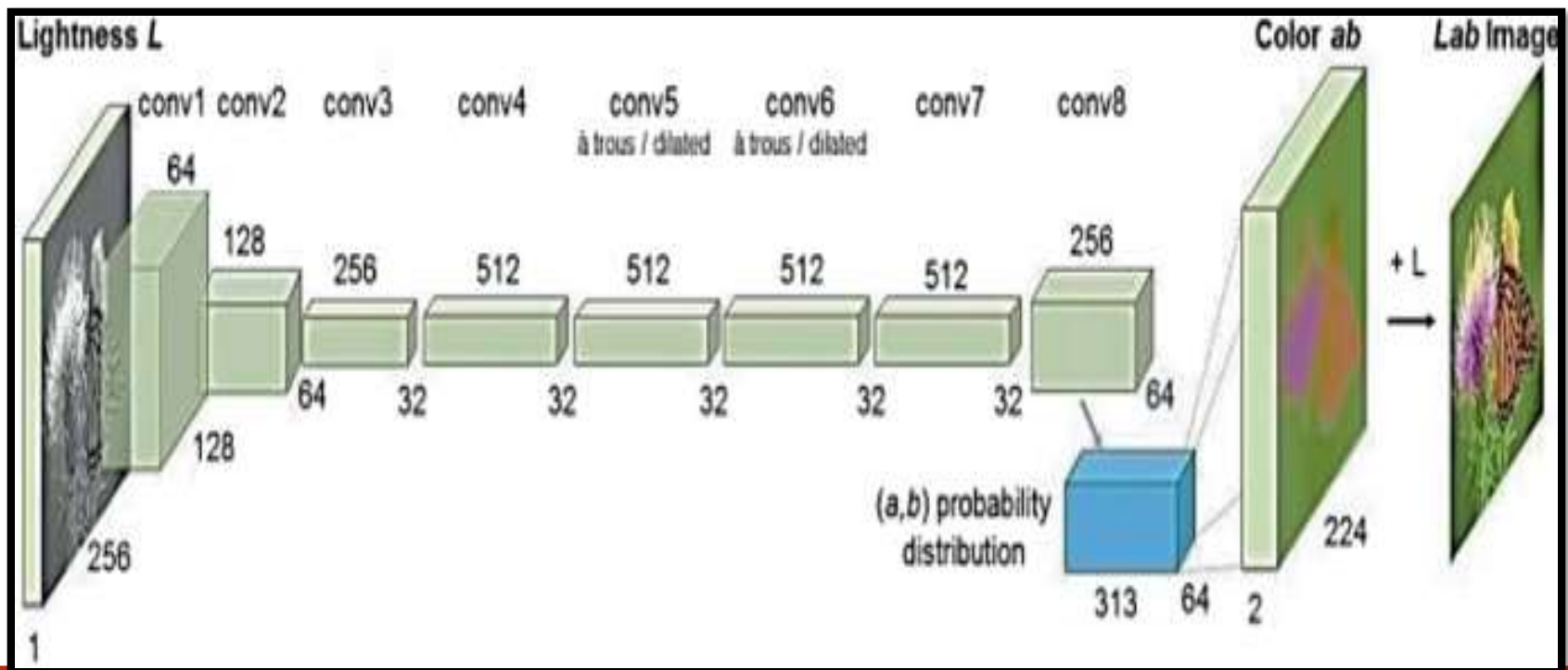
- A self-driving car (**sometimes called an autonomous car or driverless car**).
- **Tesla**, the most **popular car manufacturing company** is working on self-driving car.





○ 6. Colorizing the Images

Deep learning makes it **possible to color the black and white images**. It uses **convolutional neural network** for the same.



CONTD..



○ 7. Read Lip Movements:

Lip Reading is decoding text from the movement of the speaker's mouth. It plays a crucial role in human communication and speech understanding.

Using deep learning with python, oxford and Google's scientists developed a neural network known as lipnet. It can read people's lips with 93% success.

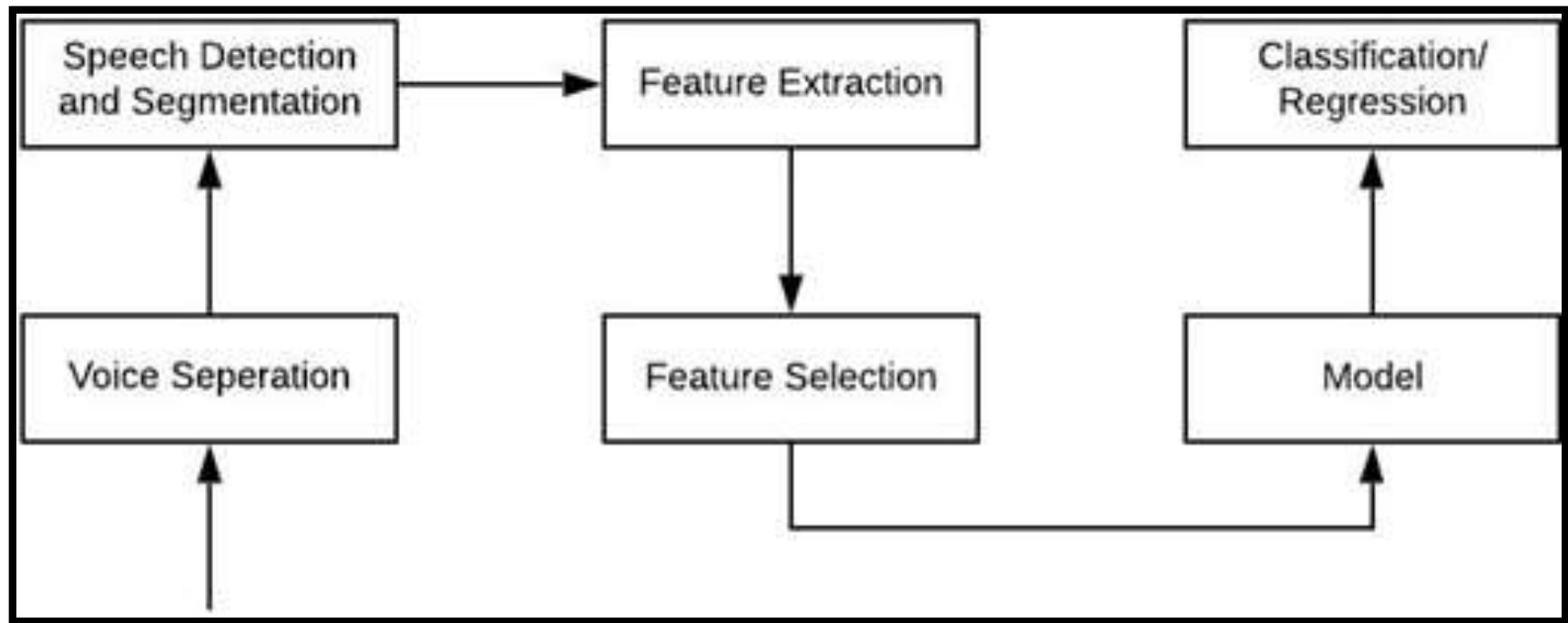
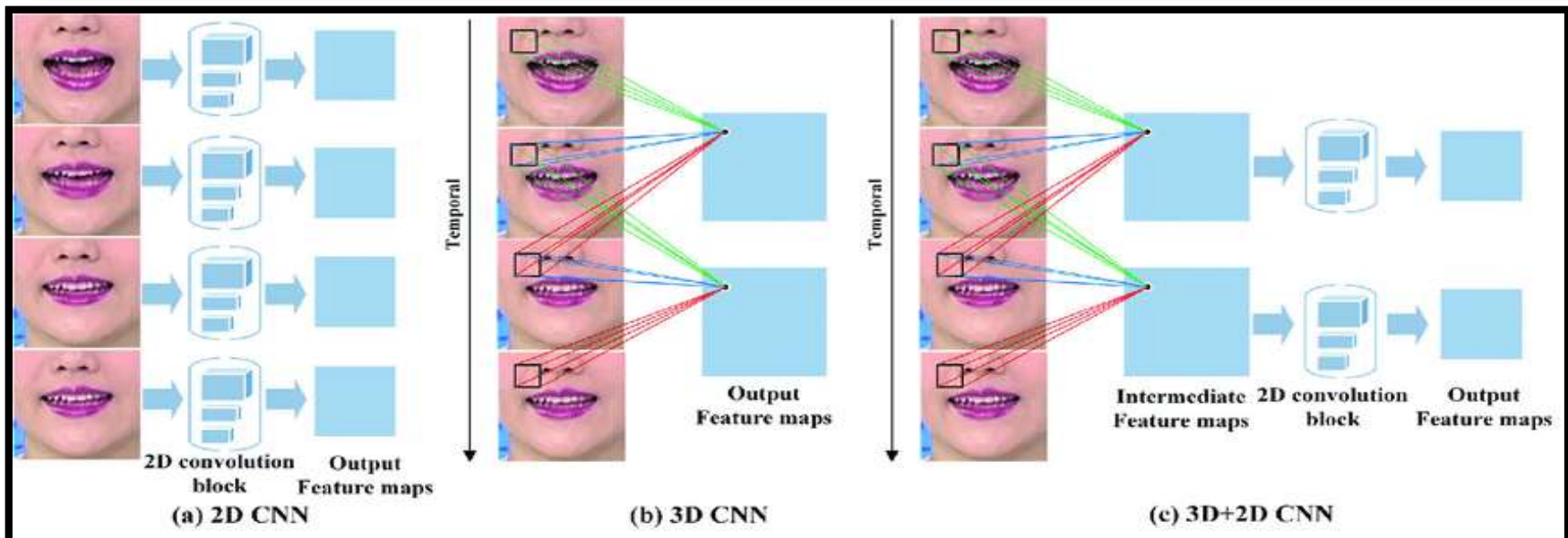
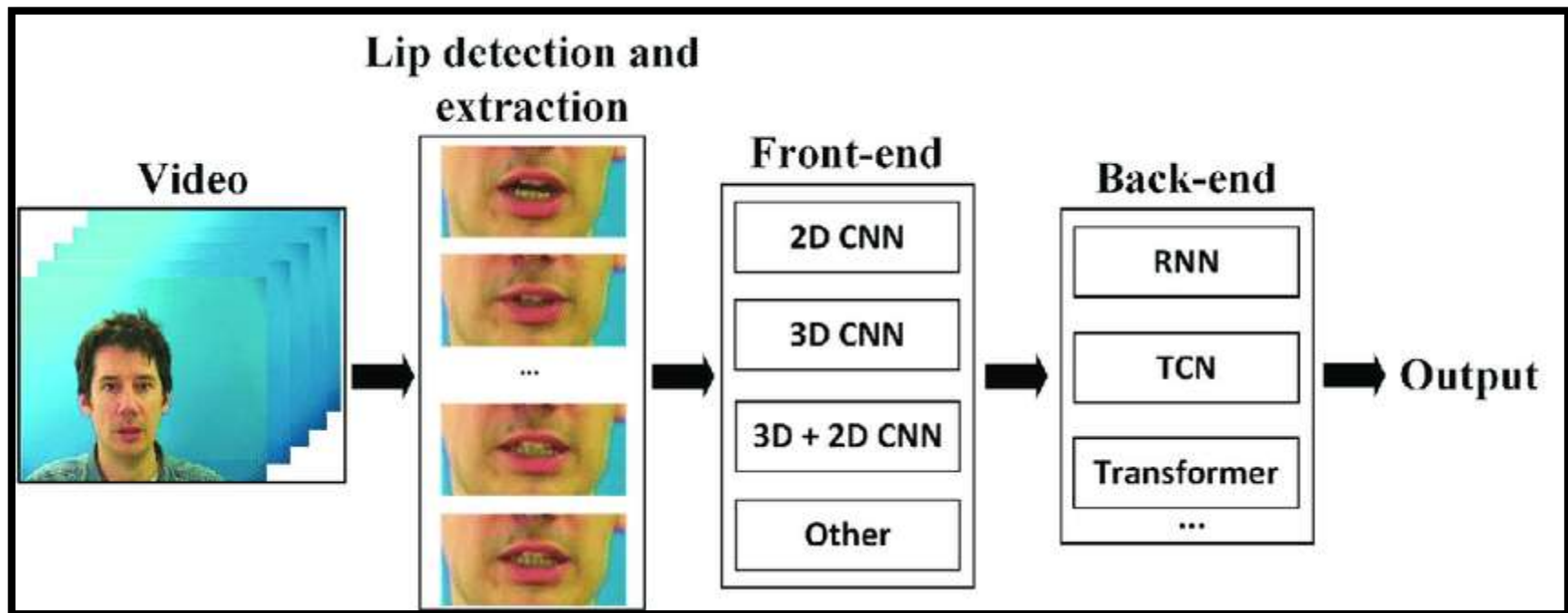


Fig: Lip Reading Process

This can be done by using Convolution Neural Networks, Long Short Term Memory, Auto Encoders, Recurrent Neural networks etc.



