Introduction to Deep Learning: Part I

June 23, 2025

0.1 What is Deep Learning?

Deep Learning (DL) is a subset of **Machine Learning (ML)** that enables computational models to **automatically learn features and patterns** from data. Unlike traditional ML techniques that **require manual feature engineering**, deep learning utilizes multiple processing layers to learn hierarchical data representations.

Over the past decade, deep learning has significantly advanced fields such as:

- **Speech Recognition** (e.g., Siri, Google Assistant)
- **Computer Vision** (e.g., image classification, object detection)
- **Natural Language Processing (NLP)** (e.g., ChatGPT, translation services)
- **Scientific Applications** (e.g., drug discovery, genomics, physics simulations)

0.1.1 Deep Learning vs. Machine Learning

Deep learning is a specialized branch within machine learning, but it differs in how it extracts features and processes data:

Features	**Machine Learning**	**Deep Learning**
Feature Extraction	Requires manual feature engineering	Learns features automatically from raw data
Scalability	Struggles with large datasets	Performs better with large datasets
Computation	Requires less computing power	Requires GPUs & HPC
Performance on Complex Tasks	Limited	Excels on image, speech, text

Table 1: Comparison of Machine Learning and Deep Learning

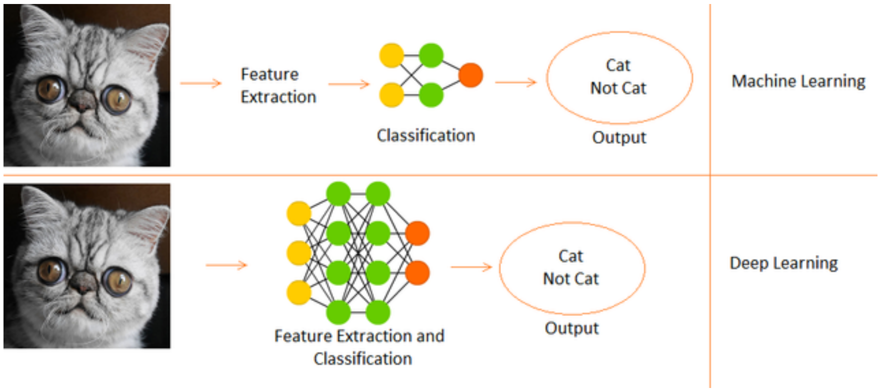

Figure 1: Key differences between Machine Learning and Deep Learning.

0.1.2 Advantages of Deep Learning

As datasets grow larger and more complex, traditional machine learning methods **struggle to extract meaningful patterns**. Deep learning scales effectively with data size and computational power, often leading to **higher accuracy and better generalization**. Key Advantages of Deep Learning include:

- Handles large-scale data efficiently
- Learns directly from raw data, eliminating manual feature engineering
- Excels in complex tasks (image recognition, speech, autonomous systems)

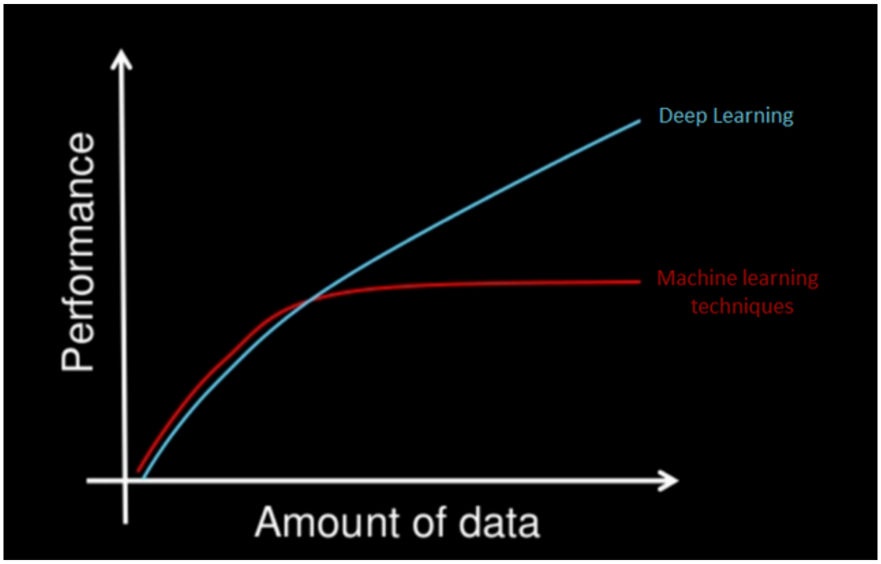

Figure 2: Performance comparison: Deep Learning vs. Machine Learning.

