

Pattern Recognition and Feed-forward Networks

Christopher M. Bishop

Microsoft Research
7 J J Thomson Avenue,
Cambridge, CB3 0FB, U.K.
cmbishop@microsoft.com

<http://research.microsoft.com/~cmbishop>

In *The MIT Encyclopedia of the Cognitive Sciences* (1999)
R. A. Wilson and F. C. Keil (editors), MIT Press, 629–632.

A feed-forward network can be viewed as a graphical representation of parametric function which takes a set of input values and maps them to a corresponding set of output values (Bishop, 1995). Figure 1 shows an example of a feed-forward network of a kind that is widely used in practical applications. Nodes in the

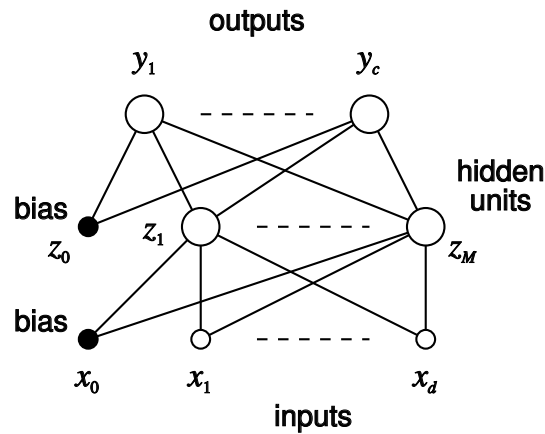


Figure 1: A feed-forward network having two layers of adaptive parameters.

graph represent either inputs, outputs or ‘hidden’ variables, while the edges of the graph correspond to the adaptive parameters. We can write down the analytic function corresponding to this network follows. The output of the j th hidden node is obtained by first forming a weighted linear combination of the d input values x_i to give

$$a_j = \sum_{i=1}^d u_{ji}x_i + b_j. \quad (1)$$

The value of hidden variable j is then obtained by transforming the linear sum in (1) using an activation function $g(\cdot)$ to give

$$z_j = g(a_j). \quad (2)$$

Finally, the outputs of the network are obtained by forming linear combinations of the hidden variables to give

$$a_k = \sum_{j=1}^M v_{kj}z_j + c_k. \quad (3)$$

The parameters $\{u_{ji}, v_{kj}\}$ are called *weights* while $\{b_j, c_k\}$ are called *biases*, and together they constitute the adaptive parameters in the network. There is a one-to-one correspondence between the variables and parameters in the analytic function and the nodes and edges respectively in the graph.

Historically feed-forward networks were introduced as models of biological neural networks (McCulloch and Pitts, 1943), in which nodes corresponded to neurons and edges corresponded to synapses, and with an activation function $g(a)$ given by a simple threshold. The recent development of feed-forward networks for pattern recognition applications has, however, proceeded largely independently of any biological modelling considerations.

The goal in pattern recognition is to use a set of example solutions to some problem to infer an underlying regularity which can subsequently be used to solve new instances of the problem. Examples include hand-written digit recognition, medical image screening and fingerprint identification. In the case of feed-forward networks, the set of example solutions (called a training set), comprises sets of input values together with corresponding sets of desired output values. The training set is used to define an error function in terms of the discrepancy between the predictions of the network, for given inputs, and the desired values of the outputs given by the training set. A common example of an error function would be the squared difference between desired and actual output, summed over all outputs and summed over all patterns in the training set. The learning process then involves adjusting the values of the parameters to minimize the value of the error function. Once the network has been trained, i.e. once suitable values for the parameters have been determined, new inputs can be applied and the corresponding predictions (i.e. network outputs) calculated.

The use of layered feed-forward networks for pattern recognition was widely studied in the 1960s. However, effective learning algorithms were only known for the case of networks in which at most one of the layers comprised adaptive interconnections. Such networks were known variously as perceptrons (Rosenblatt, 1962) and Adalines (Widrow and Lehr, 1990), and were seriously limited in their capabilities (Minsky and Papert, 1969). Research into artificial NEURAL NETWORKS was stimulated during the 1980s by the development of new algorithms capable of training networks with more than one layer of adaptive parameters (Rumelhart *et al.*, 1986). A key development involved the replacement of the non-differentiable threshold activation function by a differentiable non-linearity, which allows gradient-based optimization algorithms to be applied to the minimization of the error function. The second key step was to note that the derivatives could be calculated in a computationally efficient manner using a technique called ‘back-propagation’, so called because it has a graphical interpretation in terms of a propagation of error signals from the output nodes backwards through the network. Originally these gradients were used in simple steepest-descent algorithms to minimize the error function. More recently, however, this has given way to the use of more sophisticated algorithms, such as conjugate gradients, borrowed from the field of non-linear optimization (Gill *et al.*, 1981).