○ 8. Photo Descriptions:

System tends to **automatically classify photographs**. Deep Learning has the **capacity** to narrate every existing element in the image. **Deep learning networks** can identify the captivating areas of the **images and can describe them into the sentences**.

bouquet of
red flowers

tablet

bottle of water

glass of water with
ice and lemon

cup of coffee

dining table
with breakfast
items

plate of fruit

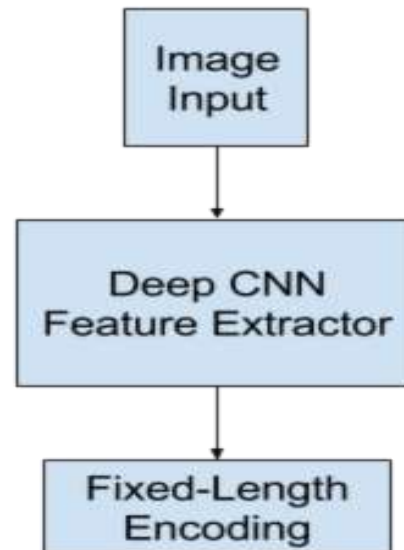
banana
slices

fork

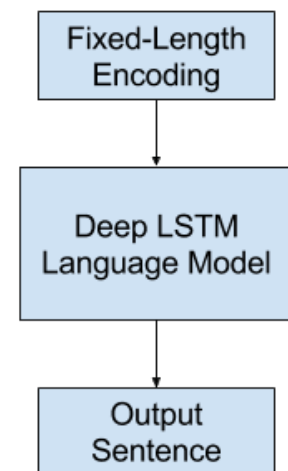
a person
sitting at a
table



- Here we can use Neural Captioning Model.
- In this Model again **divided into 2 types**
 - **Feature Extraction:** The feature extraction model is a neural network that **given an image is able to extract the salient features**, often in the form of a fixed-length vector.



- **Language Model:** a language model predicts the probability of the next word in the sequence given the words already present in the sequence. It is popular to use a **recurrent neural network**, such as **a Long Short-Term Memory network, or LSTM**, as the language model. Each **output time step generates a new word in the sequence.**



○ 9. Deep Dreaming:

It is a very interesting application of deep learning.

As the name suggests, it **allows the system hallucinates on the top of an image and generate the resembling dream.**

Dreams depend **upon the type of neural network** .

For example, **Horizon lines** tend to **get filled with towers and pagodas**. Rocks and trees turn into **buildings**. Birds and insects appear in **images of leaves**.



Horizon



Towers & Pagodas



Trees



Buildings



Leaves



Birds & Insects

○ 10. Advertising:

- Deep learning has **transformed the advertising field**. **Publishers and advertisers** use deep learning to **increase the relevancy of their ads**.
- Deep learning makes it possible for publishers to leverage the **content to create the real time bidding for the ads**.

DEEP LEARNING FRAMEWORKS



Caffe



- 1. Tensor Flow: It is one of the most popular deep learning frameworks. Developed by the Google Brain team, TensorFlow supports languages such as Python, C++, and R to create deep learning models along with wrapper libraries. It is available on both desktop and mobile.
- The most well-known use case of TensorFlow has got to be Google Translate coupled with capabilities such as natural language processing, text classification, summarization, speech/image/handwriting recognition, forecasting, and tagging.





○ 2. TORCH/PyTorch:

- PyTorch is a **Python-based scientific computing package** serving **two broad purposes**:
- A replacement for **NumPy** to use the power of GPUs and other accelerators. It is implemented by using **python**.
- An **automatic differentiation library** that is useful to **implement neural networks**.
- It is a **Lua based deep learning framework** and is used widely amongst **industry giants such as Facebook, Twitter, and Google**.
- It is widely used **Natural Language Processing, Computer Vision Applications**

○ 3. DEEPLARNING4J:

The logo for DEEPLARNING4J, featuring the text "DEEPLARNING4J" in a bold, white, sans-serif font against a dark brown rectangular background.

- The **j** in **Deeplearning4j** stands for **Java**. Needless to say, it is a **deep learning library for the Java Virtual Machine (JVM)**. It is developed in **Java** and supports other **JVM languages like Scala, Clojure, and Kotlin**.
- **Deeplearning4j** comes with **deep network** support through **Convolution Neural Networks (CNN)**, **Recurrent Neural Networks (RNN)**, **Recursive Neural Tensor Network (RNTN)** and **Long Short-Term Memory (LSTM)**.

○ 4. The Microsoft Cognitive Toolkit/Cntk:



- Popularly known for easy **training and a combination of popular model types across servers**, the Microsoft Cognitive Toolkit (earlier known as CNTK) is an **open-source deep learning framework** to train **deep learning models**. It performs efficient **Convolution Neural Networks** and **training for image, speech, and text-based data**.
- Supported by interfaces such as **Python, C++**.

- The implementation of Reinforcement Learning models or **Generative Adversarial Networks (GANs)** can be done quickly using the toolkit. The Microsoft Cognitive Toolkit is known to **provide higher performance and scalability** as compared to toolkits like **Theano** or **TensorFlow while operating on multiple machines.**
- The Microsoft Cognitive Toolkit supports **both RNN and CNN type of neural models** and is thus capable of **handling image, handwriting, and speech recognition problems.**

○ 5.Keras:



- **Keras library** was developed, and Written in **Python**, the **Keras neural networks library** supports **both convolutional and recurrent networks** that are capable of running on **either TensorFlow or Theano**.
- The **primary usage of Keras** is in **classification, text generation, and summarization, tagging, translation along with speech recognition, and others**.
- The main used of Keras is , it supports **multiple deep learning backends**

○ 6.ONNX:



- **ONNX or the Open Neural Network Exchange** was developed as an **open-source deep learning ecosystem**. Developed by **Microsoft and Facebook**, ONNX proves to be a **deep learning framework** that enables **developers to switch easily between platforms**.
- **ONNX** models are **natively supported in The Microsoft Cognitive Toolkit, Caffe2, MXNet, and PyTorch**. It also provides converters for **different machine learning frameworks** like **TensorFlow, CoreML, Keras, and Sci-kit Learn**.

○ 7.mxnet:



- MXNet (pronounced as mix-net) is a deep learning **framework** that is supported by **Python, R, C++, and Julia**.
- MXNet supports **Long Short-Term Memory (LSTM) networks**, along with both **RNN and CNN**. This deep learning framework is known for its capabilities in **imaging, handwriting/speech recognition, forecasting as well as NLP**.

○ 8.Caffee:

Caffe

- Caffe (Convolutional Architecture for Fast Feature Embedding) is an open source deep learning framework that supports a variety of **deep learning architectures** such as CNN, RCNN, LSTM and **fully connected networks**. Caffe is most popular for **image classification and segmentation tasks**.
- Caffe is primarily **a C++ library** and exposes a **modular development interface**, but not every situation requires custom compilation. Therefore, **Caffe offers interfaces** for daily use by way of the **command line**, Python and **MATLAB**.

LEARNING BEYOND CURRICULUM

- 1. Coursera
- 2. NPTEL
- 3. MIT Online Certification Courses





1. Introduction to Deep Learning
2. Deep Learning specialization
3. Neural Networks and Deep Learning
4. TensorFlow developer professional Certificate
5. Deep Learning for Object Detection



1. Introduction to Deep Learning - IIT Ropar
2. Deep Learning for Computer Vision
3. Deep Learning – IIT Kharagpur



**Massachusetts
Institute of
Technology**

1. Deep Learning for AI and Computer Vision
2. Introduction to Deep Learning.
3. Deep Learning – Mastering Neural Networks.
4. Advances in Imaging and Deep Learning: Medical, VR-AR, and Self-Driving Cars



(DEEP LEARNING)

IV B.TECH I SEM

(PVP-20)



DEEP
LEARNING

Topics :

- **Common Architectural Principles of Deep Networks**
 - **Parameters**
 - **Layers**
 - **Activation Functions**
 - **Loss Functions**
 - **Hyper Parameters**

1. PARAMETERS

- Parameters in deep learning refer to the **learnable variables** that determine the **behavior and performance of a neural network**.
- 1. Epoch: Represents **one iteration over the entire dataset** (everything put into the training model).
- An **epoch** completes **once a whole dataset** has undergone **forward propagation and back propagation**.



- 2. Batch – An epoch is made up of **batches**. Sometimes the whole dataset can not be passed through the neural network at once **due to insufficient memory or the dataset being too large.**
- We divide the entire dataset into **smaller numbers of parts called batches.** These batches are **passed through the model for training.**



- 3. Iterations: The **total number of batches** needed to **complete one epoch** is called iteration.
- For example, the **dataset consists of 1000 images**. We divide it into ten batches of size 100 each.
- **One iteration is completed** when **one batch** passes through a **neural network** that is **forward propagation** and **back propagation**.
- The **following equation** shows the relation between **batch size and iteration** with **epoch** and can be used to calculate epoch size:
$$\text{epoch_size} = \text{batch_size} \times \text{iterations}$$

- **Eg:** Let's have the **training dataset** having **1000 training samples**. And we want to **break the dataset into a batch size of 100**. Suppose we are **going for 5 epochs**, Then the **total number of iterations will be :**

- **Ans:**

Total number of training samples = **1000**

Batch size = **100**

Total number of iterations = Total number of training samples / Batch
size = **1000/100=10**

