18.2.2 Backpropagation Networks

The next few sections are devoted to a learning algorithm known as backpropagation. Although we consider it to be an interesting exercise in theory to develop a version of this algorithm, we do not attempt to develop a learning algorithm that solves the XOR problem.

Figure 18.12: A classification problem that solves the XOR problem.

The main problem is learning. The knowledge representation system employed with our approach of representing concepts in conjunction with their connections to other concepts, i.e., the hierarchic and the hierarchical structures of the propositional representation. The use of propositional representations allows the hierarchical structures of the propositional representation to be represented in a high-level language. The knowledge representation system employed with our approach of representing concepts in conjunction with their connections to other concepts, i.e., the hierarchic and the hierarchical structures of the propositional representation. The use of propositional representations allows the hierarchical structures of the propositional representation to be represented in a high-level language.

The influence of previous work on the development of the knowledge representation system employed with our approach of representing concepts in conjunction with their connections to other concepts, i.e., the hierarchic and the hierarchical structures of the propositional representation. The use of propositional representations allows the hierarchical structures of the propositional representation to be represented in a high-level language.

18.2.3 Multilayer Perceptrons

Figure 18.13: A multilayer perceptron that solves the XOR problem.
Figure 18.16: The sigmoid activation function of the perceptron.

Figure 18.15: Using a multilayer network to learn to classify handwritten digits.

The equation to the right of the figure shows the perceptron's output as a function of the inputs. The equation is:

\[ y = \frac{1}{1 + e^{-\theta x + \theta_0}} \]

where \( x \) is the input vector, \( \theta \) is the weight vector, and \( \theta_0 \) is the bias.

We now turn to backpropagation networks. The aim of a backpropagation network is to minimize the error in the output layer by adjusting the weights in the hidden layers through a process of gradient descent.

Figure 18.14: A multilayer network.

In classification, backpropagation networks are used to learn decision boundaries. The network adjusts the weights in the hidden layers to minimize the classification error, which is typically measured using cross-entropy loss.

Our problem is to find a network architecture that can learn to classify images of handwritten digits. We will use a multilayer perceptron with one hidden layer to solve this problem.
1. Compute the outputs of the units in the hidden layer demanded by the network.

2. Initialize the weights in the network. Each weight should be set randomly to a small value.

3. Initialize the activations of the neurons in the hidden layer, using forward propagation.

4. Propagate the activations from the units in the hidden layer to the units in the output layer.

5. Repeatedly go to step 4 until the error reaches a prescribed level.

6. Propagate the activations from the units in the output layer to the units in the hidden layer, using backward propagation.

7. Compute the outputs of the units in the hidden layer, using the weights and activations computed by the forward propagation.

8. Compute the errors in the hidden layer, using the weights and activations computed by the backward propagation.

9. Compute the input and output biases of the output layer.

10. Compute the input and output biases of the hidden layer.

11. Compute the output layer weights.

12. Compute the hidden layer weights.

13. Compute the output layer weights.

14. Compute the hidden layer weights.

15. Compute the output layer weights.

16. Compute the hidden layer weights.

17. Compute the output layer weights.

18. Compute the hidden layer weights.

19. Compute the output layer weights. If the performance is satisfactory, use the weights obtained. If not, go to step 2 and repeat the cycle.
18.2 Generalization

The network is trained and updated in a supervised fashion, the generalization is held to its test data.

However, new validation is added to the red-corresponding learning rate. This is not a single validation, but a combination of the red-corresponding validation rate. The network is not updated in the test data.

Equation 19.6 [P.619] [1961, P.619] is given by

\[ f(x) = \frac{1}{1 + e^{-x}} \]

Section 18.1.2 Learning in Neural Networks

The network is trained and updated in a supervised fashion, the generalization is held to its test data.

The network is not updated in the test data.

Equation 19.6 [P.619] [1961, P.619] is given by

\[ f(x) = \frac{1}{1 + e^{-x}} \]