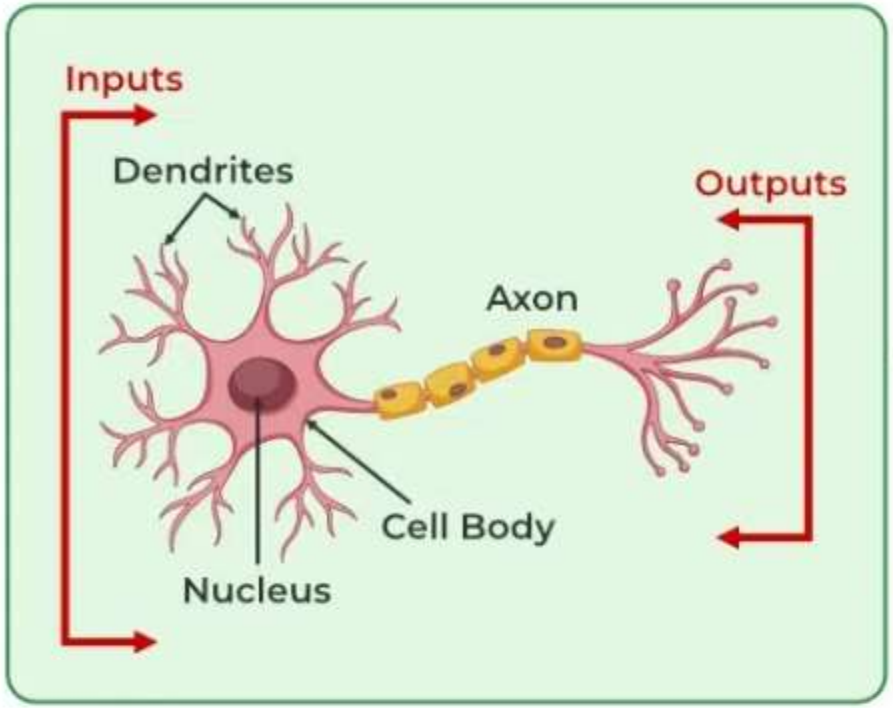
Introduction and History of Artificial Neural Networks