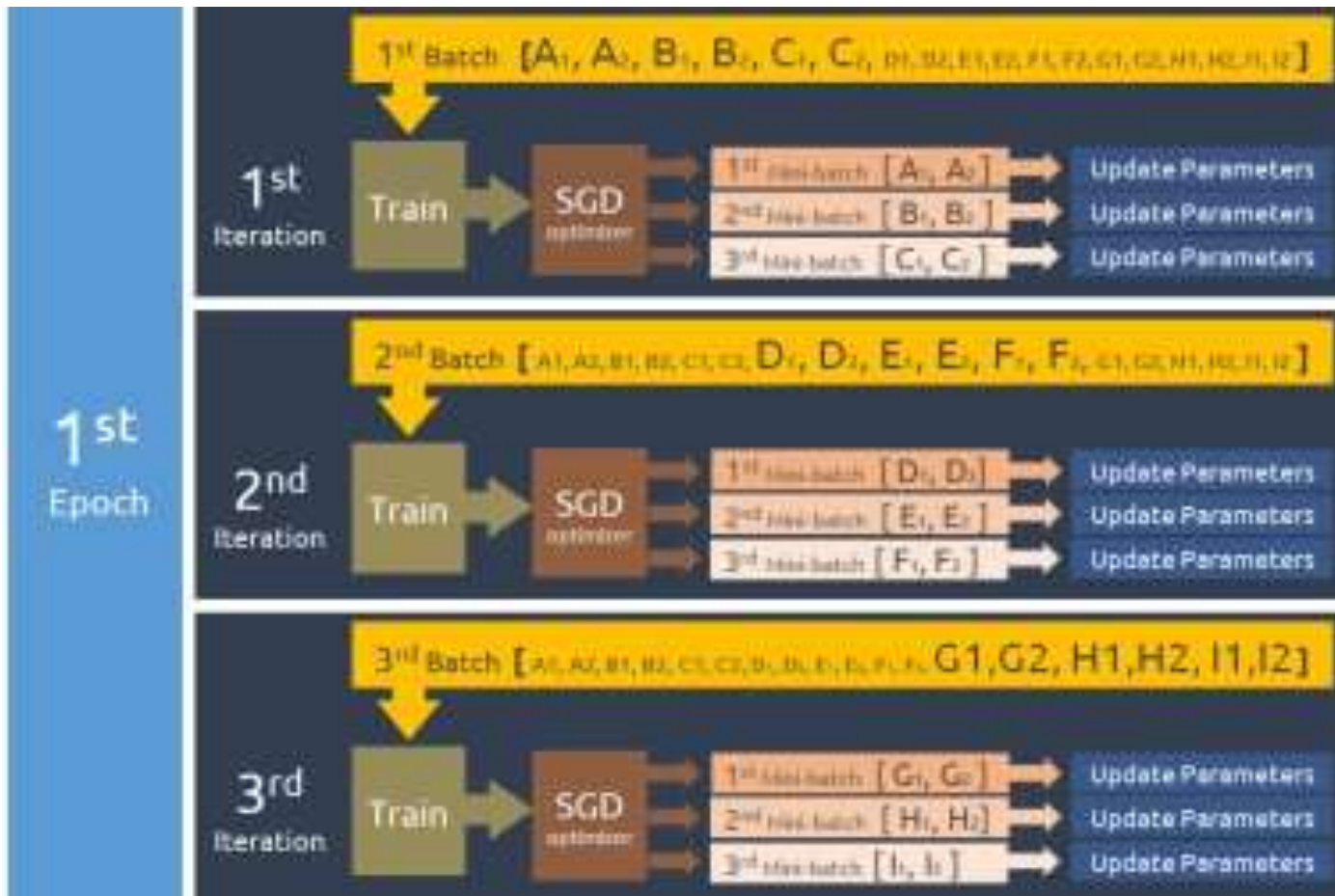
Total number of iterations = **10**

One epoch = **10 iterations**

Total number of iterations in 5 epochs = **10*5 = 50 iterations.**



- The following illustration can help in understanding the difference between the above terminologies:



2. LAYERS

- The mother art is architecture. Without an architecture of our own we have no soul of our own civilization.

-Frank Lloyd Wright

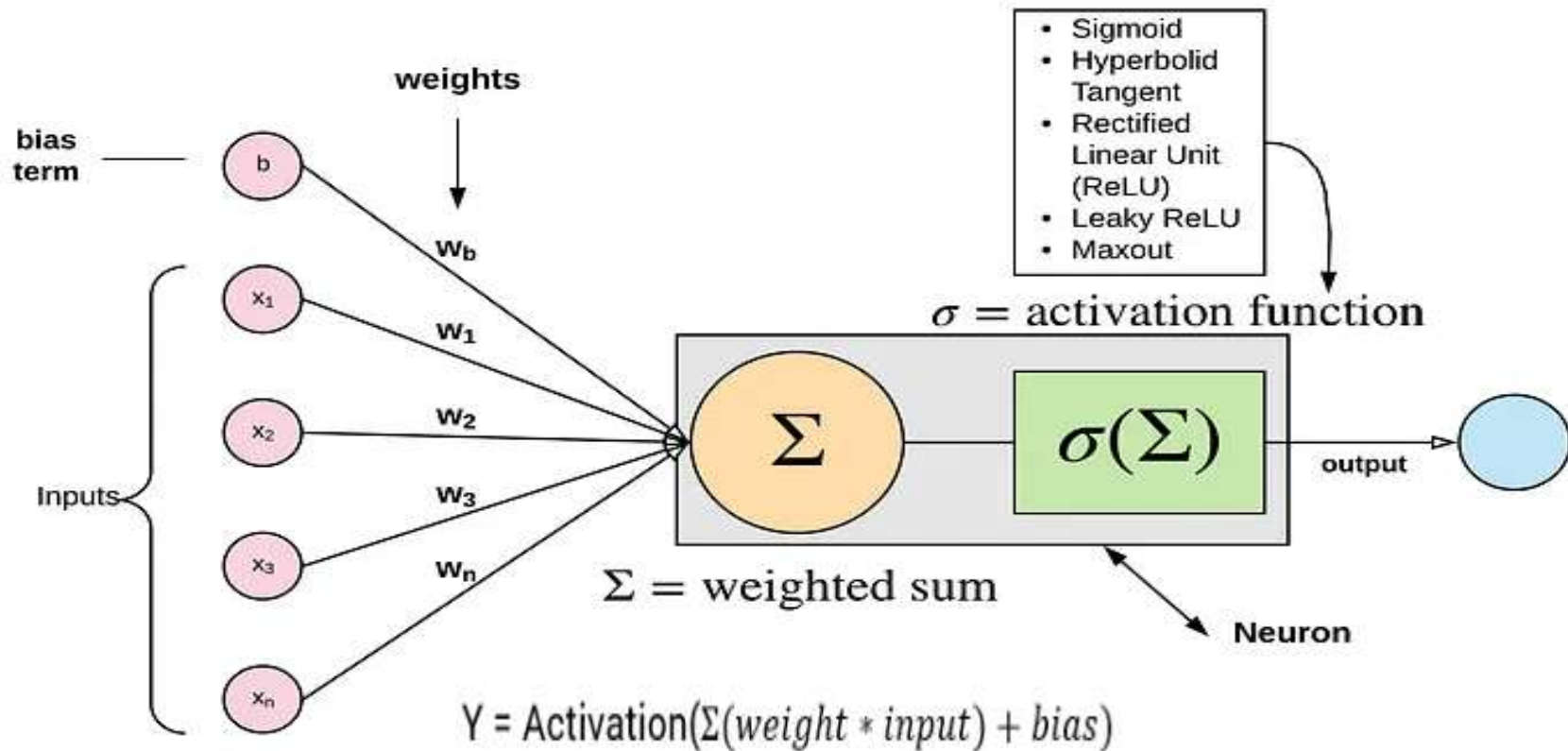
We introduced four major network architectures:

- **Unsupervised Pre trained Networks (UPNs)**
 - **Auto Encoders**
 - **Deep Belief Networks(DBNs)**
 - **Generative Adversarial Networks (GANs)**
- **Convolutional Neural Networks (CNNs)**
- **Recurrent Neural Networks**




ACTIVATION FUNCTIONS

- In the neural network **activation function** plays an important role.
- The activation function determines the **output of a deep learning network**.
- An activation function is a **mathematical operation** applied to the **output of a neuron** in a neural network. It determines **whether a neuron should be activated or not**, based on the **weighted sum of its inputs**.
- An activation function is **an internal state of a neuron** that converts an **input signal to an output signal**.



The **primary role of the Activation Function** is to transform the summed weighted input from the node into an **output value** to be fed to the **next hidden layer or as output**.

- Activation functions provide **non-linear properties** to the neural network. **Without the activation function**, the output values from the neurons can range between **(- infinity) to (+infinity)**.
 - For Eg: A **group of 4 friends is deciding to go on a vacation**. They have narrowed down their options to **three countries. Thailand, Jamaica, and Dubai**.
 - In order to chose the **final destination** they **each decide to vote for the countries** on a scale of **-10 to 10**. **-10 being least interested** and **10 being most interested**. **0 can be considered as neutral**.
- 

- From the **table below**, you can notice, **Bkon** for e.g. is **most interested in Thailand** and **least interested in visiting Jamaica**.

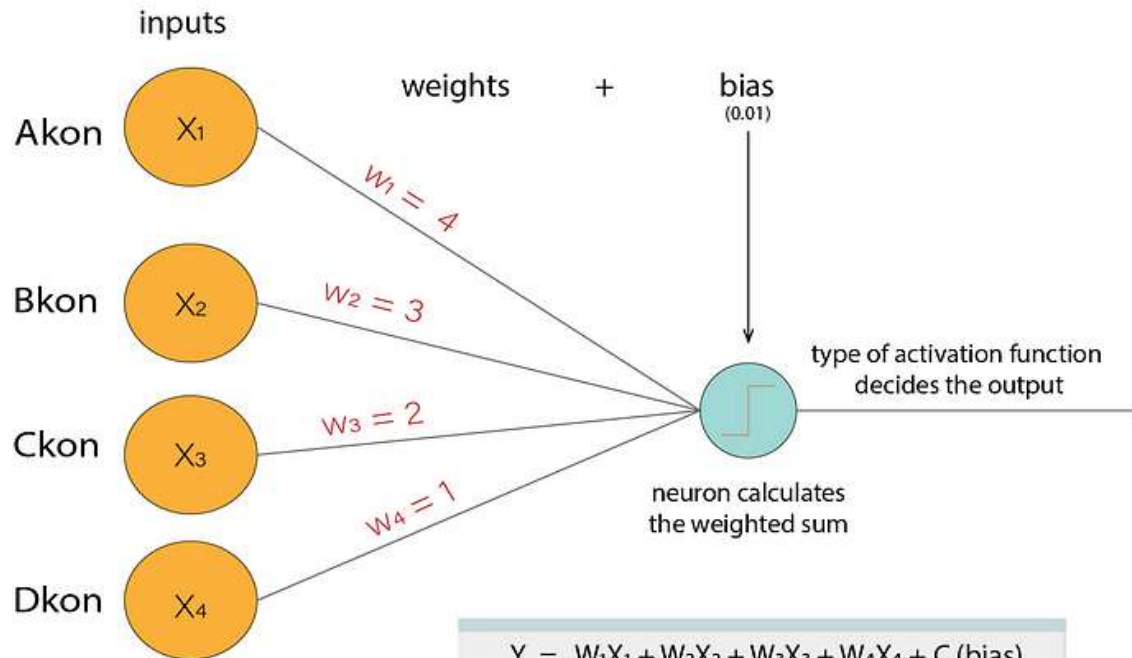
Votes for vacation destinations				
	Akon	Bkon	Ckon	Dkon
Thailand	-9	10	3	-2
Jamaica	-4	5	-1	8
Dubai	-3	6	-2	6



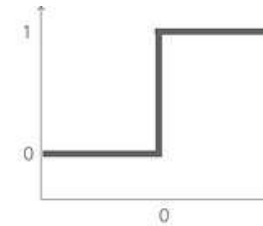
- Let us assign **initial random weights** to **each variable**.
For the sake of simplicity in calculations, we will keep the **bias value = 0.01 or simply 0**.
- Observe the figure below: The neuron calculates the weighted sum of its inputs and provides an output value.
- The neuron here, **without an activation function** would **provide 1 value** each for **Thailand, Jamaica, and Dubai**.
i.e. **Thailand = -2**, **Jamaica = 5**, and **Dubai = 8**.