0.1.3 Types of Deep Learning Tasks

Deep learning models are commonly applied to four main learning paradigms:

- Supervised Learning
- Unsupervised Learning
- Semi-Supervised Learning
- Reinforcement Learning

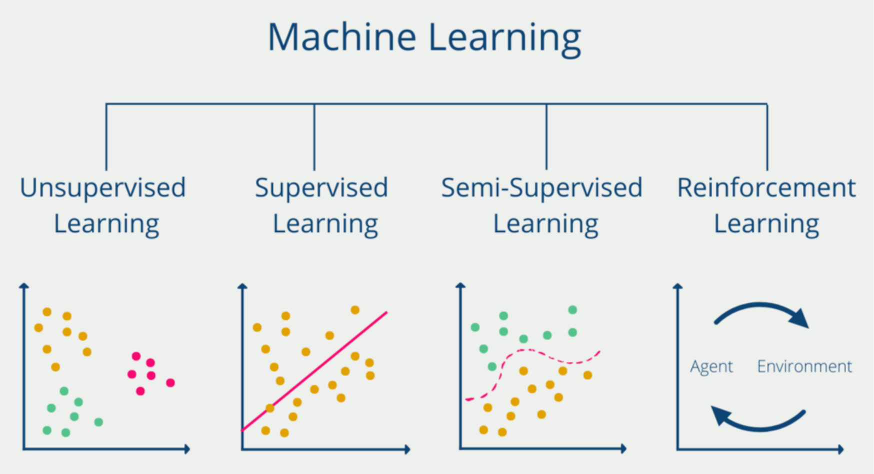

Figure 3: Overview of different learning tasks in Deep Learning.

Supervised Learning

In **supervised learning**, a model learns from **labeled input-output pairs**. It predicts outputs based on input data and compares them with known correct outputs, adjusting its internal parameters to **minimize prediction errors**. Application includes:

- Image classification (e.g., recognizing handwritten digits)
- Speech-to-text conversion
- Disease diagnosis from medical images

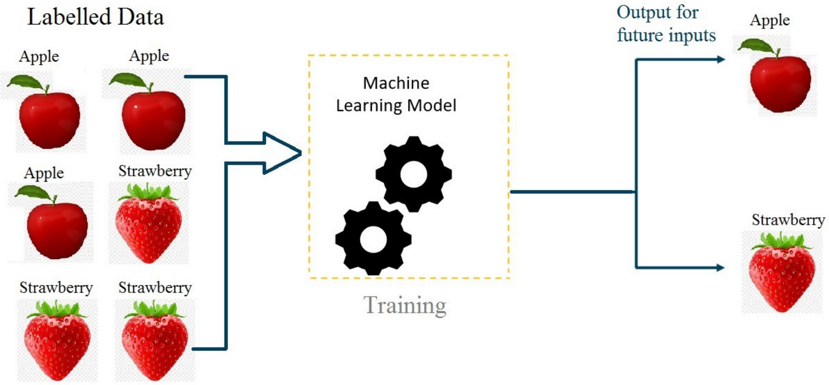

Figure 4: Supervised learning: Training with labeled data.