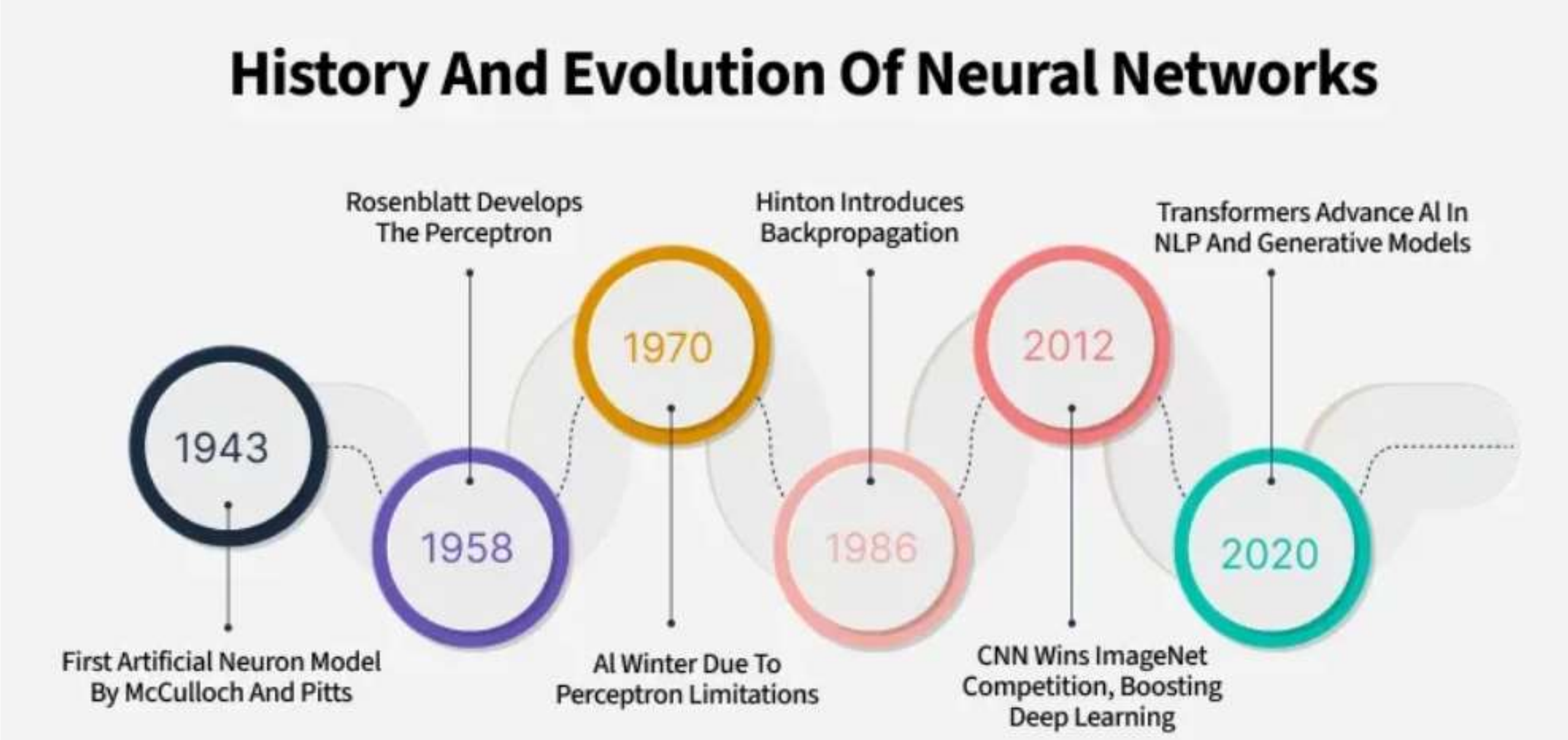
CT-466 | Week 1 - Lecture 1

Instructor: Mehar Fatima Shaikh

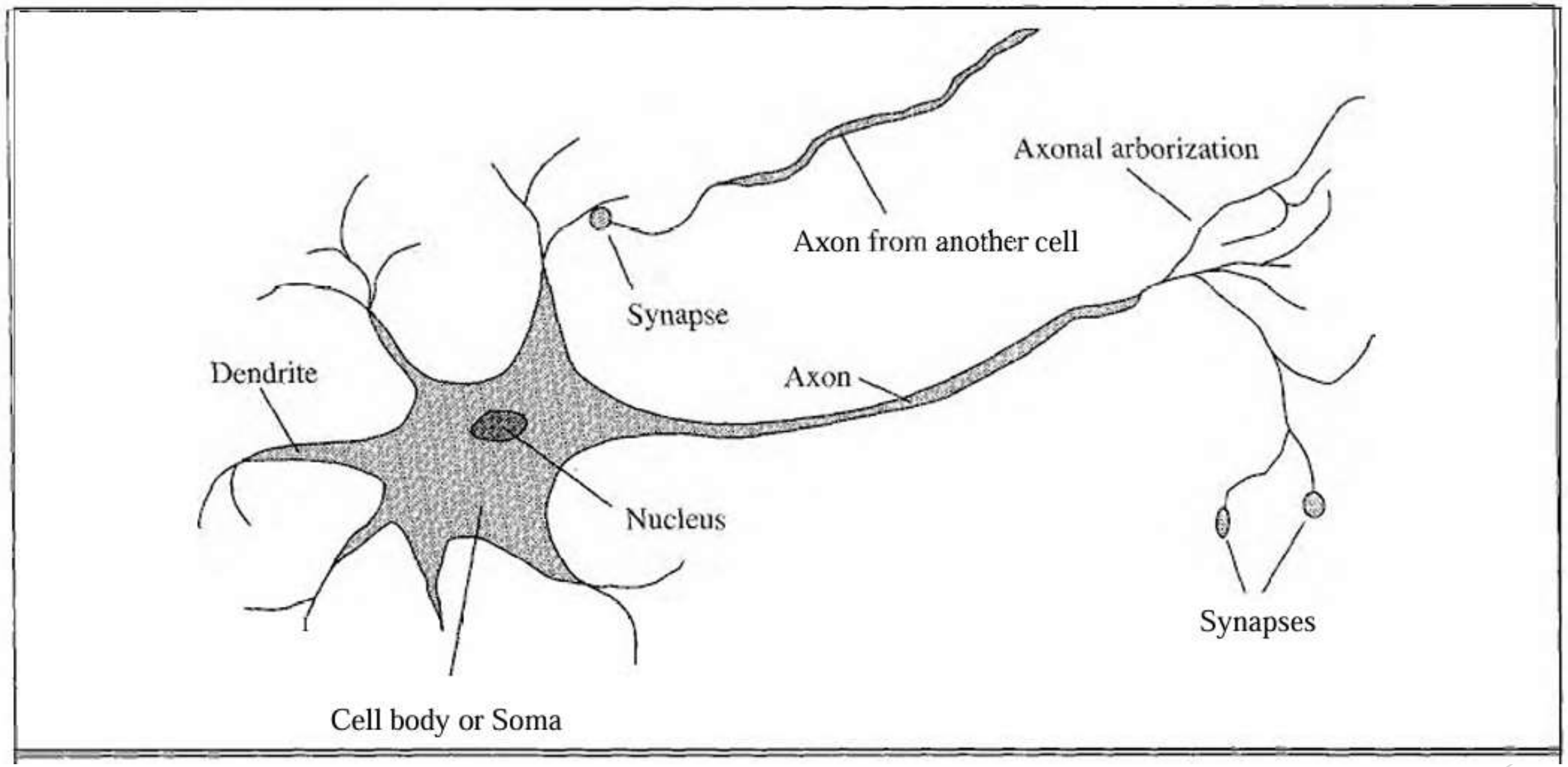
What is an Artificial Neural Network (ANN)?

- A computational model inspired by the way biological neural networks work
- Consists of layers of interconnected 'neurons'
- Used for tasks like classification, regression, and pattern recognition




History And Evolution Of Neural Networks
Rosenblatt Develops The Perceptron
Hinton Introduces Backpropagation
Transformers Advance AI In NLP And Generative Models
1943
1958
1970
1986
2012
2020
First Artificial Neuron Model By McCulloch And Pitts
AI Winter Due To Perceptron Limitations
CNN Wins ImageNet Competition, Boosting Deep Learning

Biological Neural Network



Biological Neural Network

The parts of a nerve cell or neuron. Each neuron consists of a cell body, or soma, that contains a cell nucleus. Branching out from the cell body are a number of fibers called dendrites and a single long fiber called the axon. The axon stretches out for a long distance, much longer than the scale in this diagram indicates. Typically, they are 1 cm long (100 times the diameter of the cell body), but can reach up to 1 meter. A neuron makes connections with 10 to 100,000 other neurons at junctions called synapses. Signals are propagated from neuron to neuron by a complicated electrochemical reaction. The signals control brain activity in the short term, and also enable long-term changes in the position and connectivity of neurons. These mechanisms are thought to form the basis for learning in the brain. Most information processing goes on in the cerebral cortex, the outer layer of the brain. The basic organizational unit appears to be a column of tissue about 0.5 mm in diameter, extending the full depth of the cortex, which is about 4 mm in humans. A column contains about 20,000 neurons.

