Letters

Neural Networks—Then and Now

George Nagy

Abstract—**McCulloch and Pitts are often credited with the first (1943) mechanistic interpretation of the neuron doctrine. It was not until the 1960's, however, that neural networks emerged as a significant subdiscipline with attempts at application to engineering problems. Frank Rosenblatt, a Cornell University psychologist, showed by mathematical analysis, digital computer simulation, and experiments with special-purpose parallel analog systems that neural networks with variable weight connections could be *trained* to classify spatial patterns into prespecified categories. In his attempts to provide biologically plausible explanations of the function of the central nervous system, he investigated both relatively simple networks that were amenable to analysis and more complex networks whose behavior could be predicted only in terms of gross characteristics. He built up a sizable group of theoreticians, experimentalists, technologists, and, later, biologists. His work caught the imagination of the press and led to a wave of febrile activity that subsided at the end of that decade.**

I. Introduction

The application of adaptive neural networks to pattern recognition, 30 years ago, caused considerable stir in the technical community. A number of conferences were organized and some firms dedicated a significant share of their resources to the new technology. Now that neural networks constitute a well-established field of research, it is appropriate to look back and see just far we have come. This retrospective is traced from the point of view of an interested observer and is colored by the author's own experience as a graduate student (1960–1962) and postdoctoral research associate (1962–1963 and summer 1966) in the Cognitive Systems Research Program of Cornell University.

The Cognitive Systems Research Program was established in 1959 by Dr. Frank Rosenblatt. His first widely circulated technical report, which defined *perceptron*, was issued in 1957 under the aegis of the Cornell Aeronautical Laboratory [1]. As he himself ruefully admitted later, the term *perceptron* was a mistake: it suggested an automaton rather than the wide class of models of the central nervous system that he had intended. His main objective was to demonstrate analytically and experimentally that adaptive neural networks with a rich interconnectivity and synapselike nonlinearities could mimic many observed cognitive functions, and that the existence of such structures did not conflict with the available biological evidence. Rosenblatt's ideas were influenced by Bullock, Cahal, Clark and Farley, Culbertson, Eccles, Hayek, Kohler, Holland, Hubel and Wiesel, Lettvin and Maturana, McCulloch and Pitts, Milner, Penfield, Rashevsky, Rochester, Uttley and, perhaps most, by Oliver Hebb [2].

In spite of his overriding concern with the mind/brain problem, Rosenblatt was also interested in demonstrating the application of his networks to spatial pattern recognition and to discrete and continuous speech recognition. His group (which grew to about 20 persons, many of whom are still active in related fields) programmed an extensive and flexible simulator for the largest computer of that time. Nets with over 20 000 connections were exercised. The 1959-vintage Mark I alpha-perceptron is in the Smithsonian Institution. New storage elements were developed for Tobermory, a hybrid speech processor named after H. H. Monroe's (Saki) talking cat. A 1963 bibliography on perceptrons listed 98 publications [3].

Beginning in about 1964, Rosenblatt turned his attention to neurotransmitters and genetic coding. In addition to his professional interests, he was a skilled musician, an astronomer who built his own observatory, and a mountain climber. He died in a sailing accident in 1971 on his 43rd birthday.

II. Perceptrons

Rosenblatt's neural models are networks of three types of signal-processing units: *sensory* (input) units, *associative* units (now called hidden layers), and *response* (output) units. The output of each node depends only on the *sum* of its input signals. The models are classified according to the topology and type (fixed or variable-weight) of the interconnections, and the functional characteristics (linear, threshold, S-curve) of the nodes. The collection of all possible memory states, i.e., the configuration of values of the variable-weight connections, constitutes the *phase space* of the network. The coupling coefficients between all pairs of units are represented by a time-varying *interaction matrix*. A *perceptron* is then defined as a network of S, A, and R units whose interaction matrix depends *only* on the sequence of past activity states of the network [4].

In terms of network topology, Rosenblatt differentiated between *series-coupled* perceptrons (connections permitted only between successive layers of processing units), *cross-coupled* perceptrons (connections also between units in the same layer), and *back-coupled* perceptrons (with feedback paths). He justified the fixed, semirandom layer of connections between S units and A units on the grounds of its well-established presence in biological visual and auditory systems.

He also classified the rules governing the evolution of the interaction matrix through time. *Monopolar* reinforcement changes only the weights to units whose output was strictly positive; *bipolar* reinforcement is not restricted. In *alpha-system* reinforcement, the weights are changed by a constant value, while with the conservative *gamma-system*, the total increment is kept to zero. Proportional reinforcement systems were also investigated. Reinforcement could be triggered only by certain stimuli, certain responses, or, in *error-correction* mode, by a combination of both input and output.