Unsupervised Learning

In **unsupervised learning**, the model learns patterns from **unlabeled data** without explicit instructions. Instead of mapping inputs to outputs, it identifies **structures, clusters, or relationships** within the data. Applications include:

- Customer segmentation in marketing
- Anomaly detection (e.g., fraud detection)
- Dimensionality reduction (e.g., PCA, autoencoders)

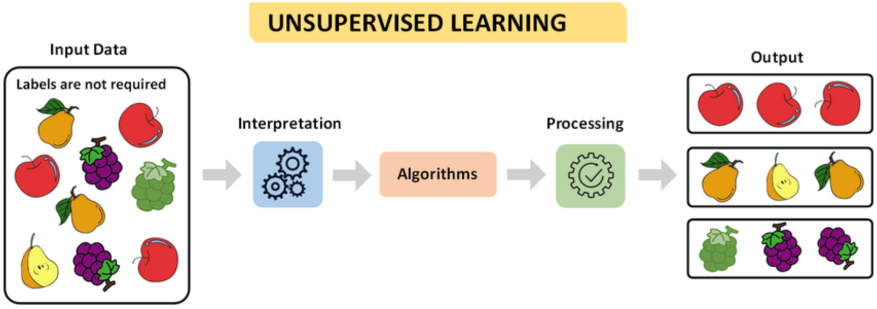

Figure 5: Unsupervised learning: Identifying patterns without labels.

Semi-Supervised Learning

Semi-supervised learning is a hybrid approach that **combines both labeled and unlabeled data**. This is especially useful when labeling data is expensive or time-consuming, allowing models to learn efficiently with a **small amount of labeled data**. Applications include:

- Speech recognition systems
- Medical diagnosis (learning from limited annotated medical images)
- Web page classification

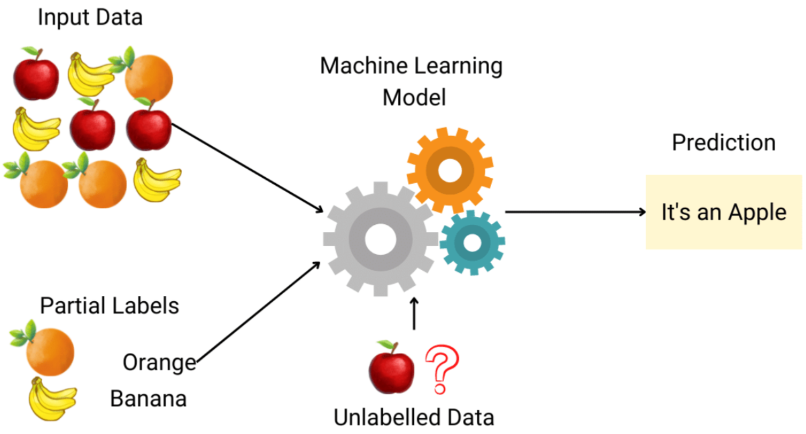

Figure 6: Semi-supervised learning: Leveraging both labeled and unlabeled data.

Reinforcement Learning (RL)

Reinforcement learning enables models to learn through **trial and error by interacting with an environment**. The model takes actions, receives feedback in the form of rewards or penalties, and gradually learns an optimal strategy to **maximize long-term rewards**. Applications include:

- Autonomous robots and self-driving cars
- AlphaGo and game AI (e.g., playing chess, Go, and video games)
- Financial market trading strategies

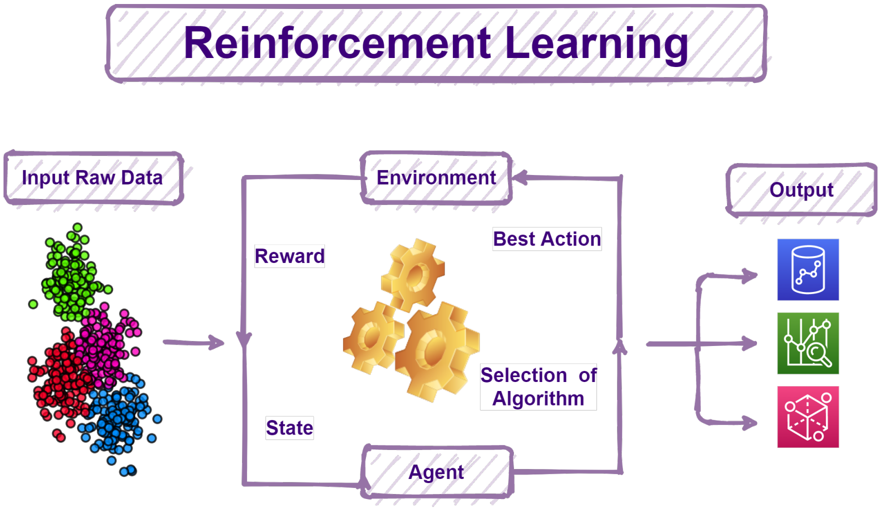

Figure 7: Reinforcement learning: Learning through rewards and penalties.