Biological Neural Network

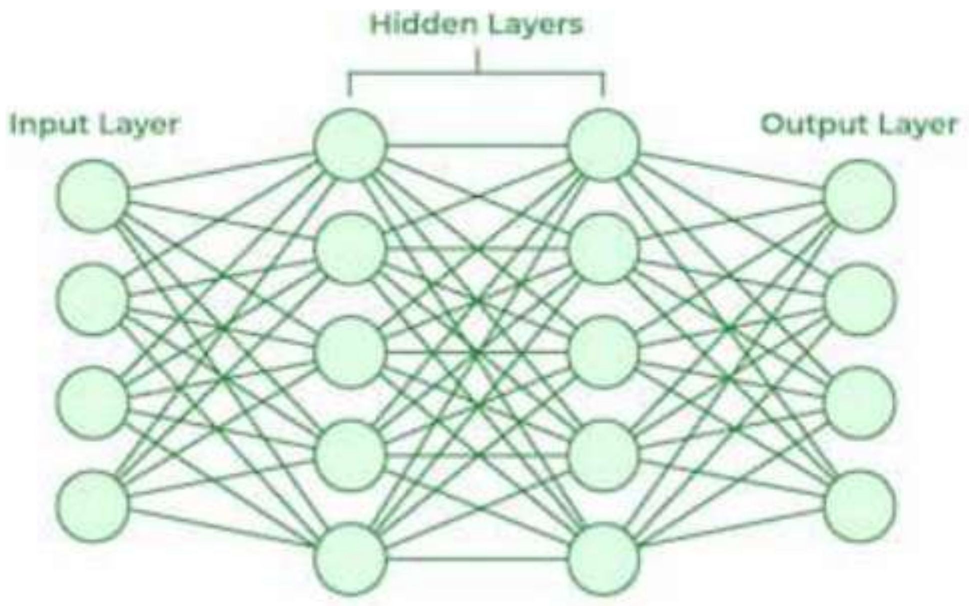
- Dendrites → Branch-like structures that receive signals from other neurons.
- Cell Body (Soma) → The main part of the neuron that processes incoming signals and contains the nucleus.
- Axon → A long fiber that carries electrical signals away from the cell body to other neurons or muscles.
- Synapses → The small junctions (gaps) where signals are passed from one neuron's axon to another neuron's dendrites or cell body.

Biological vs Artificial Neurons

- Biological neurons: Brain cells that transmit signals
- Artificial neurons: Mathematical functions that mimic behavior
- ANN simulates learning from data just like brain adapts to experiences

Basic Structure of an ANN

- Input Layer: Takes input features
- Hidden Layer(s): Process information with weighted connections
- Output Layer: Provides final prediction

