UNIT-1

1.What is MP neuron?

An **MP neuron** (McCulloch–Pitts neuron) is one of the earliest mathematical models of a biological neuron, introduced by **Warren McCulloch** and **Walter Pitts** in 1943.

It's a **simplified computational model** of how a neuron processes inputs and produces an output.

Structure of an MP neuron

It consists of:

1. **Inputs** → $x_1, x_2, \ldots, x_n$ (binary values: 0 or 1 in the original model).
2. **Weights** → $w_1, w_2, \ldots, w_n$ (representing the strength of each connection).
3. **Summation function** → Adds up weighted inputs:

$$S = \sum_{i=1}^{n} w_i \cdot x_i$$

4. **Threshold** (θ) → A value the sum must reach for the neuron to "fire."
5. **Activation function** → A step function that outp
   - 1 if $S \geq \theta$ (neuron fires)

Working principle

The MP neuron takes inputs, multiplies them by weights, sums them up, and then applies a threshold-based decision:

$$y = \begin{cases} 1 & \text{if } S \geq \theta \\ 0 & \text{if } S < \theta \end{cases}$$

Example

Suppose:

- Inputs: $x_1 = 1, x_2 = 0, x_3 = 1$
- Weights: $w_1 = 1, w_2 = 1, w_3 = 1$
- Threshold $\theta = 2$

Calculation:

$$S = 1 \cdot 1 + 0 \cdot 1 + 1 \cdot 1 = 2$$

Since $S = \theta$, the neuron outputs **1** (fires).

Key Points

- It's a **binary** model — both inputs and outputs are typically 0 or 1.
- No learning mechanism was in the original model — weights and threshold are fixed.
- It can implement basic logic gates (**AND, OR, NOT**) but **cannot** solve non-linearly separable problems like XOR.
- It laid the foundation for modern **Artificial Neural Networks (ANNs)**.

2.Tell about Thresholding logic.

Thresholding logic is the decision-making principle behind early artificial neurons (like the MP neuron) and some modern activation functions.

In **thresholding logic**, the output of a neuron (or logic unit) is determined by whether the **weighted sum of its inputs** reaches a certain **threshold** value.

How it works

1. **Take multiple inputs** $(x_1, x_2, \ldots, x_n)$.
2. **Assign weights** $(w_1, w_2, \ldots, w_n)$ to indicate the importance of each input.
3. **Calculate weighted sum:**

$$S = \sum_{i=1}^{n} w_i \cdot x_i$$

4. **Compare to threshold** (θ):
   - If $S \geq \theta$ → Output = **1** (ON / True)
   - If $S < \theta$ → Output = **0** (OFF / False)

Mathematical Representation

$$y = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} w_i x_i \geq \theta \\ 0 & \text{if } \sum_{i=1}^{n} w_i x_i < \theta \end{cases}$$

Example – AND Gate using Threshold Logic

Inputs: $x_1, x_2 \in \{0, 1\}$
Weights: $w_1 = 1, w_2 = 1$
Threshold: $\theta = 2$

x_1	x_2	Weighted Sum	Output
0	0	0	0
0	1	1	0
1	0	1	0
1	1	2	1

Key Points

- Thresholding logic is **binary** — outputs are discrete (0 or 1).
- It's the basis of **perceptrons** and **logic gate implementations** in neural networks.
- It can represent **linearly separable** decision boundaries.
- If a problem is **not linearly separable** (like XOR), pure thresholding logic fails without additional network layers.

3.Relate MP neuron with biological neuron.

Relating MP Neuron with Biological Neuron

The **MP neuron** (McCulloch-Pitts neuron) is an early mathematical model inspired directly by how a biological neuron functions in the nervous system.

Biological Neuron:

- **Structure:**

  A biological neuron has a cell body (soma), multiple **dendrites** (inputs), and a single **axon** (output).
- **Function:**
  - Dendrites receive electrochemical signals from other neurons.
  - These signals are summed (integrated) in the cell body.
  - If the combined signal reaches a certain **threshold**, the neuron "fires" (sends an electrical pulse down the axon).
  - Otherwise, it remains inactive.

MP Neuron Model:

- **Inputs:**

  The MP neuron accepts multiple binary inputs (analogous to signals from dendrites).
- **Summation & Threshold:**

  All inputs are added, and if the total meets or exceeds a defined **threshold**, the neuron "fires" (outputs 1), resembling the biological neuron's action potential.
- **Output:**

  If the threshold is not met, the output is 0 (no firing).

Comparison Table

Aspect	Biological Neuron	MP Neuron Model
Inputs	Dendrites (graded signals)	Multiple binary values (0 or 1)
Summation	Electrical summing in the soma	Mathematical addition of binary inputs
Activation	Fires if threshold reached (action potential)	Outputs 1 if threshold reached, 0 otherwise
Output	Signal sent via axon to other neurons	Single binary output (to other neurons/nodes)
Complexity	Real neurons have more complex integr[illegible]n	MP model is simplified and binary

4.Notes limitations on MP neuron.