0.2 Artificial Neural Networks (ANNs)

Artificial Neural Networks (ANNs) are **the core of deep learning**, enabling computers to recognize complex patterns, solve intricate problems, and adapt to dynamic environments. Their ability to learn from vast amounts of data has revolutionized industries such as: **Natural Language Processing (NLP)** (e.g., ChatGPT, machine translation), **Self-Driving Vehicles** (e.g., autonomous navigation and perception), **Medical Diagnosis** (e.g., detecting diseases from medical images), **Automated Decision-Making** (e.g., fraud detection, financial predictions). Neural networks streamline processes, increase efficiency, and drive innovation, making them a fundamental technology shaping the future of artificial intelligence.

0.2.1 Evolution of neural networks

Neural networks have undergone significant evolution since their inception. Here's a timeline of major milestones:

- **1940s-1950s**: McCulloch and Pitts introduced the **first mathematical model for artificial neurons**. However, limited computational power slowed further progress.
- **1960s-1970s**: Frank Rosenblatt developed the **Perceptron**, a simple neural network that could handle linearly separable problems. However, perceptrons failed on more complex tasks, limiting their practical use.
- **1980s**: The breakthrough came with **backpropagation**, introduced by **Rumelhart, Hinton, and Williams**, allowing the training of multi-layer networks. This era saw the rise of connectionism, emphasizing learning through interconnected nodes.
- **1990s**: Neural networks gained popularity in fields like **image recognition and finance**. However, high computational costs and unrealistic expectations led to an "AI winter"—a decline in interest and funding.
- **2000s**: Advancements in **computing power (GPUs), larger datasets, and new architectures** sparked a resurgence. Deep learning, with multiple hidden layers, **surpassed traditional machine learning models** in various domains.
- **2010s-Present**: **Deep learning dominates AI**, with models like: Convolutional Neural Networks (CNNs) for image recognition, Recurrent Neural Networks (RNNs) for sequential data, Transformers for NLP tasks (e.g., ChatGPT, BERT).

0.2.2 Basic structure of neural networks

Deep learning is powered by **Artificial Neural Networks (ANNs)**, which are inspired by the **human brain**. These networks consist of **multiple layers** of **neurons (nodes)**, each performing mathematical operations to process and transform input data. Key Components of a Neural Network include:

- **Input Layer**: Receives raw data (e.g., images, text, signals).
- **Hidden Layers**: Extract features and perform complex transformations.
- **Output Layer**: Produces predictions (e.g., classification labels, probabilities).

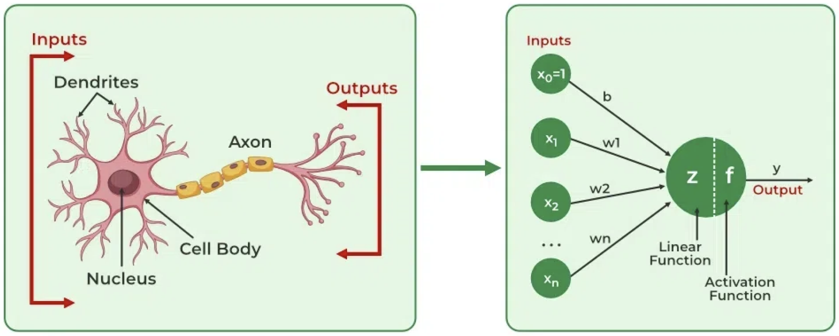

Figure 8: From biological neurons to artificial neural networks.