CONTD..

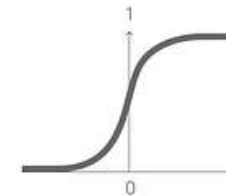


	$Y = W_1X_1 + W_2X_2 + W_3X_3 + W_4X_4 + C \text{ (bias)}$		
Thailand	$(4 \times -9) + (3 \times 10) + (2 \times 3) + (1 \times -2)$	=	-2
Jamaica	$(4 \times -4) + (3 \times 5) + (2 \times -1) + (1 \times 8)$	=	5
Dubai	$(4 \times -3) + (3 \times 6) + (2 \times -2) + (1 \times 6)$	=	8



Step

Thailand: 0
Jamaica: 1
Dubai: 1



Sigmoid (Probabilities)

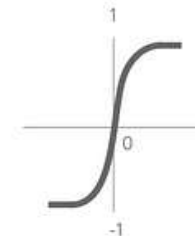
Thailand: 0.002
Jamaica: 0.85
Dubai: 0.94

*probabilities do not add to 1

Softmax (Probabilities)

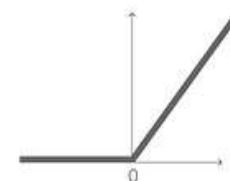
→ Thailand: 0.02
Jamaica: 0.22
Dubai: 0.76

*probabilities add to 1



tanh

Thailand: -0.2
Jamaica: 0.5
Dubai: 0.8




ReLU

Thailand: 0
Jamaica: 5
Dubai: 8

- There are 3 types of Activation Functions

- 1) Binary Step Functions
- 2) Linear Activation Function
- 3) Non Linear Activation Functions

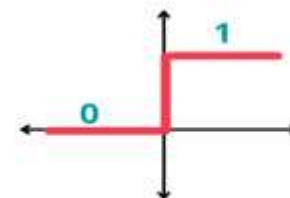
- 1) Binary Step Function: Binary step function is one of the simplest activation functions. The function produces **binary output** and thus the name *binary step function*. This activation function can be used for **binary classifications**.



- It produces **binary output as 0 or 1**. Any value below 0 is automatically given a value of 0 and **any value above 0 is given a value of 1**.
- If the **input is greater than zero**, **turn/send a pulse to activate (ON)**, **otherwise (OFF)**.
- Binary Step Function depends on the **Threshold Value**, that decides **whether the neuron should be active or not**.

Binary Step Function

$$f(x) = \begin{cases} 1 & \text{if } x > \theta \\ 0 & \text{if } x < \theta \end{cases}$$




Binary Step Activation Function



- Limitations: It cannot provide Multi Value Outputs.
- i.e, it cannot be used in Multi Class Classification Problems.



- 2. Linear Activation Function: The linear activation function, also known as "no activation," or "**identity function**", is one of the most straightforward activation functions, where the **output is identical to the input**.
- **Equation** : Linear function has the equation similar to as of a straight line i.e. **$y = x$**
- **Range** : -inf to +inf
- The linear activation function typically **adds the weighted sum with bias and passes it as output.** 

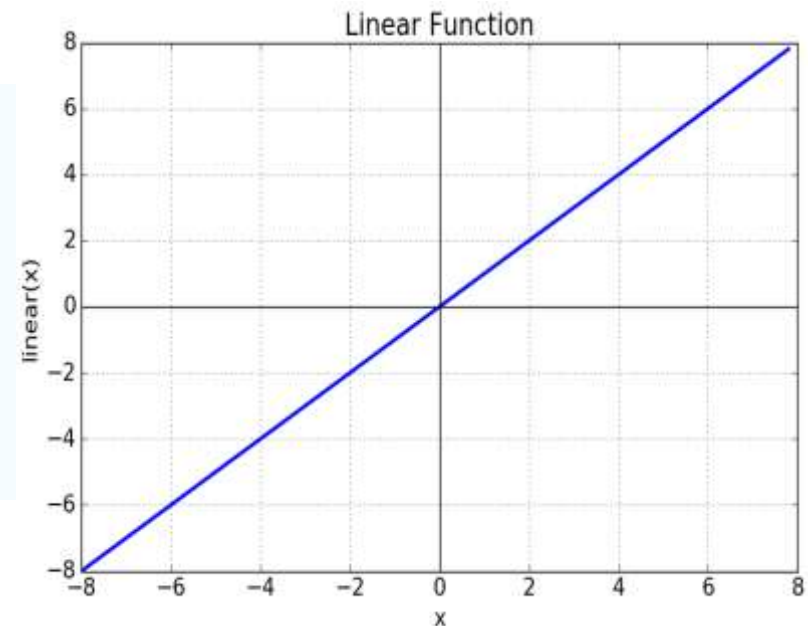
- Limitations: It cannot provide Multi Value Outputs.
- i.e, it cannot be used in **Multi Class Classification Problems.**

Mathematically, it can be defined as:

$$f(x) = x$$

where :

$$x = \sum Input * Weight + Bias(b)$$



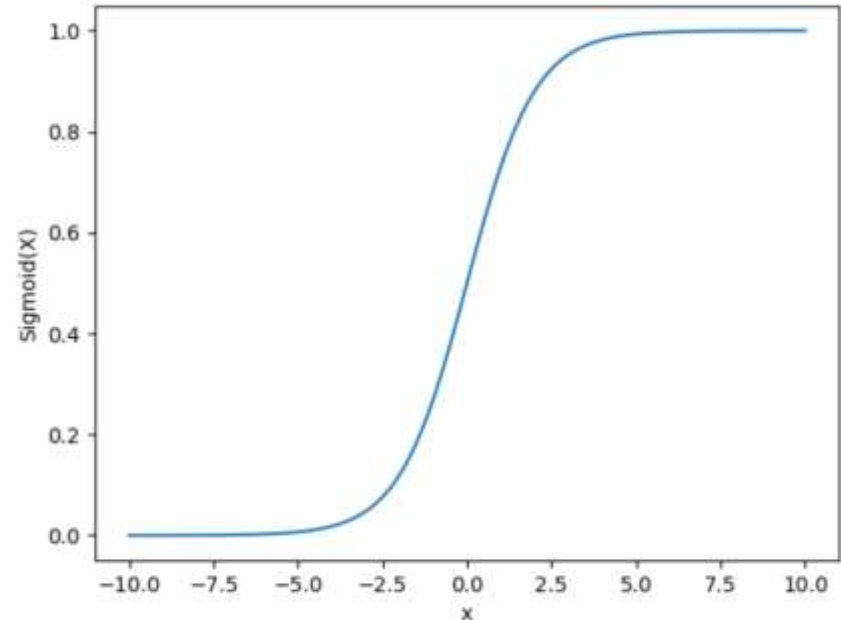
- 3. Non Linear Activation Function: A nonlinear function used to **send the output signal either on or off** to **a neuron** is known as an **activation function**.
- There are different types of Non Linear Activation Functions:
 - (a) Sigmoid Activation Function
 - (b) Tanh Activation Function
 - (c) ReLU Activation Function
 - Leaky ReLu Activation Function
 - (d) Softmax Activation Function



- A) Sigmoid Activation Function: Introduced in the early 1990s, This function is a probabilistic approach towards decision making and output range is between 0 and 1.
- Since the **range is minimum** , **predictions (decisions)** would be **more accurate**.
- The Sigmoid function, often represented as **(x)**, features an **S-shaped curve**.



$$f(x) = \frac{1}{1 + e^{-x}}$$



Use Cases: Sigmoid finds the **binary classification problems**, like **logistic regression**, where **outputs represent probabilities**. Its prevalence extends to the output layer of neural networks handling tasks such as **spam detection in emails**, where the requirement is to output a probability score.



The sigmoid function, also known as the **squashing function**, takes the **input from the previously hidden layer** and **squeezes it between 0 and 1**. Such as "yes" or "no", or 1 or 0. The sigmoid function can **output a probability between 0 and 1**, which can be used to predict the likelihood of a particular class.

Applications:

1. **Binary Classification Problems:**

We can use the sigmoid function in binary classification problems as it returns the **output between 0 and 1**.



2. Image Datasets and Neural Networks:

The sigmoid function can be used for **neural networks** on **image datasets** for performing tasks like **image segmentations, classifications, etc.**

Neural network architectures and components like **activation functions** enable **modeling of complex non-linear patterns** across domains like:


Image recognition

Natural language processing

Speech recognition

Reinforcement learning

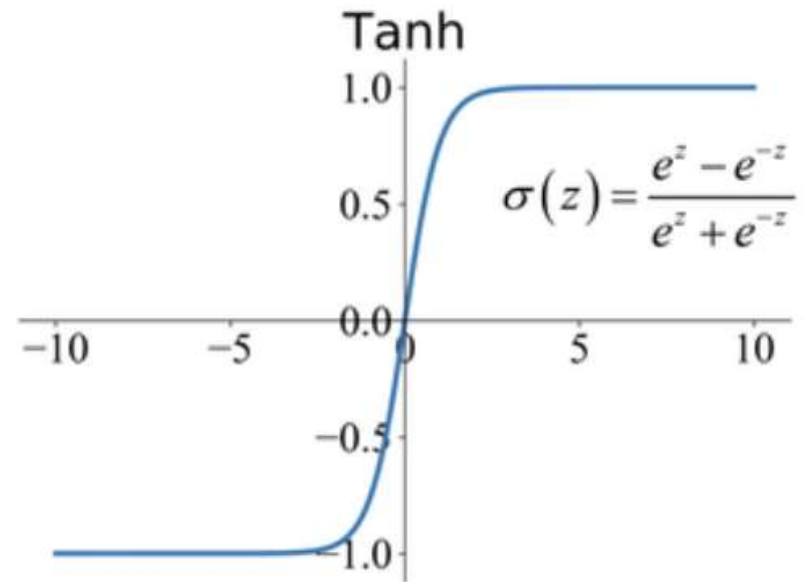


- **Drawback of the Sigmoid Function:** One of the significant issues with the sigmoid is the **lack of weight updating**. Therefore, the function sometimes returns **small values as outputs**, making **no changes in the weights and biases** that cause the **vanishing gradient problem**.
 - This problem arises in **really large Deep Neural Networks**.
 - This leads to **more computational costs and poor learning**.
- 

- B) Hyperbolic Tangent Activation Function (Tanh)
- Used widely in 1990's-2000's, it overcomes the disadvantage of the **sigmoid activation function** by *extending the range to include -1 to 1*.
- It is also having the same S-shape with the difference in **output range of -1 to 1**.
- In Tanh, the larger the input (more positive), the closer the **output value will be to 1.0**, whereas the **smaller the input (more negative)**, the closer the **output will be to -1.0**.

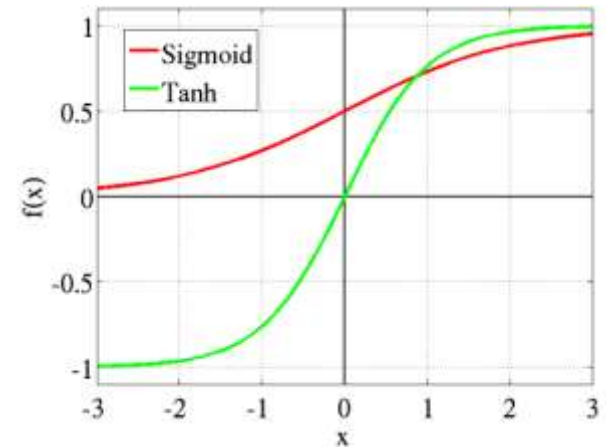
Tanh

$$f(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$$



But it did not solve the **vanishing gradient problem**.