1. Binary inputs and outputs only

- Original MP neuron works only with 000 or 111 values.
- Cannot handle real-valued or continuous signals like those in real neurons.

---

2. No learning mechanism

- Weights and threshold are **fixed** and must be set manually.
- Cannot adjust or "learn" from data like modern neural networks.

---

3. Oversimplified biology

- Ignores time-dependent effects, refractory periods, and graded potentials in real neurons.
- Assumes simultaneous arrival of inputs — no sequence or timing consideration.

---

4. Can only solve linearly separable problems

- MP neurons cannot handle problems like **XOR** without adding more layers.
- Decision boundary is always a straight line (or hyperplane in higher dimensions).

---

5. No handling of noise

- Small input changes can flip the output entirely.
- Not robust for noisy or uncertain data.

---

6. Lacks complex activation functions

- Uses only a **step function** — cannot model smooth or probabilistic firing.

5.Explain Perceptron model with architecture?

The **Perceptron model** is an improvement over the MP neuron — it's still inspired by the biological neuron, but it introduces a **learning rule** so it can adjust its weights automatically.

It was introduced by **Frank Rosenblatt in 1958** and is considered the foundation of modern neural networks.

---

1. Structure of a Perceptron

A perceptron has these main components:

1. **Inputs** $(x_1, x_2, \ldots, x_n)$ – the features or signals.
2. **Weights** $(w_1, w_2, \ldots, w_n)$ – determine the importance of each input; updated during learning.
3. **Summation Function** – computes the weighted sum:

$$S = \sum_{i=1}^{n} w_i x_i + b$$

where b is the **bias** (acts like a negative threshold).

4. **Activation Function** – applies a decision rule:
   - Step function in the basic perceptron:

$$y = \begin{cases} 1 & \text{if } S \geq 0 \\ 0 & \text{if } S < 0 \end{cases}$$

5. **Output** – the final binary prediction.

---

2. Perceptron Architecture

lua Copy Edit

```lua
x1  ----(w1)---\
x2  ----(w2)----> Σ  --> Activation --> Output (y)
x3  ----(w3)---/  ↑
                  b (bias)
```

- **Input layer**: takes the raw features.
- **Processing unit**: sums up weighted inputs + bias.
- **Activation unit**: decides the output based on the sum.
- **Output layer**: gives a binary result (0 or 1).

---

3. Learning in the Perceptron

The perceptron **adjusts weights** after each training example using the rule:

$$w_i^{new} = w_i^{old} + \eta \cdot (y_{true} - y_{pred}) \cdot x_i$$

$$b^{new} = b^{old} + \eta \cdot (y_{true} - y_{pred})$$

where:

- η = learning rate
- y_{true} = actual label
- y_{pred} = predicted output

4. Key Points

- **Improvement over MP neuron**: can learn weights automatically.
- Still only works for **linearly separable** problems (like AND, OR, NOT — but not XOR).
- Forms the building block of **multi-layer neural networks.**

6.Implement ANDNOT gate using MP Neuron

1. Truth Table for ANDNOT (A ∧ ¬B)

A (x_1)	B (x_2)	NOT B	Output (A AND NOT B)
0	0	1	0
0	1	0	0
1	0	1	1
1	1	0	0

2. MP Neuron Setup

We want:

- Output = 1 only when $x_1 = 1$ and $x_2 = 0$.

Step 1 — Assign weights

- Let $w_1 = 1$ (for A, positive effect)
- Let $w_2 = -1$ (for B, negative effect because we want NOT B behavior)

Step 2 — Choose threshold

We want:

- For **(1, 0)** → $(1)(1) + (-1)(0) = 1$ → should be ≥ threshold → output 1
- For all other cases, sum should be < threshold → output 0

If we set $\theta = 1$, this works.

3. MP Neuron Equation

$$y = \begin{cases} 1 & \text{if } (1 \cdot x_1) + (-1 \cdot x_2) \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

4. Verification

x_1	x_2	Weighted Sum $S = 1x_1 - 1x_2$	Output (S ≥ 1 ?)
0	0	0	0
0	1	-1	0
1	0	1	1
1	1	0	0

✅ Matches the ANDNOT gate truth table.

Implement ANDNOT function using McCulloch–Pitts Neuron

- Consider the truth table for ANDNOT function
- The M–P neuron has no particular training algorithm
- In M-P neuron, only analysis is being performed.
- Hence, assume the weights be w1 = 1 and w2 = -1.

x_1	x_2	y
0	0	0
0	1	0
1	0	1
1	1	0

$$y_{in} = x_1 w_1 + x_2 w_2$$

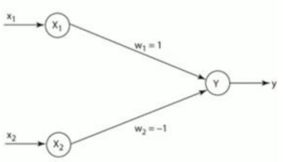
$(1, 1),\ y_{in} = 1 \times 1 + 1 \times -1 = 0$

$(1, 0),\ y_{in} = 1 \times 1 + 0 \times -1 = 1$

$(0, 1),\ y_{in} = 0 \times 1 + 1 \times -1 = -1$

$(0, 0),\ y_{in} = 0 \times 1 + 0 \times -1 = 0$

From the calculated net inputs, now it is possible to fire the neuron for input (1, 0) only by fixing a threshold of 1, i.e., θ ≥ 1 for Y unit.