The figure below illustrates the architecture of an artificial neural network (ANN):

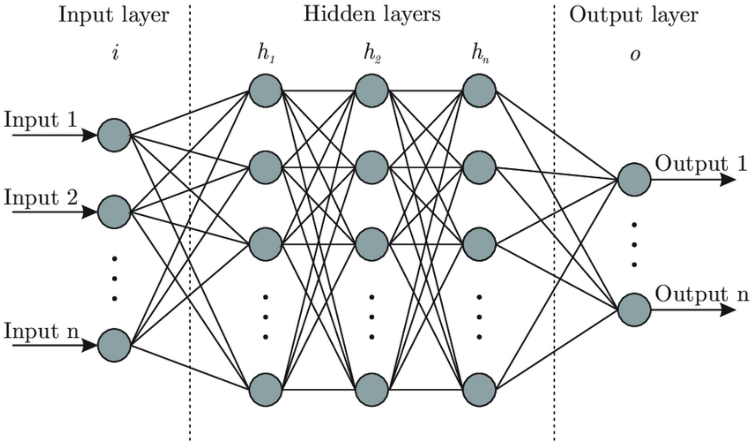

Figure 9: Architecture of artificial neural networks

Explanation of Layers:

- **Input Layer**: Each neuron represents a feature of the input data.
- **Hidden Layers**: These layers perform computations, learning representations from the input data. Deeper networks contain multiple hidden layers.
- **Output Layer**: Generates final predictions based on learned representations (regression).

0.2.3 Some types of neural networks

Different types of neural networks are designed for specific tasks. Below are some of the most common architectures:

- **Feedforward Neural Networks (FNNs)**: A feedforward neural network is the simplest type, where information flows in one direction—from input to output, without loops. Applications include:
  - Basic classification tasks (e.g., spam detection)
  - Simple regression models
- **Multilayer Perceptron (MLP)**: A Multilayer Perceptron (MLP) is a type of feedforward network with **at least one hidden layer**. It employs **nonlinear activation functions**, making it capable of solving complex problems. Applications include:
  - Handwritten digit recognition (e.g., MNIST dataset)
  - Stock price prediction

- **Convolutional Neural Networks (CNNs)**: A Convolutional Neural Network (CNN) is specifically designed for **image processing**. It uses **convolutional layers** to extract spatial features automatically, enabling effective **image recognition and classification**.

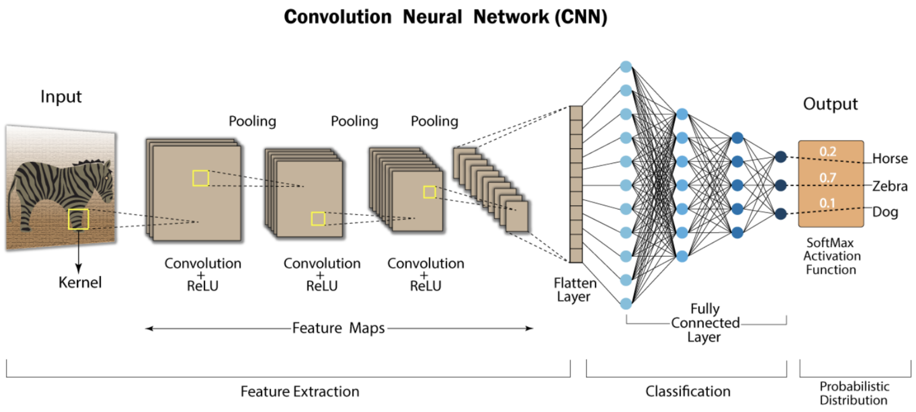

Figure 10: Architecture of a Convolutional Neural Network.

Applications include:

  - Image classification (e.g., facial recognition)
  - Medical imaging (e.g., tumor detection)

- **Recurrent Neural Networks (RNNs)**: A Recurrent Neural Network (RNN) is designed for **sequential data processing**, where past information influences future predictions. RNNs include **feedback loops**, allowing information to persist over time.

