Day73_Deep_Learning_Introduction

August 26, 2025

1 Deep Learning – Introduction

Welcome to my Deep Learning documentation!
These notes are based on my classroom learning, with additional explanations, examples, and illustrations to make concepts clearer.

The goal is to:

- Build a solid understanding of the fundamentals of Deep Learning.
- Document each concept in a structured way for future reference.
- Share knowledge with peers and the community.

This notebook covers the basics of neural networks, forward & backward propagation, epochs, and activation functions.

In the upcoming days, I'll keep adding **ANN, CNN, RNN**, and other deep learning concepts with examples.

1.1 What is Deep Learning?

Deep Learning (DL) is a subset of **Machine Learning (ML)** where we use **neural network architectures** with multiple layers to automatically learn features from data. - ML algorithms require feature engineering (manually extracting features). - DL algorithms learn these features automatically from raw data (numbers, images, audio, text).

1.2 Why Deep Learning?

- Handles **complex data** like images, videos, speech, and text.
- Learns hierarchical features:
  - Example: In image recognition, first layers detect edges, next detect shapes, final layers detect objects.
- Powers modern applications like **speech recognition, translation, self-driving cars, and recommendation systems.**

Advantages:

- Works well with **large datasets.**
- Reduces the need for **manual feature engineering.**
- Excellent at handling **unstructured data** (image, video, audio, text).

Disadvantages:

- Requires **a lot of data.**
- Needs **high computational power (GPUs/TPUs).**
- Difficult to **interpret (black box models).**
- Training can take a **long time.**

2 Types of Neural Networks

Deep learning is built on neural networks. Depending on data type:

- **ANN (Artificial Neural Network)** → For structured numerical data.
- **CNN (Convolutional Neural Network)** → For images and videos.
- **RNN (Recurrent Neural Network)** → For sequential data like text or time series.

All three are **Neural Network Architectures (NNA).**

3 Neural Network Architecture (NNA)

Neural networks are inspired by the **Human Neural Network (HNN).**

Example: Mosquito bite

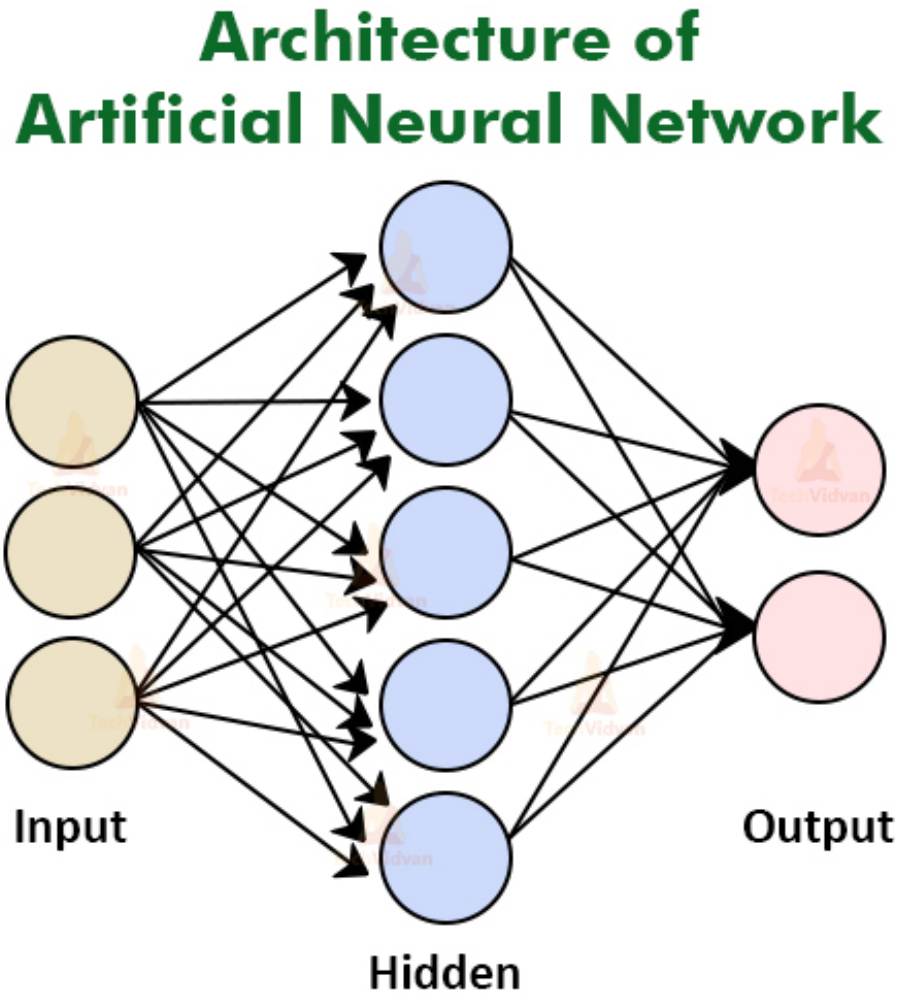
Imagine a mosquito bites your left hand:

- **Input layer:** Eyes/skin sense the mosquito.
- **Activation function:** Neurons activated.
- **Hidden layer:** Brain processes the signal.
- **Output layer:** Right hand moves to kill the mosquito.

This is similar to **ANN:**

- **Input Layer → Hidden Layer(s) → Output Layer.**

3.1 General structure:



4 Concepts We Learned Today

4.1 Neuron

- The basic unit of a neural network.
- Receives inputs, applies **weights**, passes through an **activation function**, and gives an output.

Mathematical representation:

$$y = f(w_1x_1 + w_2x_2 + ... + w_nx_n + b)$$

4.2 Neural Network

- **One neuron connected to another neuron** forms a **Neural Network**.

- The strength of the connection is represented by **weights**.

4.3 Perceptron

- The simplest form of a neuron.
- Makes decisions by combining inputs with weights.
- The **signal passing between neurons is called a synapse**.

4.4 Layers

1. **Input Layer (IP):**
   - Takes the raw features.
   - Example: `play (x1), study (x2), sleep (x3)`.
2. **Hidden Layer:**
   - Processes inputs with weights and activation functions.
   - Example:

$$h = x_1 w_1 + x_2 w_2 + x_3 w_3$$

3. **Output Layer (OP):**
   - Produces final prediction.

4.5 Forward Propagation (FP)

- Data flows from **Input → Hidden → Output**.
- We initially assign **random weights**.
- **Example:** Student's activities (play, study, sleep) are inputs → hidden layer computes → output predicts performance.

4.6 Backward Propagation (BP)

- If prediction is wrong, the model **goes back** and adjusts weights.
- The process of updating weights to reduce error is called **backpropagation**.
- Goal: Minimize the difference between **actual vs predicted output**.

4.7 Epoch

- **1 Epoch = 1 Forward Propagation (FP) + 1 Backward Propagation (BP)**.
- Multiple epochs are used to improve accuracy until the model converges.

4.8 Activation Functions

Activation functions decide whether a neuron should activate (fire) or not.

5 Activation Functions in Neural Networks

5.1 What is an Activation Function?

An **activation function** decides whether a neuron should "fire" (activate) or not.

- Without activation functions → Neural networks become just **linear functions** (like linear regression).
- With activation functions → Networks can learn **non-linear patterns** in data (like curves, images, language).

In simple words: Activation functions introduce **non-linearity**, which allows deep learning models to solve complex problems.

5.2 Why Do We Need Activation Functions?

1. **Non-linearity:** Real-world data is complex (e.g., image recognition, speech). Linear functions cannot model them.
2. **Feature learning:** Helps extract advanced features at different layers.
3. **Flexibility:** Different activation functions work better for different tasks (binary classification, multiclass, hidden layers).

6 Types of Activation Functions

6.1 Sigmoid Function

- Formula:

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

- Range: (0, 1).
- Smooth "S" shaped curve.

