Computation by neural networks

Geoffrey E. Hinton

Networks of neurons can perform computations that have proved very difficult to emulate in conventional computers. In trying to understand how real nervous systems achieve their remarkable computational abilities, researchers have been confronted with three major theoretical issues. How can we characterize the dynamics of neural networks with recurrent connections? How do the time-varying activities of populations of neurons represent things? How are synapse strengths adjusted to learn these representations? To gain insight into these difficult theoretical issues, it has proved necessary to study grossly idealized models that are as different from real biological neural networks as apples are from planets.

The 1980s saw major progress on all three fronts. In a classic 1982 paper[1], Hopfield showed that asynchronous networks with symmetrically connected neurons would settle to locally stable states, known as 'point attractors', which could be viewed as content-addressable memories. Although these networks were both computationally inefficient and biologically unrealistic, Hopfield's work inspired a new generation of recurrent network models; one early example was a learning algorithm that could automatically construct efficient and robust population codes in 'hidden' neurons whose activities were never explicitly specified by the training environment[2].

The 1980s also saw the widespread use of the backpropagation algorithm for training the synaptic weights in both feedforward and recurrent neural networks. Backpropagation is simply an efficient method for computing how changing the weight of any given synapse would affect the difference between the way the network actually behaves in response to a particular training input and the way a teacher desires it to behave[3]. Backpropagation is not a plausible model of how real synapses learn, because it requires a teacher to specify the desired behavior of the network, it uses connections backward, and it is very slow in large networks. However, backpropagation did demonstrate the impressive power of adjusting synapses to optimize a performance measure. It also allowed psychologists to design neural networks that could perform interesting computations in unexpected ways. For example, a recurrent network that is trained to derive the meaning of words from their spelling makes very surprising errors when damaged, and these errors are remarkably similar to those made by adults with dyslexia[4].

The practical success of backpropagation led researchers to look for an alternative performance measure that did not involve a teacher and that could easily be optimized using information that was locally available at a synapse. A measure with all the right properties emerges from thinking about perception in a peculiar way: the widespread existence of top-down connections in the brain, coupled with our ability to generate mental images, suggests that the perceptual system may literally contain a generative model of sensory data. A generative model stands in the same relationship to perception as do computer graphics to computer vision. It allows the sensory data to be generated from a high-level description of the scene. Perception can be seen as the process of inverting the generative model—inferring a high-level description from sensory data under the assumption that the data were produced by the generative model. Learning then is the process of updating the parameters of the generative model so as to maximize the likelihood that it would generate the observed sensory data.

Many neuroscientists find this way of thinking unappealing because the obvious function of the perceptual system is to go from the sensory data to a high-level representation, not vice versa. But to understand how we extract the causes from a particular image sequence, or how we learn the classes of things that might be causes, it is very helpful to think in terms of a top-down, stochastic, generative model. This is exactly the approach that statisticians take to modeling data, and recent advances in the complexity of such statistical models[5] provide a rich source of ideas for understanding neural computation. All the best speech recognition programs now work by fitting a probabilistic generative model.

If the generative model is linear, the fitting is relatively straightforward but can nevertheless lead to impressive results[6,7]. There is good empirical evidence that the brain uses generative models with temporal dynamics for motor control[8] (see also ref. 9, this issue). If the generative model is nonlinear and allows multiple causes, it can be very difficult to compute the likely causes of a pattern of sensory inputs. When exact inference is unfeasible, it is possible to use bottom-up, feedforward connections to activate approximately the right causes, and this leads to a learning algorithm for fitting hierarchical nonlinear models that requires only information that is locally available at synapses[10]. So far, theoretical neuroscientists have considered only a few simple types of nonlinear generative model. Although these have produced impressive results, it seems likely that more sophisticated models and better fitting techniques will be required to make detailed contact with neural reality.

The author is in the Gatsby Computational Neuroscience Unit, University College London, 17 Queen Square, London WC1N 3AR, UK.
e-mail: hinton@gatsby.ucl.ac.uk

1. Hopfield, J. J. *Proc. Natl. Acad. Sci. USA* **79**, 2554–2558 (1982).
2. Hinton, G. E. & Sejnowski, T. J. in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* Vol. 1 *Foundations* (eds. Rumelhart, D. E. & McClelland, J. L.) 282–317 (MIT Press, Cambridge, Massachusetts, 1986).
3. Rumelhart, D. E., Hinton, G. E. & Williams, R. J. *Nature* **323**, 533–536 (1986).
4. Plaut, D. C. & Shallice, T. *Cognit. Neuropsychol.* **10**, 377–500 (1993).
5. Cowell, R. G., Dawid, A. P., Lauritzen, S. L. & Spiegelhalter, D. J. *Probabilistic Networks and Expert Systems* (Springer, New York, 1999).
6. Bell, A. J. & Sejnowski, T. J. *Neural Comput.* **7**, 1129–1159 (1995).
7. Olshausen, B. A. & Field. D. J. *Nature* **381**, 607–609 (1996).
8. Wolpert, D. M., Ghahramani Z. & Jordan, M. I. *Science* **269**, 1880–1882 (1995).
9. Wolpert, D. M. & Ghahramani, Z. *Nat. Neurosci.* **3**, 1212–1217 (2000).
10. Hinton, G. E., Dayan, P., Frey, B. J. & Neal, R. *Science* **268**, 1158–1161 (1995).