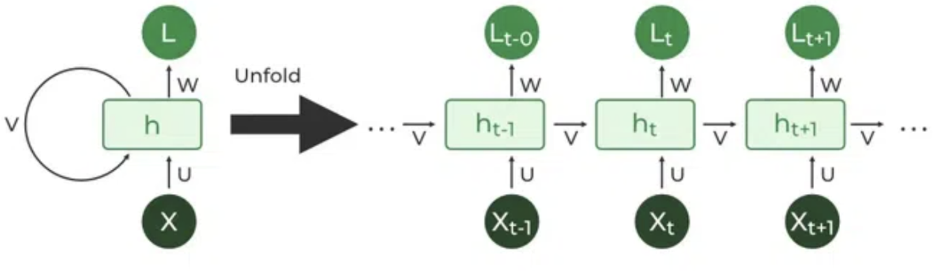

Figure 11: Architecture of a Recurrent Neural Network.

Applications include:

  - Natural Language Processing (NLP) (e.g., speech recognition, chatbots)
  - Time series forecasting (e.g., stock price prediction)

0.3 The Structure of Neural Networks

An Artificial Neural Network (ANN) is inspired by biological neural networks and is designed to **estimate or approximate functions that depend on multiple inputs**. These networks are widely used in machine learning and cognitive science to process complex data and make intelligent decisions.

Key Components of Neural Networks

- **Neurons**: Fundamental units that process information, each governed by a threshold and an activation function.
- **Connections**: Links between neurons that transmit information, regulated by weights and biases.
- **Weights and Biases**: Parameters that control the strength of neuron interactions.
- **Activation Functions**: Mathematical functions that introduce non-linearity, enabling neural networks to model complex patterns.

0.3.1 Single-Layer Perceptron (SLP)

One of the earliest and simplest neural networks is the **Single-Layer Perceptron (SLP)**, introduced by Frank Rosenblatt in 1958. It can solve simple problems like logical **AND, OR, and NOR gates** that involve binary inputs and outputs.

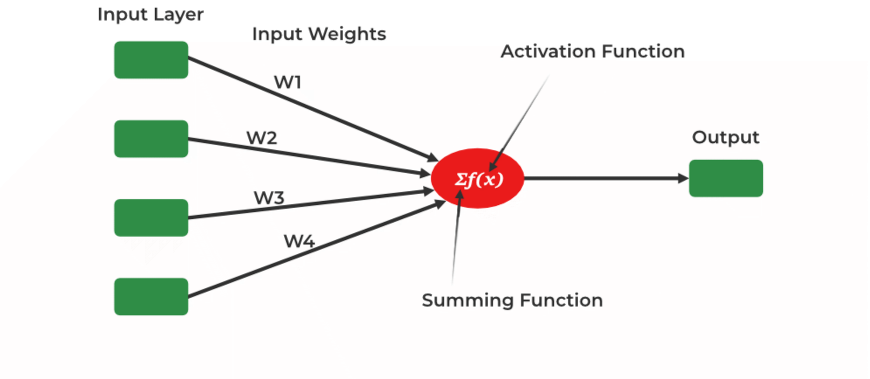

Figure 12: Structure of a Single-Layer Perceptron.

The main functionality of the perceptron is:

- **Takes input** $\{1, x_1, \cdots, x_n\}$ from the input layer.
- **Applies weights and bias**: Each input x_i is multiplied by weight w_i, and a bias b_0 is added.

- **Passes the weighted sum through an activation function** to produce the output:

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b_0 \cdot 1\right)$$

  where $\{1, x_1, \cdots, x_n\}$ are the inputs, $\mathbf{W} = \{b_0, w_1, \cdots, w_n\}$ are the parameters (weights and biases), and f is the activation function (e.g., sigmoid, tanh, ReLU).

Activation Functions

Activation functions play a crucial role in determining how signals propagate through the network. Common Activation Functions:

- **ReLU (Rectified Linear Unit):**

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

  helps mitigate the vanishing gradient problem.

- **Tanh (Hyperbolic Tangent):**

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

  produces values between -1 and 1, commonly used in hidden layers.

- **Sigmoid:**

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

  often used in binary classification problems.