For series-coupled networks with binary inputs and a single layer of variable weights, Rosenblatt and his colleagues (particularly H. D. Block, Professor of Applied Mechanics and Mathematics at Cornell, who died in 1978) proved that if a dichotomy can be achieved with *any* set of weights, the error-correcting alpha-reinforcement procedure will produce some set of solution weights after a finite number of iterations. Dozens of different proofs of this *perceptron convergence theorem* have been published, but all of the bounds require knowledge of a solution vector. Nevertheless, the procedure remains the most efficient test for linear separability. The many attempts that have been made to improve the rate of convergence by adjusting the order of presentation of input patterns and the size of the increments have not yielded any guaranteed improvement.

Manuscript received September 24, 1990.

The author is with the Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, NY 12180-3590.

IEEE Log Number 9042022.

1045-9227/91/0300-0316$01.00 © 1991 IEEE

Rosenblatt proved the existence of *universal perceptrons:* a network with one variable and one fixed layer of weights, with a large enough set of A units, can be trained for *any* dichotomy. He understood, however, that such machines have limited abilities to *generalize* to patterns similar to those in the training set. The fact that nontrivial topological properties are beyond the reach of single-layer machines was conclusively demonstrated in 1967 by Marvin Minsky (Rosenblatt's former schoolmate) and Seymour Papert in a monograph that marked the end of the first halcyon era of neural networks [5]. The current expanded edition of this influential text contains an epilogue addressing some of the controversies that have surrounded neural networks from the very beginning. Minsky's own 1954 PhD. dissertation at Princeton University was titled "Neural Nets and the Brain Model Problem" and many contemporary students of artificial intelligence mistakenly credit him with the invention of perceptrons.

Noting the accumulated experimental evidence that human performance on coherent patterns was far superior to performance on random patterns, Rosenblatt investigated configurations of connections that duplicated this phenomenon. He also showed that generalization over arbitrary groups of transformations, such as translation or rotation, not only can be wired in but also can be *learned* by cross-coupled perceptrons from sequences of patterns with equivalent classes in temporal proximity [6]. He derived some equilibrium conditions for fully cross-coupled networks (now known as *Hopfield nets*). He analyzed back-coupled systems where the weights were changed depending on the activity of the nodes at consecutive time intervals but was not able to derive a workable reinforcement regime for series-coupled perceptrons with multiple layers of variable weights. Such regimes have since been devised, but there is still no guarantee that they will obtain a solution if one exists, in the sense that a single-layer system can.

Beginning in 1963, Rosenblatt turned his attention to explaining how sequences of sensory experiences can be stored and recalled over periods comparable to a human lifetime. His goal was to model phenomena such as selective recall, retention of originally "unnoticed" events, transient and permanent forgetting, recovery from retrograde amnesia, and effects of localized lesions and electrical stimulation in aphasia, agnosia, and related disorders. His model consisted, in addition to the usual S units, cross-coupled A units, and R units, of a clocking system with reinforceable weights to the associative system. To recall a string of experiences starting with an arbitrary event, the associative network would induce the state corresponding to that event in the clocking network, which would then cycle through a sequence of states, each of which would trigger the corresponding event in the associative system. Recondite mathematical analysis indicated that with a number of units comparable to the estimated number of neurons in the brain, very long sequences could be accurately recalled [7]. However, no convincing simulation experiments were ever conducted.

III. Simulation and Hardware

An elaborate perceptron simulation program for IBM 7090/94 systems was developed [8]. The command language allowed specification of families of complex random or deterministic networks, the generation of elaborate stimulus (input pattern) sets, a variety of training and testing procedures, flexible output reports, repetition of experiments for a range of parameters, signal-propagation delays, and exponential decay over time of the weights (*forgetting*). Simulations were limited only by the available computer memory (32K words). The simulator was used to optimize parameters in the application of neural networks to alphabetic character recognition; particle tracks in bubble-chamber photographs; phoneme, isolated word, and continuous speech recognition; speaker verification; and center-of-attention mechanisms for image processing.

From the very beginning, Rosenblatt insisted on parallel, analog implementations of his models for large-scale experiments. The Mark I perceptron, built at the Cornell Aeronautical Laboratory in 1958, had a retina of 20 × 20 photocells (S units), 512 stepping motors that controlled potentiometers (the weights of the A–R connections), and eight R units [9]. Replaceable plug boards, similar to those then used to "program" I/O devices, were used for the S–A connections; each S unit could be connected to up to 40 A units.