Advantages of using this activation function are:

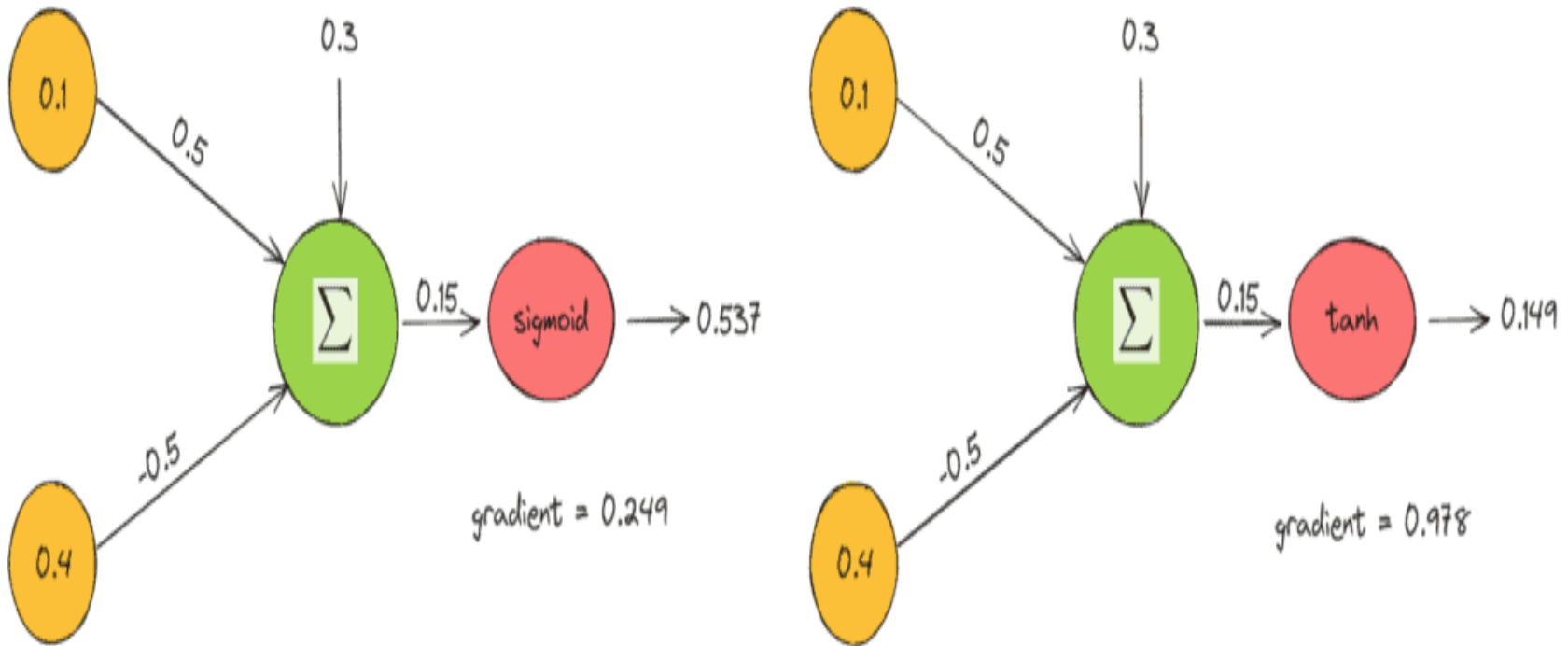
The output of the tanh activation function **is Zero centered**; hence we can easily map the output values as **strongly negative, neutral, or strongly positive**.

Usually **used in hidden layers of a neural network** as its values lie **between -1 to**; therefore, the mean for the hidden layer comes out to be 0 or very close to it. It helps in **centering the data and makes learning for the next layer much easier**.

○ Comparision of Sigmoid and Tanh Activation Function:

- Finally, we'll present an example of applying these activation functions in a **simple neuron of two input features and weights** . Therefore, we can see the **output value and the gradient** when we use the **sigmoid (left) and the tanh (right) activation function**:





The above example verifies our previous comments.

Furthermore, the output value of tanh is closer to zero, and the gradient is four times greater.

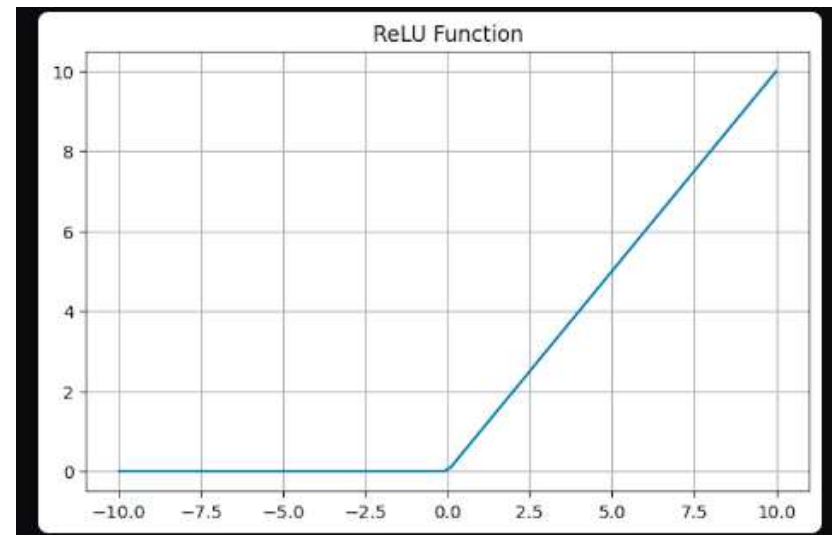


- Tanh function can be used when the input data has a range between **negative and positive values**.
 - For instance, it can be **used in Neural Networks** which is made for **sentiment analysis**, it can be used to model the **sentiment of text data**, where **negative sentiment is represented by negative values**, **positive sentiment is represented by positive values**, and **Neutral can be represented by zero or close to zero**.




- C) ReLU Activation Function: **ReLU** is nothing but **Rectified Linear Unit**.
- ReLU is a piecewise linear function that results in the input directly if it is positive. Otherwise, it **outputs zero**.
- **Equation** :- It gives an output x if x is positive and 0 otherwise.

$$f(x) = \max(0, x)$$

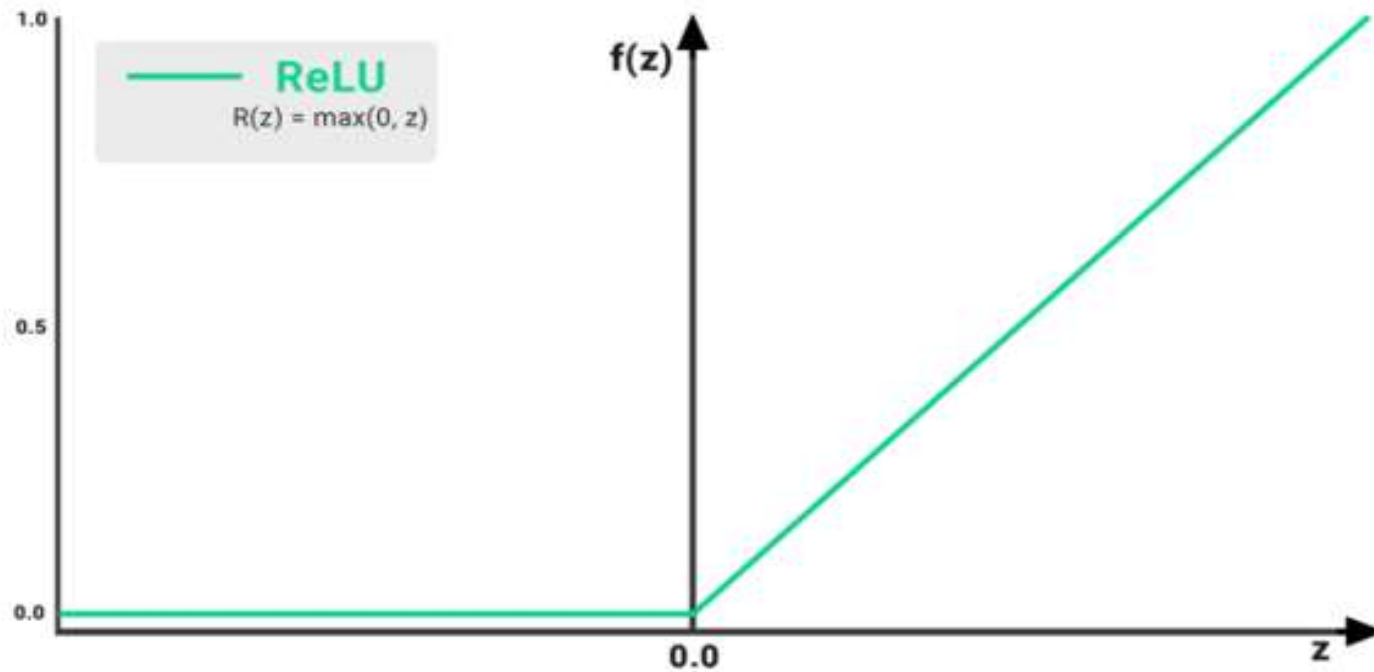


- ReLU is most commonly used in **Large Deep Neural Networks** with **lots and lots of hidden layers**. Since ReLU only **requires some threshold operations**, it is computationally **efficient** even though the **dataset and the network are large**.
- Also, ReLU is a great choice to **avoid the Vanishing Gradient Problem**
- This is the most frequently used activation unit in deep learning. $R(x) = \max(0, x)$. Thereby, if $x < 0$, $R(x) = 0$ and if $x \geq 0$, $R(x) = x$.

- Use Cases: Many types of neural networks **use ReLU as their default activation function**
- Its notable efficiency makes it especially advantageous in **Convolutional Neural Networks (CNNs)** and other deep learning models, leading to its **adoption in advanced object detection frameworks such as YOLO** (You Only Look Once) **etc.**
- Nature :- It is **non-linear**, which means we can easily **back propagate the errors** and have **multiple layers of neurons** being activated by the **ReLU function**. 

- ReLU is used as a **middle-layer (either convolution or dense) activation function** because it is a **non-linearity that works well and is robust to the vanishing gradient problem (as opposed to tanh or sigmoid)**.
- The main catch here is that the **ReLU function does not activate all the neurons at the same time.**
- This means that the **neurons will only be deactivated** if the **output of the linear transformation is less than 0.**






Function	Equation	Range	Derivative
ReLu (Rectified Linear Unit)	$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$0, +\infty$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$

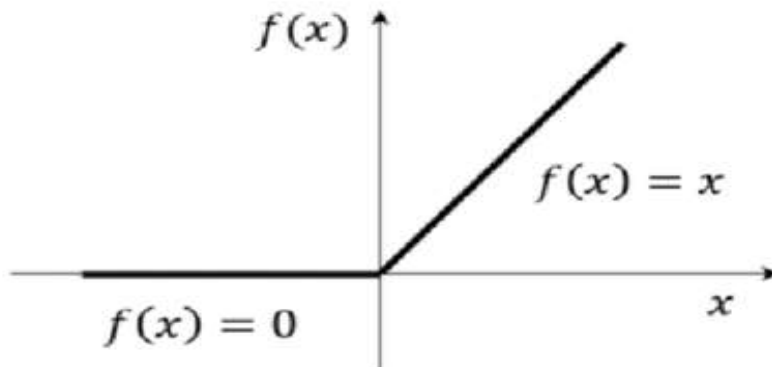
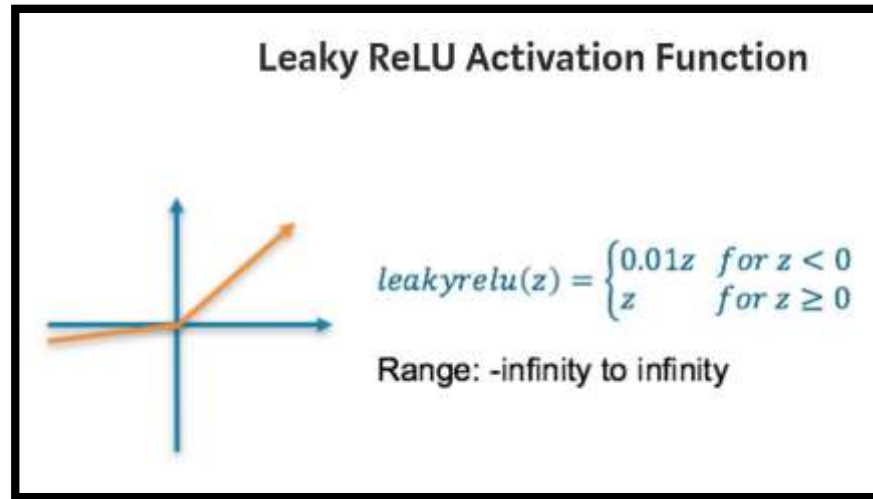


- For the **negative input values**, the **result is zero**, that means the **neuron does not get activated**. Since only a certain number of neurons are activated, the ReLU function is far more computationally efficient when **compared to the sigmoid and tanh function**.

