therefore explain the key concepts of backpropagation, and explore the framework of automatic differentiation in detail.

Note that the term 'backpropagation' is used in the neural computing literature in a variety of different ways. For instance, a feed-forward architecture may be called a backpropagation network. Also the term 'backpropagation' is sometimes used to describe the end-to-end training procedure for a neural network including the gradient descent parameter updates. In this book we will use 'backpropagation' specifically to describe the computational procedure used in the numerical evaluation of derivatives such as the gradient of the error function with respect to the weights and biases of a network. This procedure can also be applied to the evaluation of other important derivatives such as the Jacobian and Hessian matrices.

8.1. Evaluation of Gradients

We now derive the backpropagation algorithm for a general network having arbitrary feed-forward topology, arbitrary differentiable nonlinear activation functions, and a broad class of error function. The resulting formulae will then be illustrated using a simple layered network structure having a single layer of sigmoidal hidden units together with a sum-of-squares error.

Many error functions of practical interest, for instance those defined by maximum likelihood for a set of i.i.d. data, comprise a sum of terms, one for each data point in the training set, so that

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w}). \tag{8.1}$$

Here we will consider the problem of evaluating $\nabla E_n(\mathbf{w})$ for one such term in the error function. This may be used directly for stochastic gradient descent, or the results could be accumulated over a set of training data points for batch or mini-batch methods.

8.1.1 Single-layer networks

Consider first a simple linear model in which the outputs y_k are linear combinations of the input variables x_i so that

$$y_k = \sum_i w_{ki} x_i \tag{8.2}$$

together with a sum-of-squares error function that, for a particular input data point n, takes the form

$$E_n = \frac{1}{2} \sum_k (y_{nk} - t_{nk})^2 \tag{8.3}$$