A number of faster analog storage devices were also investigated. In all of these devices the output current or voltage was proportional to their setting and could be summed in parallel using some impedance element. The *memistor*, an electrochemical integrating device, was developed by Bernard Widrow at Stanford University for adaptive pattern recognition and control systems. *Thermistors*, *photochromic devices*, *solions*, and *transpolarizers* were also considered. However, magnetic flux integration appeared most promising. A number of magnetic storage devices with stable and linear integrating characteristics, as well as new reinforcement procedures, were developed by Charles Rosen and his group at the Stanford Research Institute. Rosenblatt collaborated closely with this group (which included Ted Brain, George Forsen, Richard Duda, and Nils Nilsson), with other groups at Astropower (R. D. Joseph, P. M. Kelly, and S. S. Viglione) and Aeronutronics (J. K. Hawkins and C. J. Munsy), and with scientists and engineers in Germany and the Soviet Union.

The four-layer Tobermory perceptron, designed and built at Cornell University between 1961 and 1967, had 45 S units, 1600 A1 units, 1000 A2 units, and 12 R units. Intended for speech recognition, the input section consisted of 45 band-pass filters attached to 80 difference detectors, with the output of each detector sampled at 20 time intervals. Its 12 000 weights consisted of toroidal cores capable of storing over 100 different amplitudes. Each A2 unit could be connected to any of 20 A1 units by means of a wall-sized plugboard. As has happened with so many other projects in the last three decades, by the time Tobermory was completed, the technology of commercial Von Neumann computers had advanced sufficiently to outperform the special-purpose parallel hardware.

IV. Biology

In the mid-1960's Rosenblatt's interest shifted to the biological basis of learning. He attempted to duplicate with maze-trained rats the reported transfer of learning in worms by means of homogenized brain extracts. He also supervised a number of Ph.D. students who investigated the role of DNA in memory.

V. Conclusions

Inspired partly by Rosenblatt's results (many of which were sensationalized by the media), large companies established groups dedicated to applying neural networks to practical problem domains. A number of small start-up companies also set their sights in that direction. There was considerable interest from the military: the Cognitive Systems Research Program was generously funded by the Office of Naval Research. Expectations were high, and hyperbole on self-organizing systems, including those modeled on genetic evolution, abounded. A number of large hardware development projects were undertaken in the hope that the few remaining bugs in the learning mechanisms would be ironed out by the time they were completed.

The author cannot help but view the recent buildup of enthusiasm with pleasure and some nostalgia, but also with trepidation that past excessive expectations will again be raised, and again disappointed. As Santayana reminds us, "Those who cannot remember the past are condemned to repeat it."

References

[1] F. Rosenblatt, "The perceptron, A perceiving and recognizing automation," Cornell Aeronautical Laboratory Report No. 85-460-1, Jan. 1957.

[2] O. Hebb, *The Organization of Behavior*. New York: Wiley, 1949.
[3] F. Rosenblatt, "A bibliography of perceptron literature," in *Collected Technical Papers*, vol. 2, *Cognitive Systems Research Program*, Report no. 4, Cornell University, July 1963.
[4] F. Rosenblatt, *Principles of Neurodynamics*. East Lansing, MI: Spartan Books, 1962.
[5] M. Minsky and S. Papert, *Perceptrons, an Introduction to Computational Geometry*. Cambridge, MA: MIT Press, 1st ed. 1969, expanded ed., 1988.
[6] F. Rosenblatt, "Perceptual generalization over transformation groups," in *Self-Organizing Systems*, (Yovits and Cameron, Eds.) Elmsford, NY: Pergamon Press, 1960.
[7] F. Rosenblatt, *A Model for Experiential Storage in Neural Networks* (Computer and Information Sciences, Tou and Wilcox, Eds.). East Lansing, MI: Spartan Books, 1964.
[8] T. H. Barker, "A computer program for simulation of perceptrons and similar neural networks, *Cognitive Systems Research Program*, Report no. 8, Cornell University, July 1966.
[9] H. C. Hay and C. W. Wightman, "The Mark I perceptron, design and performance," in *IRE Nat. Convention Rec.*, part 2, 1960.

