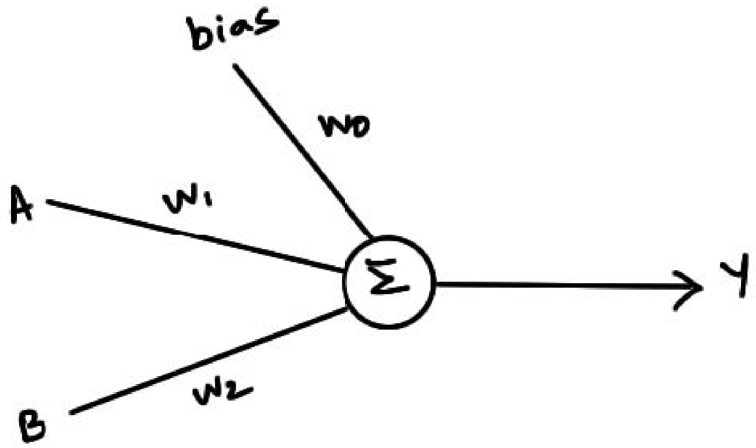
CS7641 Week 3 Tuesday Problem Solution

Designing a Perceptron for $A \land \neg B$

To implement the boolean function $A \land \neg B$ using a perceptron, we need to set up weights and a threshold such that the perceptron only outputs 1 (true) when A is true and B is false. The truth table for $A \land \neg B$ is as follows:

A	B	$A \land \neg B$
0	0	0
0	1	0
1	0	1
1	1	0

Table 1: Truth table for $A \land \neg B$



Looking at the truth table and the perceptron, we can infer that,

$$w_0 > 0$$
$$w_1 > w_0$$
$$w_1 + w_2 < w_0$$

We can see that w_2 has to be negative. Any combination of weights that satisfy these conditions will model Y.

Designing a Two-Layer Network for $A \oplus B$ (XOR)

The XOR function $A \oplus B$ is more complex and cannot be implemented using a single layer of perceptrons as XOR is not linearly separable. The truth table for XOR is:

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Table 2: Truth table for $A \oplus B$ (XOR)

To implement XOR, we can use a two-layer network:

Layer 1:

- Neuron 1 (implements $A \vee B$): $w_1 = 1, w_2 = 1, b_1 = -0.5$
- Neuron 2 (implements $\neg(A \wedge B)$): $w_1 = -1, w_2 = -1, b_2 = 1.5$

Layer 2:

- Single Neuron (implements logical AND of the outputs from Layer 1): $w_1 = 1, w_2 = 1, b = -1.5$

The output from the first layer are:

$$o_1 = (A + B > 0.5) \text{ (logical OR)}$$
$$o_2 = \neg(A \& B) \text{ (NAND)}$$

These outputs are inputs to the second layer which outputs:

$$o = (o_1 \wedge o_2)$$

We derive the perceptron training rule and the gradient descent training rule for a single unit with output o, given by the equation:

$$o = w_0 + w_1 x_1 + w_1 x_1^2 + \ldots + w_n x_n + w_n x_n^2$$