During the late 1980s and early 1990s, research into feed-forward networks emphasised their role as function approximators. For example, it was shown that a network consisting of two layers of adaptive parameters could approximate any continuous function from the inputs to the outputs to arbitrary accuracy provided the number of hidden units is sufficiently large and provided the network parameters are set appropriately (Hornik *et al.*, 1989). More recently, however, feed-forward networks have been studied from the much richer probabilistic perspective (see FOUNDATIONS OF PROBABILITY) which sets neural networks firmly within the field of *statistical pattern recognition* (Fukunaga, 1990). For instance, the outputs of the network can be given a probabilistic interpretation, and the role of network training is then to model the probability distribution of the target data, conditioned on the input variables. Similarly, the minimization of an error function can be motivated from the well-established principle of maximum likelihood which is widely used in statistics. An important advantage of this probabilistic viewpoint is that it provides a theoretical foundation for the study and application of feed-forward networks (see STATISTICAL LEARNING THEORY), as well as motivating the development of new models and new learning algorithms.

A central issue in any pattern recognition application is that of generalization, in other words the performance of the trained model when applied to previously unseen data. It should be emphasised that a small value of the error function for the training data set does not guarantee that future predictions will be similarly accurate. For example, a large network with many parameters may be capable of achieving a small error on the training set, and yet fail to model the underlying distribution of the data and hence achieve poor performance on new data (a phenomenon sometimes called ‘over-fitting’). This problem can be approached by limiting the complexity of the model thereby forcing it to extract regularities in the data rather than simply memorising the training set. From a fully probabilistic viewpoint, learning in feed-forward networks involves using the network to define a *prior* distribution over functions, which is converted to a *posterior* distribution once the training data have been observed. It can be formalised through the framework of BAYESIAN LEARNING, or equivalently through the MINIMUM

DESCRIPTION LENGTH approach (MacKay, 1992; Neal, 1996).

In practical applications of feed-forward networks, attention must be paid to the representation used for the data. For example, it is common to perform some kind of pre-processing on the raw input data (perhaps in the form of ‘feature extraction’) before they are used as inputs to the network. Often this pre-processing takes into consideration any prior knowledge we might have about the desired properties of the solution. For instance, in the case of digit recognition we know that the identity of the digit should be invariant to the position of the digit within the input image.

Feed-forward neural networks are now well established as an important technique for solving pattern recognition problems, and indeed there are already many commercial applications of feed-forward neural networks in routine use.

References

- Anderson, J. A. and E. Rosenfeld (Eds.) (1988). *Neurocomputing: Foundations of Research*. Cambridge, MA: MIT Press.
- Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.
- Fukunaga, K. (1990). *Introduction to Statistical Pattern Recognition* (Second ed.). San Diego: Academic Press.
- Gill, P. E., W. Murray, and M. H. Wright (1981). *Practical Optimization*. London: Academic Press.
- Hornik, K., M. Stinchcombe, and H. White (1989). Multilayer feedforward networks are universal approximators. *Neural Networks* **2** (5), 359–366.
- MacKay, D. J. C. (1992). A practical Bayesian framework for back-propagation networks. *Neural Computation* **4** (3), 448–472.
- McCulloch, W. S. and W. Pitts (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* **5**, 115–133. Reprinted in Anderson and Rosenfeld (1988).
- Minsky, M. L. and S. A. Papert (1969). *Perceptrons*. Cambridge, MA: MIT Press. Expanded Edition 1990.
- Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. Springer. Lecture Notes in Statistics 118.
- Rosenblatt, F. (1962). *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Washington DC: Spartan.
- Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland, and the PDP Research Group (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Volume 1: Foundations, pp. 318–362. Cambridge, MA: MIT Press. Reprinted in Anderson and Rosenfeld (1988).
- Widrow, B. and M. A. Lehr (1990). 30 years of adaptive neural networks: perceptron, madeline, and backpropagation. *Proceedings of the IEEE* **78** (9), 1415–1442.