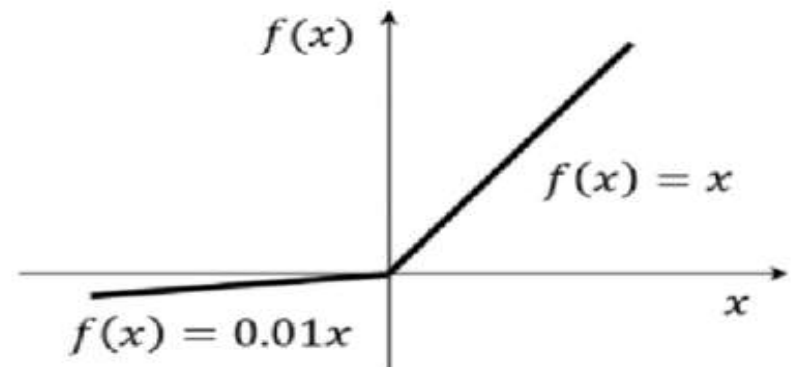


- i) Leaky ReLU Activation Function: Leaky ReLU function is nothing but an improved version of the ReLU function. As we saw that for the ReLU function, the gradient is 0 for $x < 0$, which would deactivate the neurons in that region.
 - Leaky ReLU is defined to address this problem. Instead of defining the Relu function as 0 for negative values of x , we define it as an extremely small linear component of x .
- 

- Here is the **mathematical expression**-



ReLU activation function



LeakyReLU activation function

- Can you see the Leak?
- The leak helps to increase the range of the ReLU function. Usually, the value of x is 0.01 or so.
- Therefore the range of the Leaky ReLU is (-infinity to infinity).

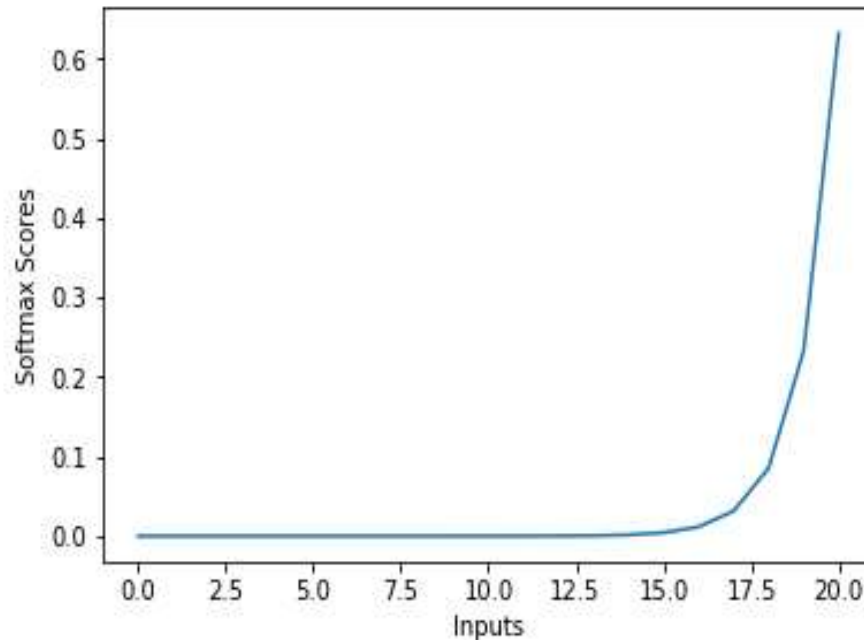


(D) Softmax Activation Function:

The softmax function is **more popular** for performing **multiclass classification tasks**.

- It simply takes **a vector of numbers as inputs** and produces **another vector containing the probability distribution of inputs**.
- For instance, given an input vector $x=[x_1x_2...x_n]$, the Softmax function computes a new vector $y=[y_1y_2...y_n]$ where each element **y_i** is given by:





The softmax function is **also a type of sigmoid function** but is handy when we are trying to **handle multi- class classification problems.**



Uses :- Usually used when trying to handle multiple classes.

- The softmax function was commonly found in the output layer of **image classification problems**.
- **Softmax function** is often described **as a combination of multiple sigmoid**. We know that **sigmoid returns values between 0 and 1**, which can be treated as probabilities of a data point belonging to **a particular class**.



Here is the mathematical expression of the same-

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}} \quad \text{for } j = 1, \dots, K.$$

- While building a **network for a multiclass problem**, the output layer would have as **many neurons as the number of classes in the target**.
- For instance if **you have three classes**, there would be **three neurons in the output layer**. Suppose you **got the output from the neurons as [1.2 , 0.9 , 0.75]**.

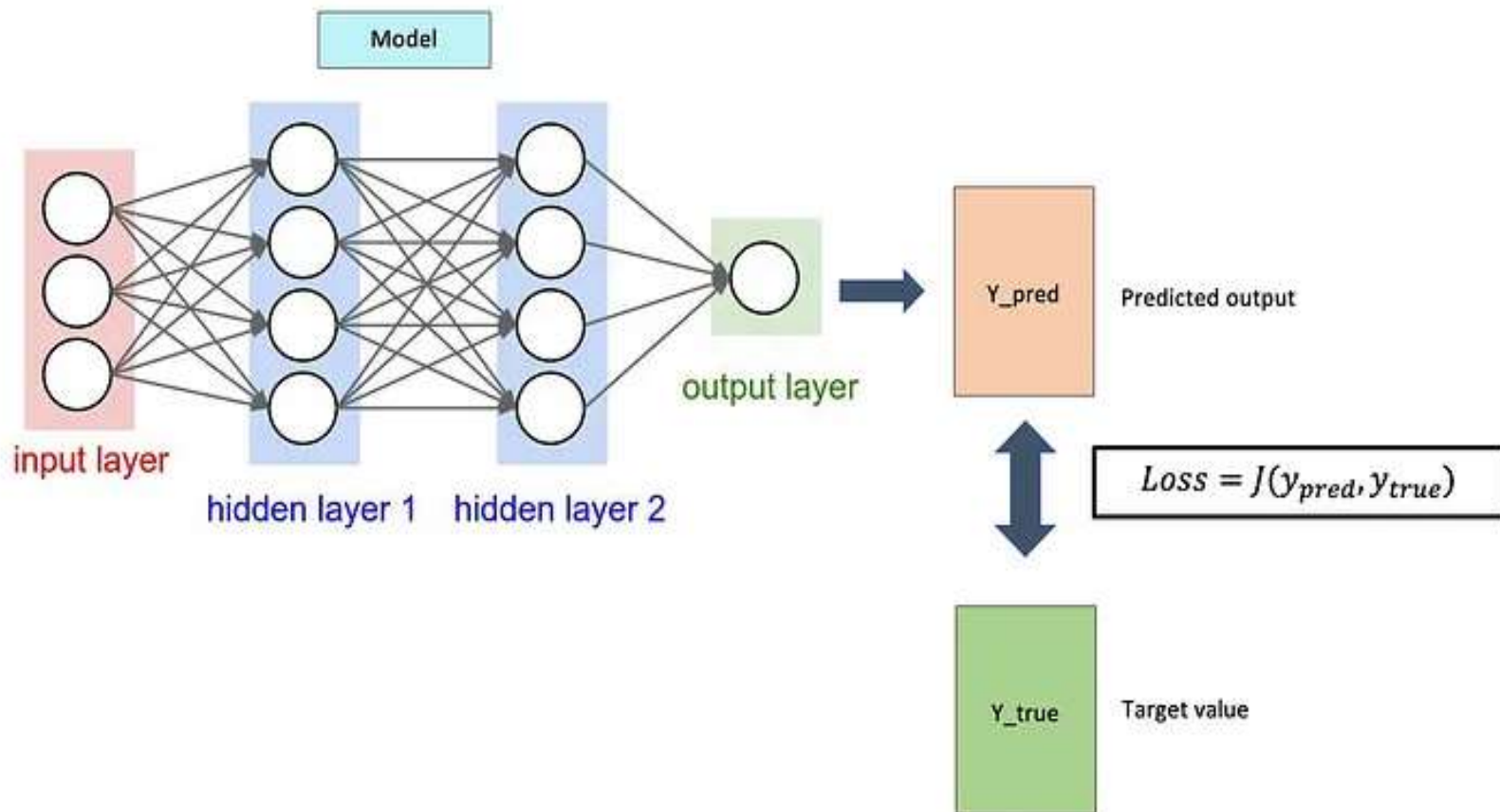
- Applying the **softmax function** over these values, you will get the following result - **[0.42 , 0.31, 0.27]**. These represent the probability for the **data point belonging to each class**.
- Note that the **sum of all the values is 1**.
- . For instance, if you want to **classify images of dogs, cats, and, rabbits**, **Softmax is the best option**.



4. LOSS FUNCTION

- **Loss function** is a method of evaluating “how well your algorithm is modeling in your dataset”. And also you can use various method to overcome the issue of losses w.r.t your actual output.
- A **loss function(or) Error Function** is a **mathematical function** that measures the difference between the predicted output of the model and the actual output.
- It **quantifies** how well the model is performing on a single training example.
- The **goal** of the deep learning model is to **minimize this loss function**, indicating a better fit of the model to the data.

- Let's define a **loss function (J)** that takes the following **two parameters**
 - Predicted output (y_pred)**
 - Target value (y_true)**



Types of Loss Functions:

1. Regression Loss Function

- a) Mean Square Error(MSE)
- b) Mean Absolute Error(MAE)
- c) Root Mean Square Error(RMSE)

2. Classification Loss Function

- i) Binary Cross Entropy(Log Loss)
- ii) Categorical Cross Entropy



Regression Loss Functions: Regression models deals with **predicting a continuous value** .

For Eg: Given floor area, number of rooms, size of rooms, predict the price of the room. The loss function used in the regression problem is called “**Regression Loss Function**”.

1. Mean Squared Error/Squared loss/ L2 loss: The Mean Squared Error (MSE) is a **straightforward and widely used loss function**.

