# Backpropagation in Neural Networks

Generated by LectureMate

February 21, 2025

## 1 Introduction

In this lecture, we discussed the concept of backpropagation, a crucial algorithm in training neural networks. We began with a recap of the perceptron model, which serves as the foundation for understanding more complex architectures like the multilayer perceptron (MLP).

## 2 Recap of the Perceptron

The perceptron is a simple neural structure that learns to associate input patterns with output patterns. It consists of:

- Inputs $x_1, x_2, \ldots, x_n$
- Weights $w_1, w_2, \ldots, w_n$
- An activation function that determines the output based on a weighted sum of inputs.

The output $y$ is generated using a threshold function:

$$y = \begin{cases} 1 & \text{if } u(x) \geq \theta \\ 0 & \text{otherwise} \end{cases}$$

where $u(x) = \sum_{i=1}^{n} w_i x_i$.

### 2.1 Limitations of the Perceptron

The perceptron can only solve linearly separable problems. For example, it cannot solve the XOR problem, which requires a non-linear decision boundary.

## 3 Multilayer Perceptron (MLP)

To overcome the limitations of the single-layer perceptron, we introduced the multilayer perceptron, which consists of:

- An input layer

- One or more hidden layers
- An output layer

The MLP can approximate any function due to its ability to learn non-linear decision boundaries.

### 3.1 Architecture of MLP

The MLP is fully connected, meaning each neuron in one layer is connected to every neuron in the next layer. This architecture allows for complex computations and learning.

## 4 Learning in MLP

The learning process involves adjusting the weights based on the error between the predicted output and the target output. This is achieved through the backpropagation algorithm.

### 4.1 Error Calculation

The error for a given output is calculated as:

$$\text{Error} = T - O$$

where $T$ is the target output and $O$ is the actual output.

### 4.2 Weight Update Rule

The weights are updated using the learning rule:

$$\Delta w_{ij} = \eta \cdot \delta_j \cdot x_i$$

where $\eta$ is the learning rate, $\delta_j$ is the error term for neuron $j$, and $x_i$ is the input to the neuron.

## 5 Backpropagation Algorithm

Backpropagation involves two main steps:

1. **Forward Pass:** Compute the output of the network for a given input.
2. **Backward Pass:** Calculate the error and propagate it back through the network to update the weights.

### 5.1 Error Propagation

For output neurons, the error is directly calculated. For hidden neurons, the error is propagated back from the output layer, weighted by the connections:

$$\delta_i = \sum_j \delta_j w_{ij}$$

## 6 Activation Functions

We discussed the transition from a step function to a sigmoid activation function, which is differentiable and allows for smoother updates during training. The sigmoid function is defined as:

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

### 6.1 Advantages of Sigmoid Function

The sigmoid function has several advantages:

- It is differentiable, allowing for gradient-based optimization.
- It outputs values in a continuous range (0, 1), facilitating learning.

## 7 Gradient Descent

Gradient descent is used to minimize the error function. The weight updates are performed iteratively until convergence is achieved. The update rule is:

$$w_{ij} \leftarrow w_{ij} - \eta \frac{\partial E}{\partial w_{ij}}$$

where $E$ is the error function.

## 8 Conclusion

In conclusion, backpropagation is a powerful algorithm that enables the training of multilayer perceptrons by efficiently computing gradients and updating weights. Understanding this process is essential for working with neural networks and deep learning.