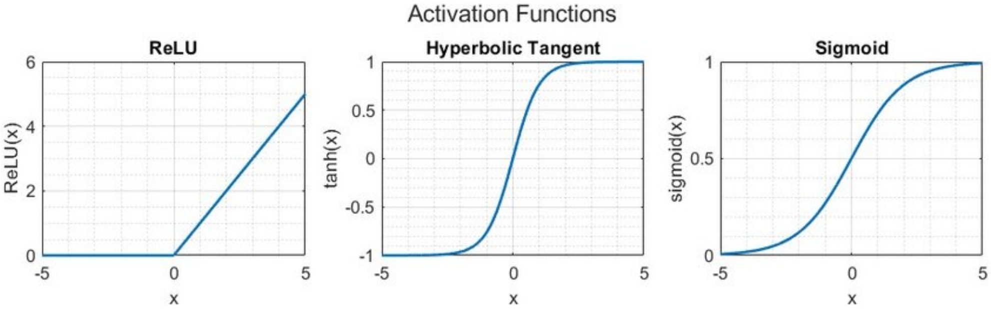

Figure 13: Common Activation Functions.

0.3.2 Multi-Layer Perceptrons (MLP)

A Multi-Layer Perceptron (MLP) consists of fully connected dense layers that transform input data from one dimension to another. It is called "multi-layer" because it contains an input layer, one or more hidden layers, and an output layer. The purpose of an MLP is to model complex relationships between inputs and outputs, making it a powerful tool for various machine-learning tasks.

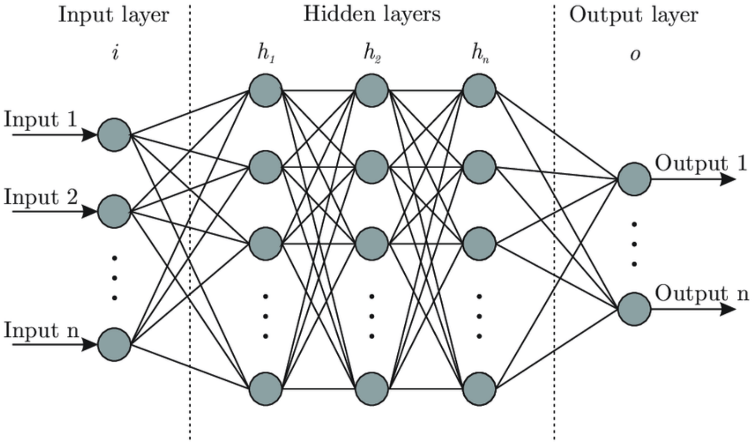

Figure 14: Structure of a Multi-Layer Perceptron (MLP).

Mathematical Representation of MLP

The output of the j-th neuron in the l-th hidden layer can be expressed as:

$$h_j^{(l)} = f_l \left(\sum_{i=1}^{m} w_{i,l} x_i + b_{j,l} \cdot 1 \right)$$

where:

- $h_j^{(l)}$ is the output of neuron j in layer l,
- $w_{i,l}$ are the weights of the connections,
- $b_{j,l}$ is the bias term,
- f_l is the activation function for layer l.

Overall Representation of an MLP

In a general neural network model, the final output y can be described as:

$$y = NN(x;\theta) = W^{(l)} f^{(l)} \odot \left(W^{(l-1)} \odot f^{(l-1)}(\cdots) + b_{l-1} \right) + b_l$$

where:

- θ represents all trainable parameters,

- $\odot$ denotes element-wise multiplication,
- $W^{(l)}$ and b_l are the weight matrix and bias for layer l.

0.3.3 Universal Approximation Theorem

One of the most powerful results in neural network theory is the **Universal Approximation Theorem**, which states:

> A sufficiently large neural network with at least one hidden layer can approximate any continuous function to an arbitrary degree of accuracy.

The mathematical formulation is given as follows:

> For any continuous function $u : K \to \mathbb{R}$ in a compact $K \subset \mathbb{R}^d$ and $\epsilon > 0$, there exists an MLP $NN(x;\theta)$ such that:
>
> $$\sup_{x\in K} |u(x) - NN(x,\theta)| < \epsilon$$