To calculate the MSE, you take **the difference between the actual value and the model prediction**, **square it**, and then **average it across the entire dataset**.

$$\text{MSE} = \frac{1}{N} \sum_i^N (Y_i - \hat{Y}_i)^2$$



Example:

	A	B	C	D	E
1	Sno.	Observed(O)	Expected(E)	(O-E)	(O-E)^2
2	1	1.36	1.6	-0.24	0.0576
3	2	1.49	3.3	-1.81	3.2761
4	3	2.82	1.6	1.22	1.4884
5	4	1.65	4.4	-2.75	7.5625
6	5	1.59	3.1	-1.51	2.2801
7					14.6647
8					
9		n=No of observations=	5		
10					
11				MSE=	2.93294

Applications: Time Series Analysis, Student Academic Performance,

2. Mean Absolute Error/ L1 loss Functions: The Mean Absolute Error (MAE) is another **simple loss function**. It calculates the **average absolute** difference between the **actual value** and the **model prediction** across the dataset.

The diagram illustrates the Mean Absolute Error (MAE) formula with the following components and annotations:

- Divide by the total number of data points:** Points to the $\frac{1}{n}$ term in the formula.
- Sum of:** Points to the summation symbol \sum .
- Actual output value:** Points to the y term inside the absolute value.
- Predicted output value:** Points to the \hat{y} term inside the absolute value.
- The absolute value of the residual:** Points to the entire absolute value expression $|y - \hat{y}|$.

$$MAE = \frac{1}{n} \sum |y - \hat{y}|$$

Example:

E12						
	A	B	C	D	E	F
1	Sno.	Observed(O)	Expected(E)	(O-E)	O-E	
2	1	1.36	1.6	-0.24	0.24	
3	2	1.49	3.3	-1.81	1.81	
4	3	2.82	1.6	1.22	1.22	
5	4	1.65	4.4	-2.75	2.75	
6	5	1.59	3.1	-1.51	1.51	
7					7.53	
8						
9		n=No of observations=		5		
10						
11				MAE=	1.506	
12						
13						

Case Studies:

1. A Cloth Retailer
2. A pharmaceutical company
3. An e-commerce platform



3. Root Mean Square Error(RSME): Root mean square error or **root mean square deviation** is one of the most commonly used **measures for evaluating the quality of predictions.**

$$RMSE = \sqrt{\frac{\sum_{i=1}^N (\text{Predicted}_i - \text{Actual}_i)^2}{N}}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n}}$$

Where **N** is the number of data points,
y(i) is the **i-th** measurement, and
ŷ(i) is its corresponding prediction.



2. Classification Loss Functions:

i) Binary Cross Entropy(Log Loss): It is one of the most **common loss functions** used for **binary classification problems**.

It's suitable when dealing with **probability outputs**. For each instance, it penalizes the model **proportionally to the distance between the predicted probability and the actual label**.

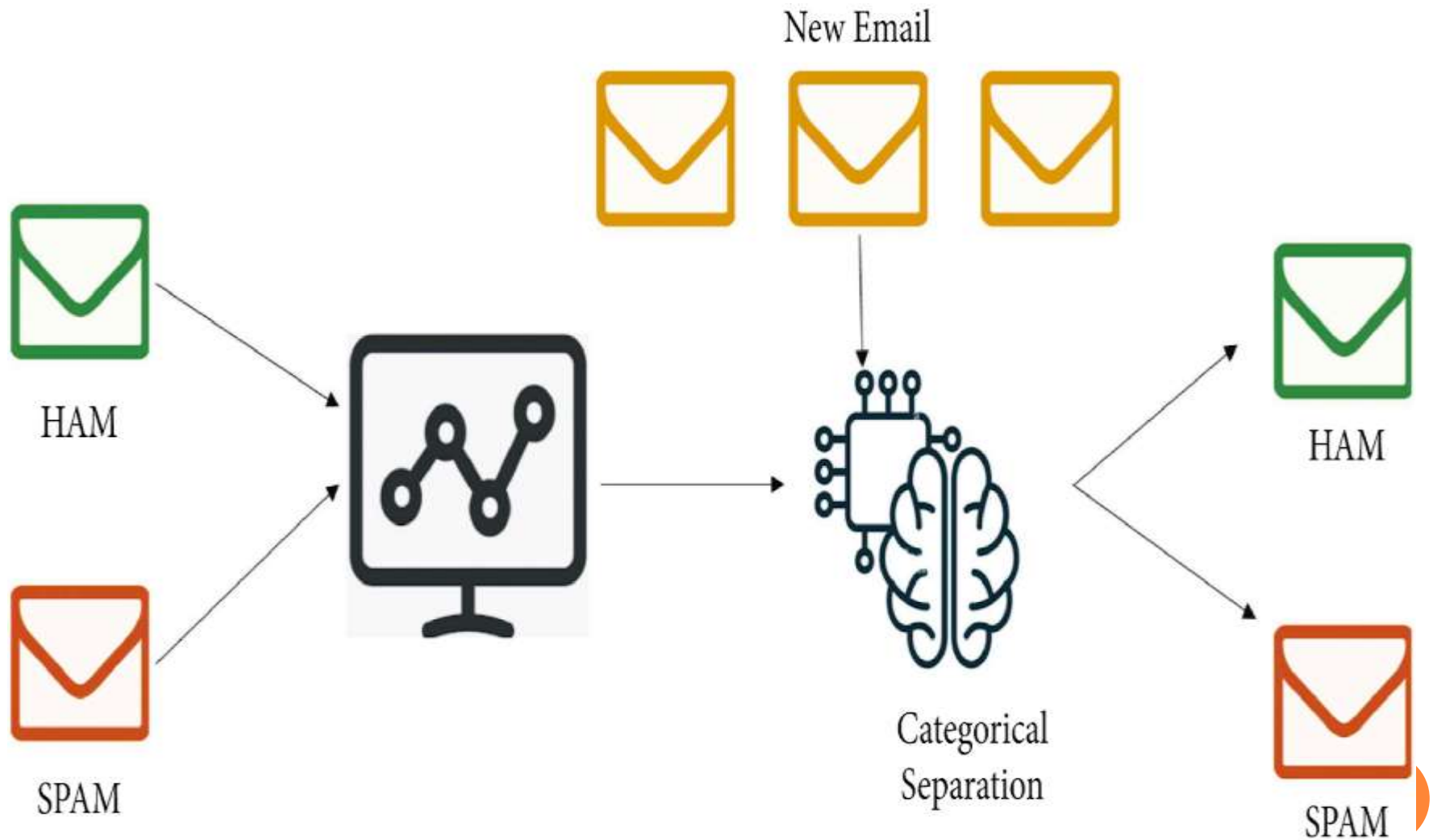


$$\text{Log Loss} = -\frac{1}{N} \sum_{i=1}^N y_i \log \hat{y}_i + (1-y_i) \log(1-\hat{y}_i)$$

- y_i – actual values
- \hat{y}_i – Neural Network prediction



Email Spam Detection (Binary Cross-Entropy)



ii) Categorical Cross Entropy: Categorical Cross entropy is used for **Multiclass classification and softmax regression.**

$$\text{Loss} = - \sum_{j=1}^K y_j \log(\hat{y}_j)$$

where k is number of classes in the data


where

k is classes,

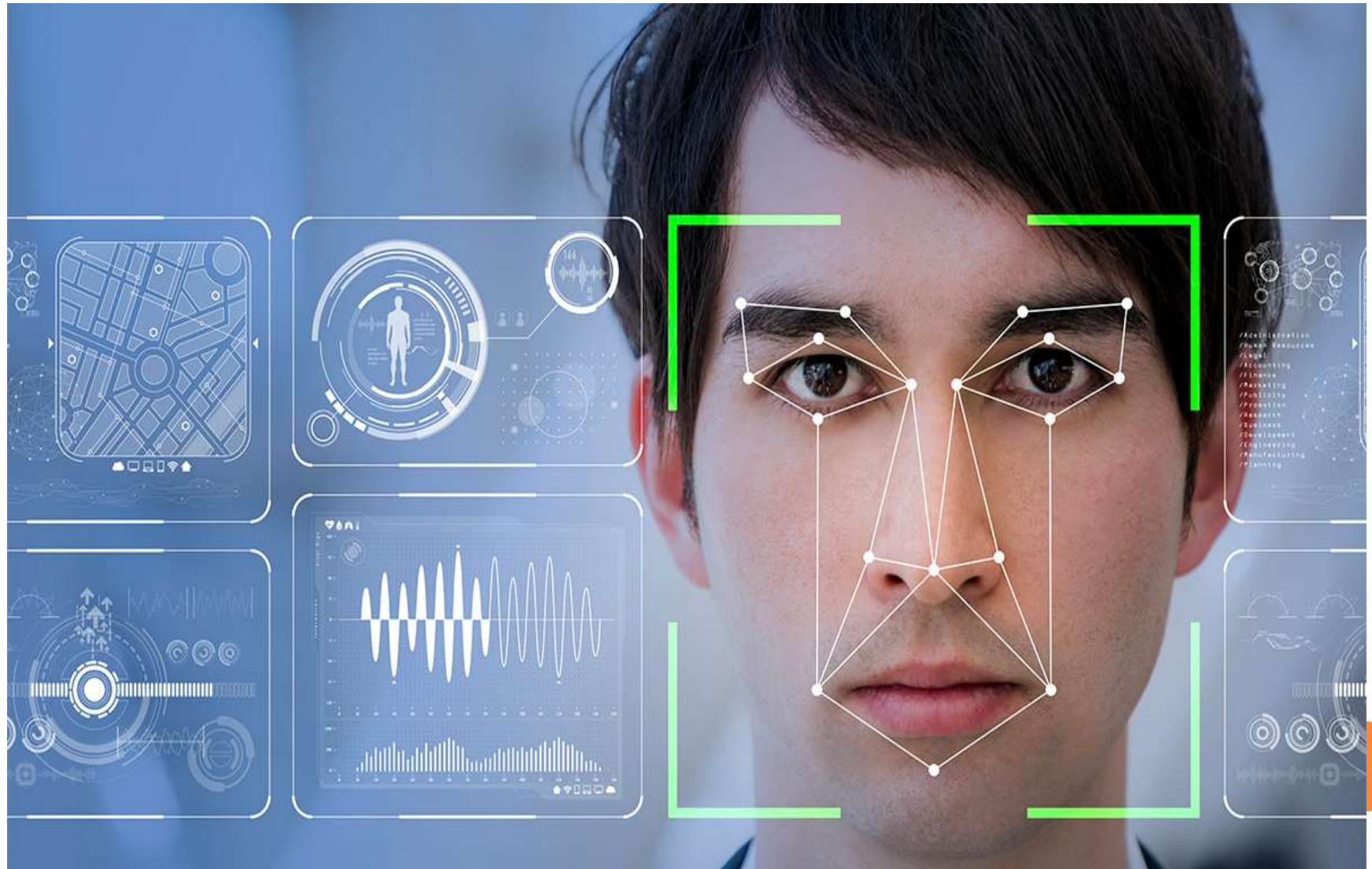
y = actual value

yhat – Neural Network prediction



- It finds applications in various domains, including **image classification**, **natural language processing**, **speech recognition**, **computer vision**, and **recommendation systems**.
 - In **image classification**, it helps the model assign the **correct label** to an input image.
 - While in **natural language processing**, it enables accurate **text classification or generation**.
 - For **speech recognition**, it aids in **transcribing spoken words**.
- 

Face Recognition:



Medical Image Diagnosis (Binary/Categorical Cross-Entropy)



5. HYPER PARAMETERS

- Hyperparameters are those **parameters** that are explicitly defined by the user to **control the learning process**.
- Hyperparameters determine **key features** such as **model architecture**, **learning rate**, and **model complexity**.
- Examples of hyperparameters include the **number of nodes** and **layers in a neural network** and the **number of branches in a decision tree**.