Adaptive Nearest Neighbor Pattern Classification

Shlomo Geva and Joaquin Sitte

Abstract—**We describe a variant of nearest neighbor pattern classification (NN) [1] and supervised learning by learning vector quantization (LVQ) [2], [3]. The decision surface mapping method, which we call DSM, is a fast supervised learning algorithm, and is a member of the LVQ family of algorithms. A relatively small number of prototypes are selected from a training set of correctly classified samples. The training set is then used to adapt these prototypes to map the decision surface separating the classes. This algorithm is compared with NN pattern classification, learning vector quantization (LVQ1) [2], and a two-layer perceptron trained by error backpropagation [4]. When the class boundaries are sharply defined (i.e., no classification error in the training set) the DSM algorithm outperforms these methods with respect to error rates, learning rates, and the number of prototypes required to describe class boundaries.**

I. Introduction

The nearest neighbor (NN) method assigns an unclassified sample vector to the class of the nearest of a set of correctly classified prototypes, or codebook vectors. Cover and Hart have shown that in a large sample, the error of this rule is bounded above by twice the Bayes probability of error [1].

Learning vector quantization (LVQ1, LVQ2, LVQ2.1, and LVQ3), described by Kohonen [2], [3], is a nearest neighbor classification method in which a fixed number of prototype vectors are progressively modified to cover the input space. The LVQ family of algorithms is concerned with optimal placement of these prototypes, so as to reflect the probability distribution of the training samples. The adaptive decision surface mapping (DSM) algorithm is a variation of the LVQ method, but we have dropped the requirement that the prototypes reflect the probability distribution of the classes. Instead, the algorithm adapts the prototype vectors to closely map the decision surface separating classes. DSM is described in detail in the next section. In Section III we present comparative results for three classification problems, which indicate a drastic performance improvement over the LVQ and backpropagation.

Manuscript received October 5, 1990.

The authors are with the Faculty of Information Technology, Queensland University of Technology, GPO Box 2434, Brisbane, Queensland, 4001 Australia.

IEEE Log Number 9042019.

II. Adaptive Decision Surface Mapping

The DSM algorithm starts by selecting a small subset of prototypes from the training set. The initial prototypes are selected at random. We have found that good results are obtained when the proportion of prototypes from each class in the initial subset matches the *a priori* probabilities of the classes, and it is indeed the procedure commonly followed with LVQ. This information is usually available in the training set, when random sampling is used, or may have to be provided externally if the training set does not reflect the *a priori* probabilities of the classes.

In the learning stage, the training set is used to modify the prototypes in order to gradually adapt the decision surface they define to that defined by the entire training set and reduce the classification error rate. Samples from the training set are cyclically or randomly presented for classification. When a training sample is correctly classified, that is, the training sample is of the same class as the nearest prototype, no modifications are applied. When misclassification occurs, modifications take place to apply both punishment and reward.

The punishment step takes the nearest neighbor prototype, which, in this case, is of the wrong class, and moves it away from the training sample, along the line connecting the two vectors

$$\vec{m}_w(t+1) = \vec{m}_w(t) - \alpha(t)\left[\vec{x}(t) - \vec{m}_w(t)\right]. \quad (1)$$

The reward step searches for the nearest correct prototype and moves it towards the training sample, along the line connecting the two vectors

$$\vec{m}_c(t+1) = \vec{m}_c(t) + \alpha(t)\left[\vec{x}(t) - \vec{m}_c(t)\right]. \quad (2)$$

The term α is a scalar gain factor, monotonically decreasing with time. For the cases discussed below, we have found that very good results are obtained when α starts from a value of 0.3 or less, and linearly decreases to 0, at a rate consistent with the desired training limit (number of presentations). The algorithm is not very sensitive to initial values of α, but if alpha starts too small, training takes longer.

In the earlier stages of the training process α is relatively large; therefore the process is allowed to rapidly modify prototypes to remove large classification errors caused by the initial conditions. In later stages, as α decreases, a more refined adaptation takes place to correct smaller classification errors or to arrive at a compromise configuration where errors are minimized.

The algorithm modifies prototypes only on misclassification, and since errors are more likely to occur with samples near class boundaries, it rearranges prototypes, in pairs, on each side of a class boundary, to correct or at least reduce the magnitude of these errors.

It is possible that a configuration eliminating all classification errors on the training set could be arrived at before α reaches a value of 0. In that instance training is complete.

DSM is different from all the variants of LVQ. In LVQ1 modifications are applied at each presentation, either to punish an incorrect classification or to reward a correct one. LVQ2 modifies the nearest and next-to-nearest neighbors whenever the nearest neighbor is of a different class, and the next nearest neighbor is of the same class, as the training sample. Furthermore, LVQ2 requires the training vector to fall within a window which is determined by the relative distances of the training sample from the pro-

1045-9227/91/0300-0318$01.00 © 1991 IEEE