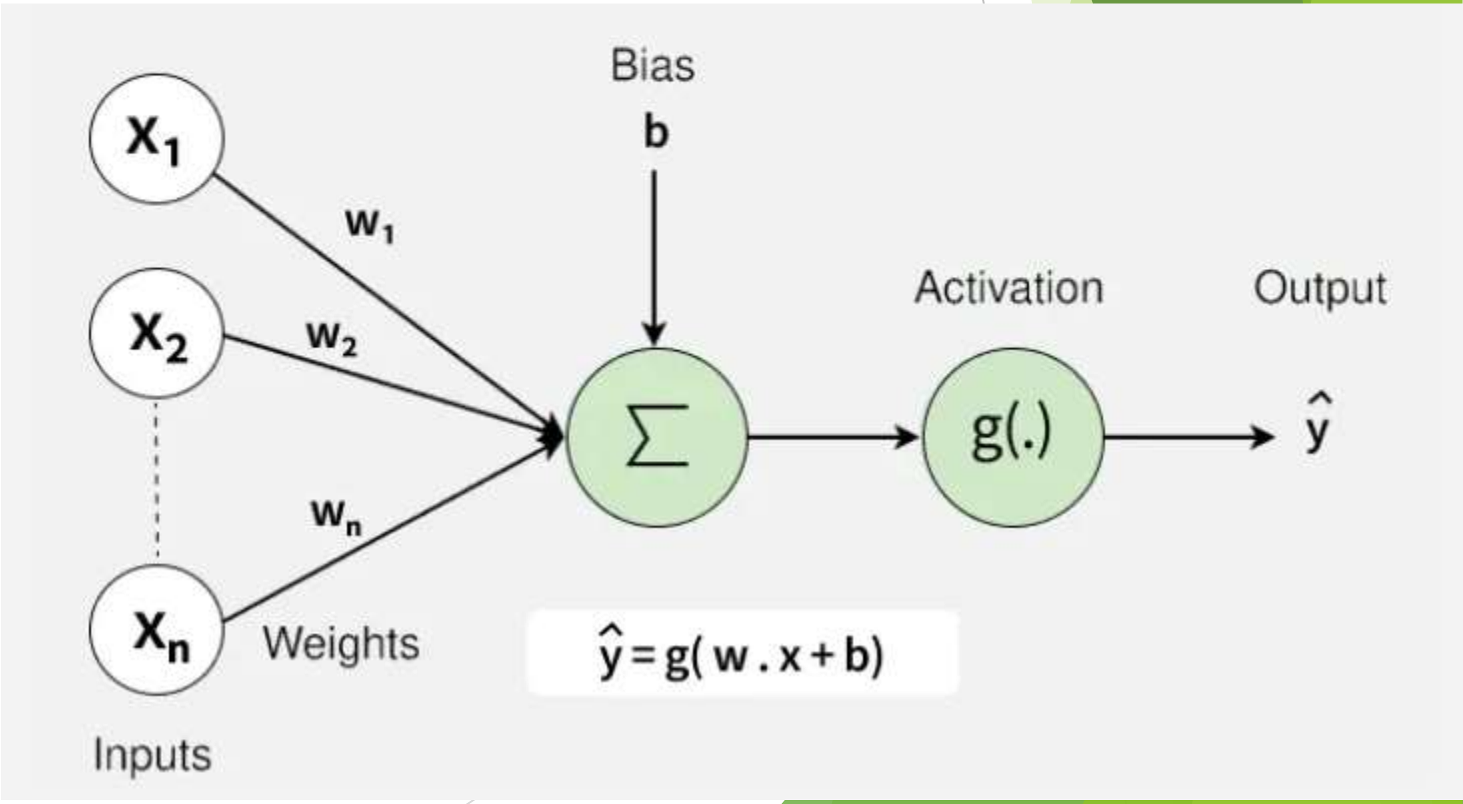


Applications of ANN

- Image Recognition
- Speech Processing
- Predictive Analytics
- Fraud Detection
- Robotics and Control Systems