- Hyperparameters **control many aspects** of **DL algorithms**.
 - They can decide the **time and computational cost** of **running the algorithm**.
 - They can **define the structure of the neural network model**
 - They affect the **model's prediction accuracy**
- In other words, hyperparameters control the **behavior and structure of the neural network models**.

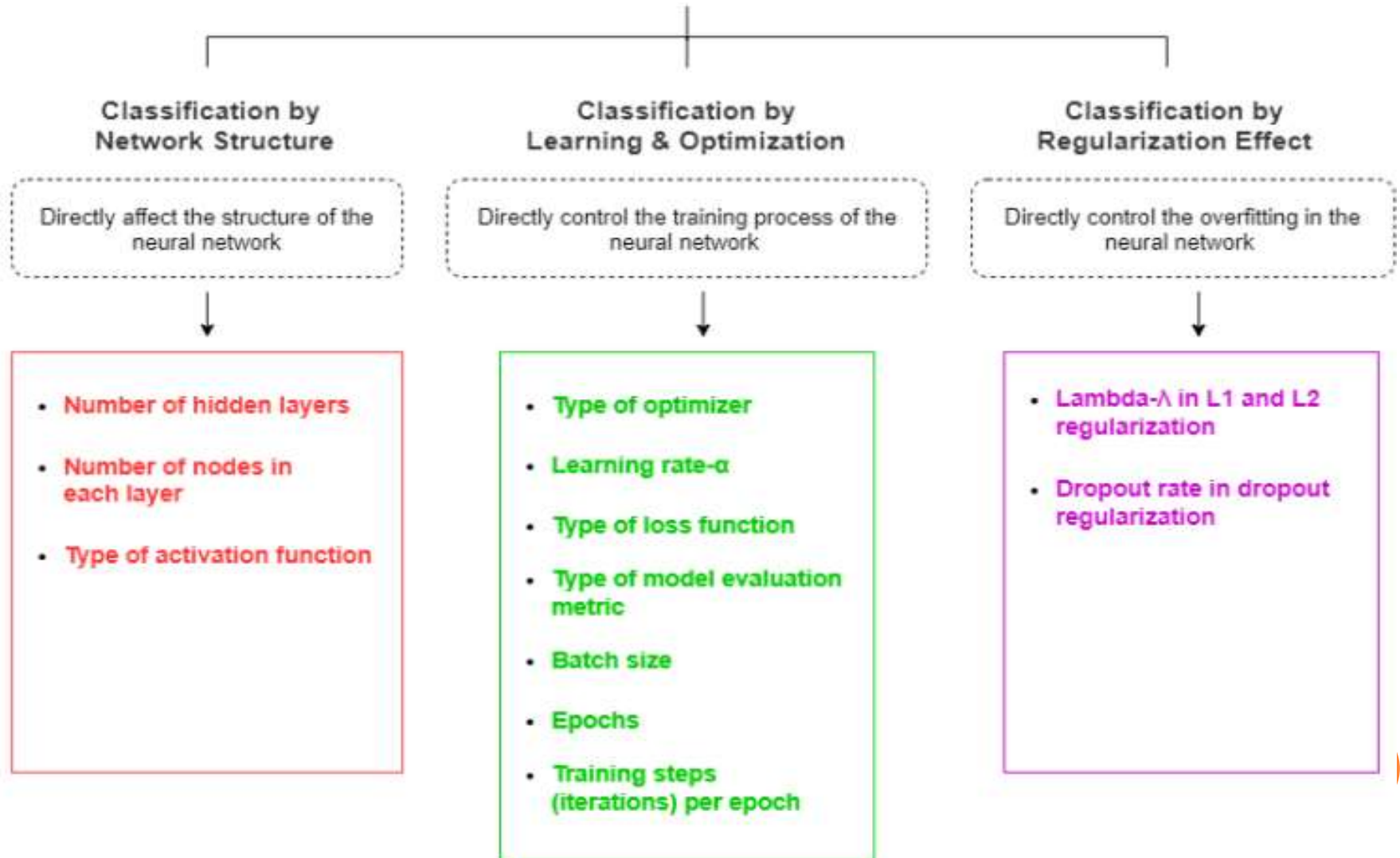


○ The Main Key points used in Hyperparameters are:

- Main parameters of the NN is **W and b**
- Hyper parameters (parameters that control the algorithm) are like:
 - **Learning rate.**
 - Number of **iterations.**
 - Number of **hidden layers L.**
 - Number of **hidden units n.**
 - Choice of *activation functions.*
 - Check the Training and Testing



Classification of Neural Network Hyperparameters



- Learning Rate: It is a Hyperparameter that provides the model a scale of how much model **weights should be updated, to minimize the Loss Function.**
- Optimizer: In deep learning, optimizers are crucial as algorithms that **dynamically adjust the model's parameters** to **minimize the loss function.**
- There are **various optimization techniques** to **change model weights and learning rates**, like AdaGrad, RMSProp, AdaDelta, and Adam.



➤ For eg :

➤ **Adagrad** stands for **Adaptive Gradient Optimizer**.

It is used to **reduce the loss function** with **respect to the weights**. The weight updating formula is as follows:

$$(w)_{\text{new}} = (w)_{\text{old}} - \eta \frac{\partial L}{\partial w(\text{old})}$$

➤ Based on **iterations**, this formula can be written as:

$$w_t = w_{t-1} - \eta \frac{\partial L}{\partial w(t-1)}$$

where

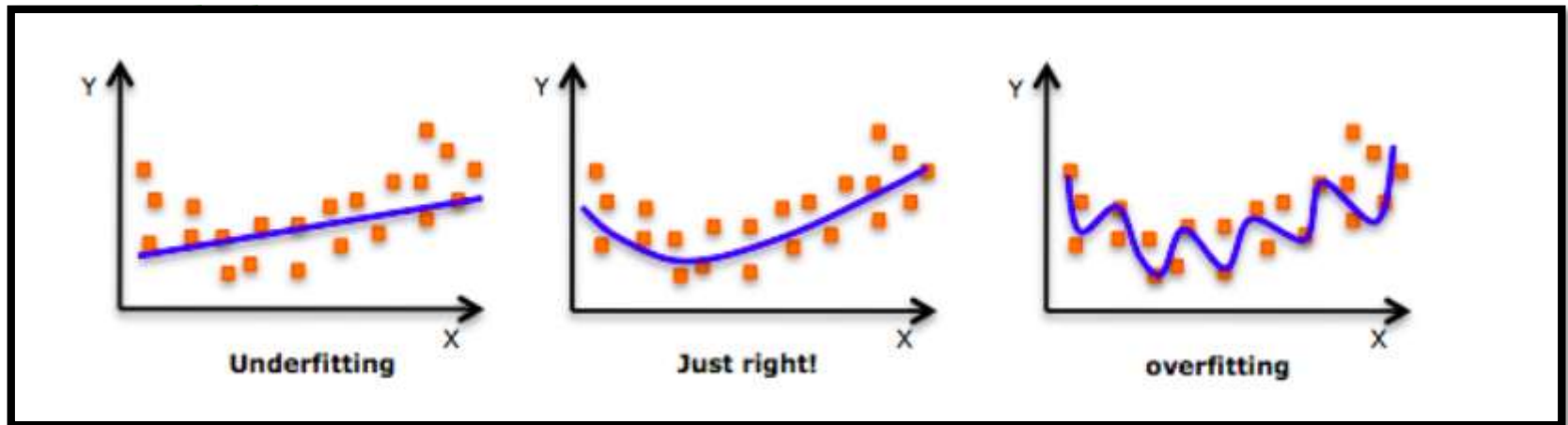
w(t) = value of w at **current iteration**,

w(t-1) = value of w at **previous iteration** and

η = **learning rate**.



- Regularization: Regularization is a set of methods for **reducing overfitting in Deep Learning**

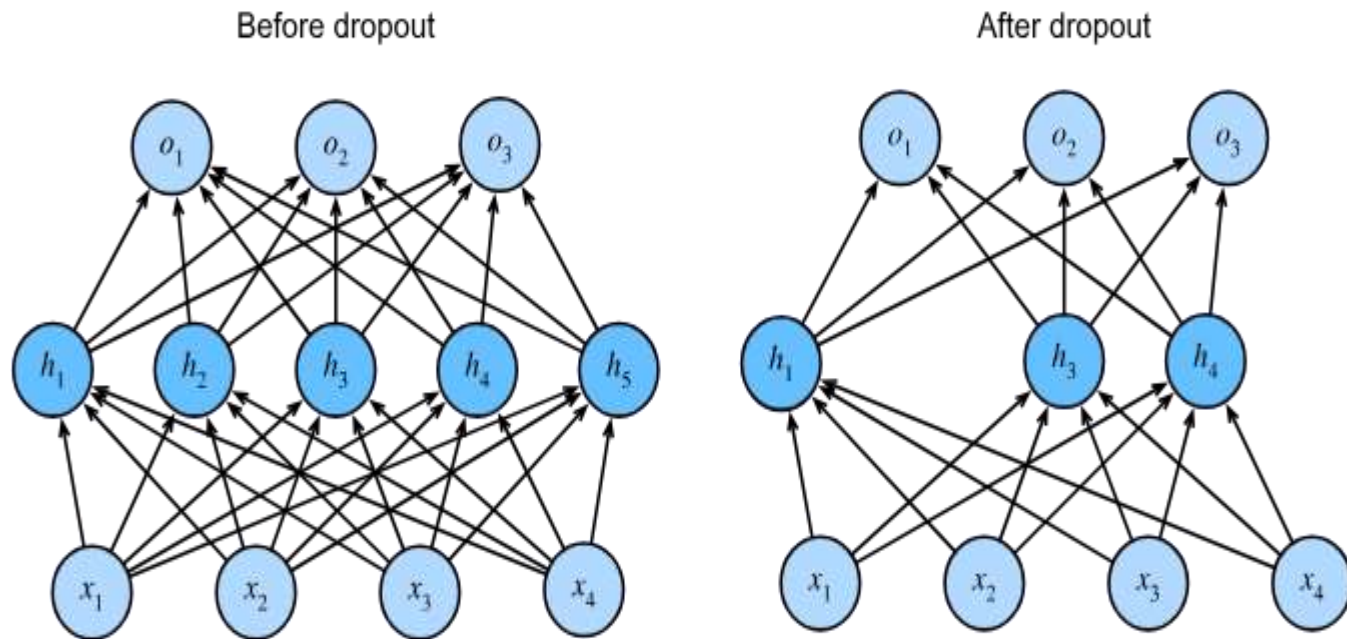


Just have a look at the above figure, and we can immediately predict that once we try to **cover every minutest feature of the input data**, there can be **irregularities in the extracted features**, which can **introduce noise in the output**. This is referred to as "Overfitting".

- This may also happen with the **lesser number of features extracted** as some of the **important details might be missed out**. This will leave an effect on the accuracy of the **outputs produced**. This is referred to as **"Underfitting"**.
- To eliminate this, **regularization is used**, in which we have to make the **slightest modification in the design of the neural network**, and we can **get better outcomes**.



- **Dropout**: Dropout was introduced by "Hinton et al" and this method is now very popular. It consists of setting to zero the output of each hidden neuron in chosen layer with some probability and is proven to be very effective in reducing overfitting.



- To achieve **dropout regularization**, some **neurons in the artificial neural network** are randomly disabled. That prevents them from being too dependent on one another as they learn the correlations.
- Dropout is driven by **randomly dropping a neuron** so that it will **not contribute** to the **forward pass and back-propagation**.





THANK YOU!