7.Differentiate between McCulloch-Pits Model and Perceptron model?

Feature	McCulloch–Pitts (MP) Model	Perceptron Model
Introduced by	Warren McCulloch & Walter Pitts (1943)	Frank Rosenblatt (1958)
Purpose	Logical modeling of a neuron using threshold logic.	Pattern recognition and classification with learning capability.
Inputs	Binary only (0 or 1).	Real-valued inputs allowed.
Weights	Fixed (predefined, no adjustment).	Learnable (adjusted during training).
Bias term	Not explicitly used.	Explicitly included for flexible decision boundary.
Activation function	Step function (binary output).	Step function in basic form (can be extended).
Learning rule	None — no training process.	Uses Perceptron Learning Rule to update weights.
Capability	Implements basic logic gates (AND, OR, NOT).	Can solve linearly separable classification problems.
Adaptability	Cannot adapt to new data.	Can adapt weights to fit data.
Limitation	Only models fixed logical operations.	Still fails on non-linearly separable problems like XOR.

💡 **In short**:

- **MP neuron** = theoretical, fixed-logic model → no learning, binary only.
- **Perceptron** = practical, trainable model → can learn from data but still limited to linear separability.

8.What are the limitations of the MP neuron model, and how are these limitations addressed by the perceptron model? Discuss in detail.

Limitations of the MP Neuron Model

The McCulloch-Pitts (MP) neuron model is foundational to artificial neural networks but has several significant limitations:

1. **Binary Inputs and Outputs:**
   - MP neurons only accept binary inputs (0 or 1) and produce binary outputs.
   - This prevents modeling problems with continuous or multi-valued data typically found in real-world applications.
2. **No Learning Mechanism:**
   - MP neuron weights and structure are fixed; they cannot adapt or learn from data through training.
   - There is no algorithm to modify weights or thresholds in response to mistakes or new information.
3. **Linear Separability Constraint:**
   - MP neurons can only solve linearly separable problems such as AND, OR, NOT logic gates.
   - They cannot represent non-linear functions like XOR, which require more complex separation.
4. **Simplistic Activation Function:**
   - The MP neuron only uses a simple threshold activation (step function), limiting its ability to model complex decision boundaries.

5. **Insensitivity to Noise and Variability:**
   - Because of binary processing, MP neurons are highly sensitive to noise and cannot handle real-world, noisy data flexibly.
6. **No Weights (in classic form):**
   - Early MP neuron models did not include adjustable weights for inputs, restricting their computational power.

How the Perceptron Model Addresses MP Neuron Limitations

The **Perceptron model**, introduced by Frank Rosenblatt, significantly improves upon the MP neuron through several key mechanisms:

1. **Supports Continuous Inputs:**
   - Perceptrons work with both binary and real-valued (continuous) input features, greatly expanding the types of data they can process.
2. **Incorporates Adjustable Weights and Bias:**
   - Every input is assigned a trainable weight, and an additional bias term is included.
   - These weights and bias allow the perceptron to learn and modify its behavior based on training data.
3. **Learning Ability:**
   - The perceptron uses a learning algorithm that adapts weights with each input/output pair, minimizing errors and enabling the model to improve over time (using techniques such as gradient descent).
4. **Flexible Decision Boundaries:**
   - With adjustable weights and bias, perceptrons can model a wider range of linearly separable decision boundaries than the rigid MP neuron.
5. **Improved Noise Handling:**
   - Real-valued weights and bias make the perceptron less sensitive to noise and able to better handle small variations in input data.
6. **Foundation for Multi-layer Networks:**
   - While a single-layer perceptron is still limited to linearly separable problems, its architecture can be expanded to form multi-layer neural networks, which can solve non-linear tasks (such as XOR).

Summary Table: MP Neuron vs. Perceptron

Feature	MP Neuron Model	Perceptron Model
Input Type	Only binary (0 or 1)	Binary and continuous (real-valued)
Weights and Bias	Not adjustable (fixed)	Trainable, adjusts during learning
Learning Capability	None	Yes, through supervised learning
Activation Function	Simple threshold (step)	Step, with trainable boundary
Problem Types Handled	Linear (AND, OR) only	Linear separable with flexible boundary
Handling of Noise & Variability	Poor	Better, due to continuous adjustment

9.State and prove Convergence theorem for Perceptron Learning Algorithm.