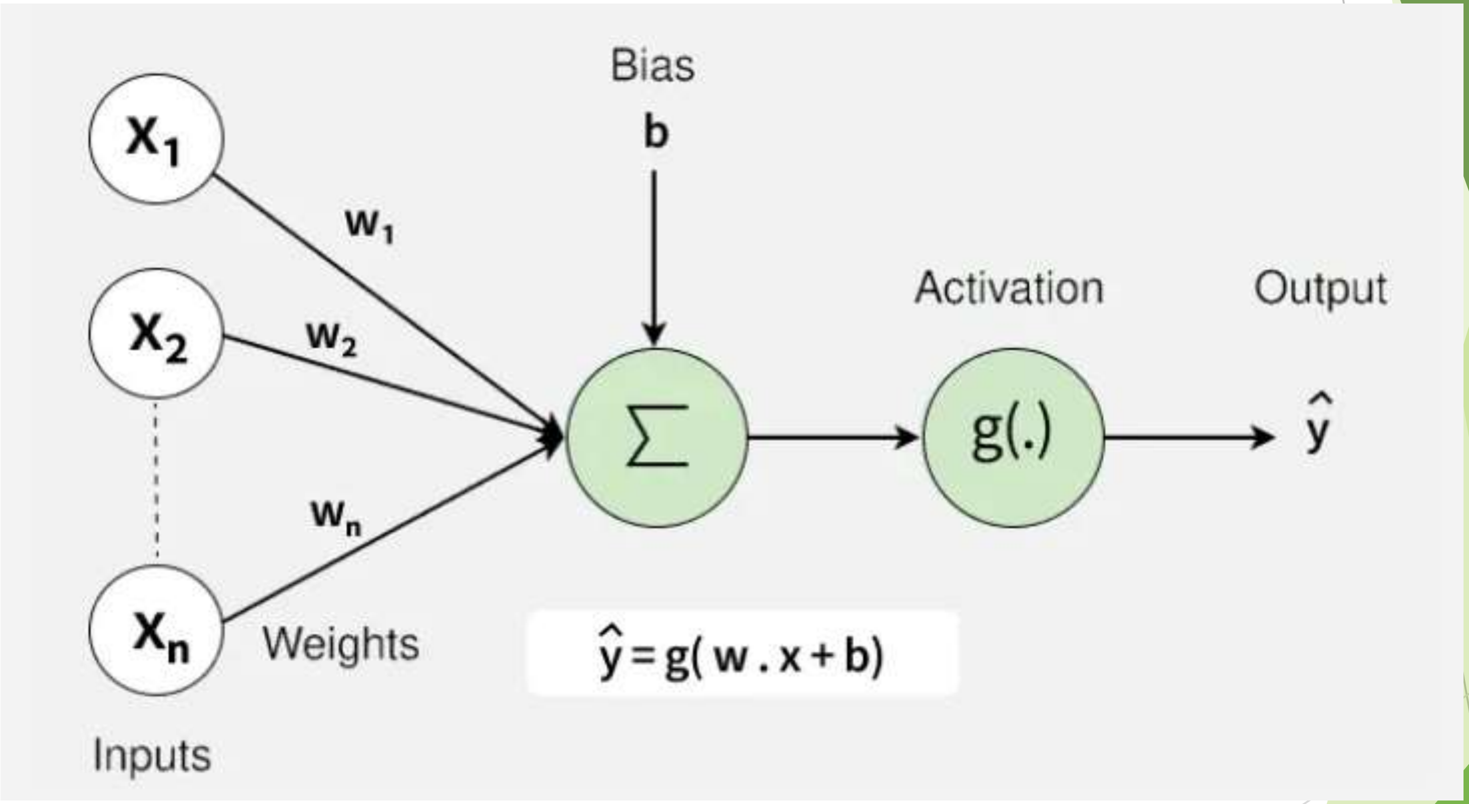
How ANN Works

1. Inputs are passed to neurons
2. Each input is multiplied by weights
3. A bias is added
4. Activation function determines output
5. Result is passed to next layer



Basic units of ANN

- Neuron (Node)
- Weight
- Bias
- Activation Function
- Epoch, Batch
- Loss Function



Activation Functions

- Sigmoid: Smooth gradient, outputs between 0 and 1
- Tanh: Outputs between -1 and 1
- ReLU: Rectified Linear Unit, fast convergence

Python Code: Single Neuron

```
import numpy as np
def sigmoid(x):
    return 1 / (1 + np.exp(-x))

inputs = np.array([1.0, 2.0])
weights = np.array([0.5, -0.6])
bias = 0.0

output = sigmoid(np.dot(inputs, weights) + bias)
print('Neuron Output:', output)
```

Numerical Example 1

- Input = [1, 2], Weights = [0.5, 0.5], Bias = 0
- Output = sigmoid(1*0.5 + 2*0.5 + 0) = sigmoid(1.5) ≈ 0.8176

Numerical Example 2

- Input = [2, -1], Weights = [0.7, 0.3], Bias = -0.2
- Output = sigmoid(2*0.7 + (-1)*0.3 - 0.2) = sigmoid(0.9) ≈ 0.7109

Numerical Example 3

- Input = [0, 3], Weights = [0.2, 0.8], Bias = 0.1
- Output = sigmoid(0*0.2 + 3*0.8 + 0.1) = sigmoid(2.5) ≈ 0.9241

Learning in ANN

- Weights are updated based on error
- Loss Function calculates error
- Optimizer (e.g., Gradient Descent) updates weights

Limitations of ANN

- Requires large data and computing power
- Can overfit on small datasets
- Black-box nature: hard to interpret

Summary

- ANN is inspired by the brain and used in many fields
- Neurons apply weights, bias, and activation
- Simple models can be built using Python
- We will explore deeper architectures next