Feedforward Neural Network with tanh Activation – Version 3

Python Code Example

```
import numpy as np

# ----- tanh Activation Function -----
def tanh(x):
  return np.tanh(x)

# Input values
X = np.array([[0.05], [0.10]])

# Biases
bias1 = 0.5
bias2 = 0.7

# Random Weights initialization in range [-0.5, 0.5]
np.random.seed(100)
W_input_to_hidden = np.random.uniform(-0.5, 0.5, (2, 2))
W_hidden_to_output = np.random.uniform(-0.5, 0.5, (2, 2))

# ----- Forward Pass -----

# Hidden layer computation
hidden_input = np.dot(W_input_to_hidden, X) + bias1
hidden_output = tanh(hidden_input)

# Output layer computation
final_input = np.dot(W_hidden_to_output, hidden_output) + bias2
final_output = tanh(final_input)

print("Hidden Layer Output:\n", hidden_output)
print("Final Network Output:\n", final_output)
```

Feedforward Steps

1 . Hidden Layer Input:

Z_h = W_input_to_hidden * X + bias 1

1 . Hidden Layer Activation:

H = tanh(Z_h)

1. Output Layer Input:

Z_o = W_hidden_to_output * H + bias 2

1. Output Activation:

Output = tanh(Z_o)

Sample Output (Random Weights – Values May Vary)

Hidden Layer Output: [[0.4 0 2] [0.2 9 8]]

Final Network Output: [[0.6 1 5] [0.5 7 4]]

Backpropagation Overview

1️⃣ Compute Error:

Error = Target - Output

2️⃣ Output Layer Delta:

delta_o = Error * tanh'(Z_o)

3️⃣ Hidden Layer Delta:

delta_h = (W_hidden_to_output.T * delta_o) * tanh'(Z_h)

4️⃣ Weight Update:

W_new = W_old + learning_rate * delta * input