# Simple Neural Network Forward Propagation Example

Example: Forward propagation for 2 images with 3 input features.  
Input: X with shape (3, 2) representing 3 pixels and 2 images.  
Network structure:  
- Input layer: 3 neurons (pixels)  
- Hidden layer: 3 neurons  
- Output layer: 2 neurons (2 classes)  
- Activation: ReLU (hidden), Softmax (output)  
  
Parameters initialized:  
W1 (3x3), B1 (3x1), W2 (2x3), B2 (2x1) with example values.  
  
Step-by-step calculation:

Input X (3 pixels, 2 images):

|  |  |  |
| --- | --- | --- |
| Pixel # | Image 1 | Image 2 |
| 1 | 0.1 | 0.3 |
| 2 | 0.2 | 0.4 |
| 3 | 0.3 | 0.5 |

Weights W1 (3x3):

[[ 0.2, 0.1, -0.1],  
 [ 0.0, 0.1, 0.2],  
 [-0.1, 0.2, 0.1]]

Biases B1 (3x1):

[[0.1], [0.2], [0.3]]

Calculate Z1 = W1 · X + B1

W1 · X:

Neuron 1:

Image 1: 0.2\*0.1 + 0.1\*0.2 + (-0.1)\*0.3 = 0.02 + 0.02 - 0.03 = 0.01

Image 2: 0.2\*0.3 + 0.1\*0.4 + (-0.1)\*0.5 = 0.06 + 0.04 - 0.05 = 0.05

Neuron 2:

Image 1: 0\*0.1 + 0.1\*0.2 + 0.2\*0.3 = 0 + 0.02 + 0.06 = 0.08

Image 2: 0\*0.3 + 0.1\*0.4 + 0.2\*0.5 = 0 + 0.04 + 0.1 = 0.14

Neuron 3:

Image 1: -0.1\*0.1 + 0.2\*0.2 + 0.1\*0.3 = -0.01 + 0.04 + 0.03 = 0.06

Image 2: -0.1\*0.3 + 0.2\*0.4 + 0.1\*0.5 = -0.03 + 0.08 + 0.05 = 0.10

Add biases B1 (broadcasted):  
Z1 = [[0.01+0.1, 0.05+0.1],  
 [0.08+0.2, 0.14+0.2],  
 [0.06+0.3, 0.10+0.3]] =  
[[0.11, 0.15],  
 [0.28, 0.34],  
 [0.36, 0.40]]

Apply ReLU activation: A1 = max(0, Z1)

A1 = [[0.11, 0.15],  
 [0.28, 0.34],  
 [0.36, 0.40]] (all positive)

Weights W2 (2x3):

[[ 0.3, -0.2, 0.1],  
 [ 0.4, 0.1, -0.3]]

Biases B2 (2x1): [[0.1], [0.2]]

Calculate Z2 = W2 · A1 + B2

Output neuron 1:  
Image 1: 0.3\*0.11 + (-0.2)\*0.28 + 0.1\*0.36 = 0.033 - 0.056 + 0.036 = 0.013  
Image 2: 0.3\*0.15 + (-0.2)\*0.34 + 0.1\*0.40 = 0.045 - 0.068 + 0.04 = 0.017

Output neuron 2:  
Image 1: 0.4\*0.11 + 0.1\*0.28 + (-0.3)\*0.36 = 0.044 + 0.028 - 0.108 = -0.036  
Image 2: 0.4\*0.15 + 0.1\*0.34 + (-0.3)\*0.40 = 0.06 + 0.034 - 0.12 = -0.026

Add biases B2:  
Z2 = [[0.013+0.1, 0.017+0.1],  
 [-0.036+0.2, -0.026+0.2]] =  
[[0.113, 0.117],  
 [0.164, 0.174]]

Apply softmax column-wise for each image:  
Image 1: softmax([0.113, 0.164]) ≈ [0.487, 0.513]  
Image 2: softmax([0.117, 0.174]) ≈ [0.486, 0.514]

Final output probabilities A2:  
[[0.487, 0.486],  
 [0.513, 0.514]]