Where to use:
- **Binary classification problems** (output = probability).
- Always used in **output layer** when prediction must be probability between 0 and 1.

Example:
Predicting if an email is spam (1) or not spam (0).

6.2 Tanh (Hyperbolic Tangent) Function

- Formula:

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

- Range: (-1, 1).

- "S" shaped curve, but allows negative values.

Where to use:
- When the data can have both positive and negative outputs.
- Better than sigmoid in hidden layers because it is centered around 0.

Example:
Sentiment analysis (negative sentiment = -1, positive sentiment = +1).

6.3 ReLU (Rectified Linear Unit)

- Formula:

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

- Range: [0, ∞).

- Very fast and simple.

Where to use:
- Most common choice for **hidden layers** in deep learning.
- Works well for large datasets and deep networks.

Why:
- Solves the vanishing gradient problem (partially).
- Computationally efficient.

Example:
Image recognition (hidden layers in CNN).

6.4 Leaky ReLU

- Formula:

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{if } x \leq 0 \end{cases}$$

  where is a small value (e.g., 0.01).

- Range: (-∞, ∞).

- Allows small negative values.

Where to use:
- Hidden layers when we want to avoid "dead neurons" (ReLU problem where some neurons output 0 always).

Example:
Used in deep CNNs where many layers might cause neurons to die.

6.5 Softmax Function

- Formula:

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- Converts raw outputs (logits) into **probabilities**.

- Range: (0, 1), and all outputs add up to 1.

Where to use:
- Always used in the **output layer for multiclass classification**.
- Example: Predicting whether an image is a **cat, dog, or horse**.

6.6 Small Intuitive Examples

Example 1: Sigmoid (Binary Classification)

- Input: Student studies 5 hours.

- Output: Probability of passing = 0.85 → Student likely passes.

Example 2: Tanh

- Input: Movie review text.

- Output: -0.9 → strongly negative review.

Example 3: ReLU

- Input: Pixel value in an image = -20 → Output = 0.

- Input: Pixel value = 150 → Output = 150.

Example 4: Leaky ReLU

- Input: -20 → Output = -0.2 (small negative allowed).

Example 5: Softmax (Multiclass)

- Input Image: Could be Cat, Dog, or Horse.

- Output: [Cat=0.7, Dog=0.2, Horse=0.1].

- Prediction = Cat.

6.7 Summary of Activation Functions

Activation	Range	Where Used	Example Use Case
Sigmoid	(0, 1)	Output layer (binary classification)	Spam email detection
Tanh	(-1, 1)	Hidden layers / output	Sentiment analysis
ReLU	[0, ∞)	Hidden layers	Image recognition
Leaky ReLU	(-∞, ∞)	Hidden layers (avoid dead neurons)	Deep CNNs
Softmax	(0, 1) sum=1	Output layer (multiclass classification)	Cat vs Dog vs Horse

7 Summary

- Deep Learning is a collection of **neural network architectures (ANN, CNN, RNN)**.
- Neural networks are inspired by the **human brain**.
- Key concepts:
  - Neuron
  - Neural Network
  - Perceptron (Synapse)
  - Input Layer, Hidden Layer, Output Layer
  - Forward Propagation & Backward Propagation
  - Epochs
  - Activation Functions (Sigmoid, Tanh, ReLU, Leaky ReLU, Softmax)

8 Conclusion

In this notebook, we introduced the foundation of Deep Learning:

- What Deep Learning is and why it is used.
- Neural Network Architecture (Input, Hidden, Output layers).
- Key concepts: Neuron, Perceptron, Forward & Backward Propagation, Epochs.

- Activation Functions (Sigmoid, Tanh, ReLU, Leaky ReLU, Softmax).

Deep Learning is inspired by the human brain and forms the backbone of modern AI systems such as speech recognition, computer vision, and natural language processing.

This is just the beginning

In the next notes, I'll document more about **Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNN)** with real-world examples.

Thanks for reading!