Lesson 7: Multi-Layer Perceptron

7.1 Introduction 1
7.2 Multilayer perceptron 1
7.2.1 Forward propagation 1
7.2.2 Backward propagation 2
7.3 Error backpropagation learning 2
7.4 Lesson 7 Questions 3

7.1 Introduction

In this lesson, we will be discussing the Multi-Layer Perceptron (MLP) and the Error Backpropagation Learning (EBL) algorithm. An MLP is a type of artificial neural network that is composed of multiple layers of interconnected nodes, or "neurons." These layers work together to process and analyze complex inputs, such as images or audio data. MLPs are particularly useful for tasks such as image classification, speech recognition, and natural language processing.

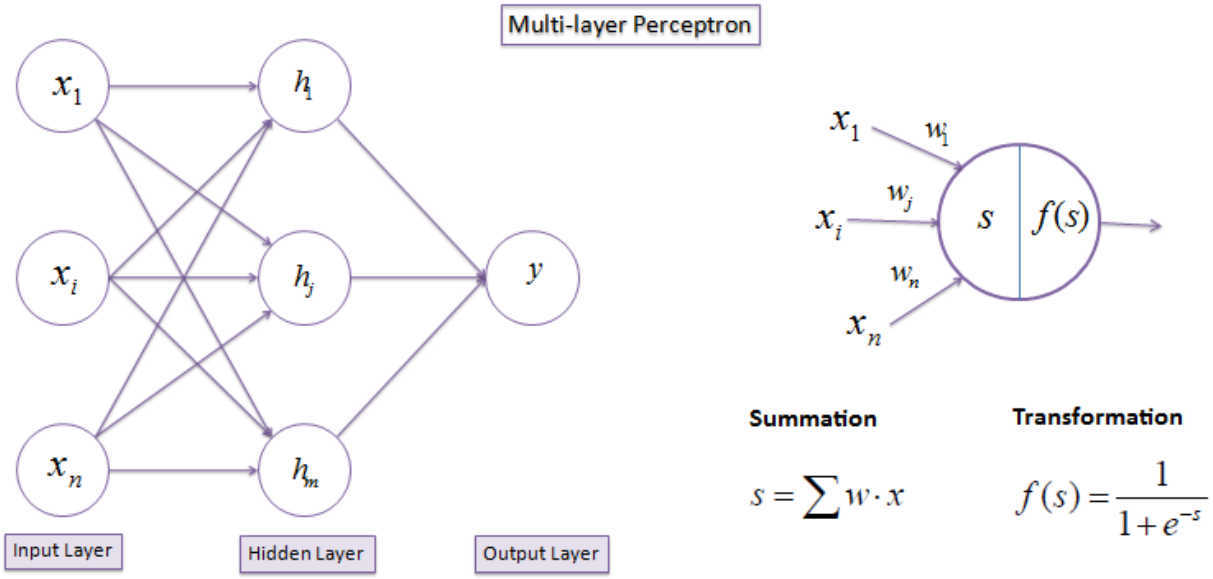
EBL is a supervised learning algorithm that is used to train MLPs. It utilizes the gradient descent method to adjust the weights of the network in order to minimize the error between the predicted output and the actual output. The process of EBL involves forwarding the input through the network, calculating the error, and then propagating this error backwards through the network to adjust the weights. This process is repeated for multiple iterations, or until the error reaches a certain threshold. By the end of this lesson, you will have a deeper understanding of how MLPs and EBL work together to improve the accuracy of predictions in artificial neural networks.

7.2 Multilayer perceptron

A multi-layer perceptron (**MLP**) has the same structure of a single layer perceptron with one or more hidden layers. The backpropagation algorithm consists of two phases: the forward phase where the activations are propagated from the input to the output layer, and the backward phase, where the error between the observed actual and the requested nominal value in the output layer is propagated backwards in order to modify the weights and bias values.

7.2.1 Forward propagation

Propagate inputs by adding all the weighted inputs and then computing outputs using sigmoid threshold



7.2.2 Backward propagation

Propagates the errors backward by apportioning them to each unit according to the amount of this error the unit is responsible for.

1. Error in any ***output*** neuron

$$d_o = y \times (1-y) \times (t-y)$$

2. Error in any ***hidden*** neuron

$$d_i = y_i \times (1-y_i) \times (w_i \times d_o)$$

3. Change the ***weights***

$$\Delta w = \eta \times d \times x$$

7.3 Error backpropagation learning

Error backpropagation learning, also known as backpropagation, is a supervised learning algorithm used in artificial neural networks to adjust the weights of the network in order to minimize the error between the predicted output and the actual output. The algorithm is based on the concept of gradient descent, which is a method used to optimize functions by finding the steepest descent towards the minimum value of the function. The backpropagation algorithm starts by feeding the input data through the network, which results in an output prediction. The error between the predicted output and the actual output is then calculated using a loss function, such as mean squared error. The error is then propagated backwards through the network, starting from the output layer and working backwards through the hidden layers.

As the error is propagated backwards, the weights of the network are adjusted in order to minimize the error. The adjustment is done using a method called gradient descent, which calculates the gradient of the error function with respect to the weights of the network. The gradient is then used to update the weights in the opposite direction of the gradient, which will decrease the error. The process of updating the weights and propagating the error backwards is repeated multiple times until the error reaches a satisfactory level or the algorithm converges. The backpropagation algorithm is a powerful method for training neural networks and has been successfully applied to a wide range of applications, such as image recognition, speech recognition, and natural language processing.

One of the key advantages of backpropagation is that it can handle complex, non-linear relationships between inputs and outputs by adjusting the weights of the network in a way that minimizes the error. However, the algorithm can be sensitive to the choice of learning rate, which can affect the speed of convergence and the overall performance of the network. Additionally, backpropagation can be computationally intensive, particularly for large networks with many hidden layers and a large number of neurons.

7.4 Lesson 7 Questions

1. What is a multilayer perceptron and how does it differ from a single-layer perceptron?
2. How does the error backpropagation algorithm work in a multilayer perceptron network?
3. What is the role of the gradient descent algorithm in the error backpropagation learning process?
4. How is the error calculated in a multilayer perceptron network during the backpropagation process?
5. How does the choice of learning rate affect the performance of a multilayer perceptron network trained using error backpropagation?
6. What are some common challenges and limitations of using error backpropagation learning in multilayer perceptron networks?
7. How can overfitting be avoided when using error backpropagation learning in a multilayer perceptron network?
8. How does the number of hidden layers in a multilayer perceptron network affect the performance of the network when trained using error backpropagation?
9. What are some applications of multilayer perceptron networks trained using error backpropagation?
10. How can the performance of a multilayer perceptron network trained using error backpropagation be evaluated and improved,