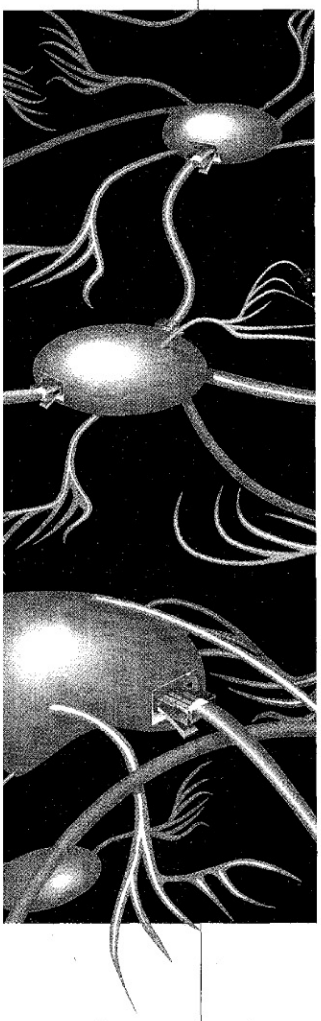

Neurocomputing

Motivation, Models, and Hybridization

Sankar K. Pal
Indian Statistical Institute

Pradip K. Srimani
Colorado State University

Several novel modes of computation have recently emerged that are collectively known as *soft computing*. The raison d'être of this mode is to exploit the tolerance for imprecision and uncertainty in real-world problems to achieve tractability, robustness, and low cost. Soft computing is usually used to find an approximate solution to a precisely (or an imprecisely) formulated problem. Neurocomputing, with its artificial neural networks, is one of the major components of this approach.

Although this emerging technology is rooted in various disciplines, the concept of artificial neural networks was inspired by biological neural networks. Biological neurons, believed to be the structural constituents of the brain, are much slower than silicon logic gates. But inferencing in biological neural networks is faster than the fastest computer. The brain compensates for the relatively slower operation by having an enormous number of massively interconnected neurons. A biological neural network is a nonlinear, highly parallel device characterized by robustness and fault tolerance. It also can

- learn by adapting its synaptic weights to changes in the surrounding environment;
- handle imprecise, fuzzy, noisy, and probabilistic information; and
- generalize from known tasks or examples to unknown ones.

Artificial neural networks (ANNs) are an attempt to mimic some—or all—of these characteristics.[1-3] This soft computational paradigm differs from a programmed instruction sequence in that information is stored in the synaptic connections. Each neuron is an elementary processor with primitive operations, like summing the weighted inputs coming to it and then amplifying or thresholding the sum. Even a synchronous assembly of McCulloch-Pitts neurons can, in principle, perform universal computations for suitably chosen weights. Such an assembly can perform the same computations as an ordinary digital computer.

A neural network is characterized by the network topology, the connection strength between pairs of neurons (weights), node properties, and the status-updating rules. The updating or learning rules control weights and/or states of the processing elements (neurons). Normally, an objective function is defined that represents the complete status of the network, and its set of minima corresponds to different stable states of the network.

There are three broad paradigms of learning: supervised, unsupervised (or self-organized), and reinforcement.[2] (Reinforcement is sometimes viewed as a special case of supervised learning.) Each category has many algorithms.

In supervised learning (learning with a teacher), adaption occurs when the system directly compares the network output with a known correct or desired answer. In unsupervised learning, the network is tuned to the statistical regularities of the input data so that it can form categories by optimizing—with respect to the network's free parameters—a task-independent measure of the quality of the net's category representation. Reinforcement learning, on the other hand, attempts to learn the input-output mapping through trial and error with a view to maximizing a performance index called the reinforcement signal. The system knows whether the output is correct or not, but does not know the correct output.

POPULAR NEURAL NETWORKS

ANNs have become a technical folk legend. The market is flooded with new, increasingly technical software and hardware products, and many more are sure to come. Among the most popular hardware implementations are the Hopfield, multilayer perceptron, Self-Organizing Feature Map, Learning Vector Quantization, radial basis function, cellular neural, and Adaptive Resonance Theory (ART) networks.

Hopfield network

This recurrent net is completely connected. It acts as a nonlinear associative memory that can retrieve an internally stored pattern when presented with an incomplete or noisy version of that pattern. It can also be used as an optimization tool. The status of any neuron can be updated a random number of times—independent of other neurons, but in parallel. Because all neurons interact, the collective property inherently reduces the computational task.

In the Boltzmann Machine, a generalization of the Hopfield network, operation is based on a concept of statistical thermodynamics known as simulated annealing.[3] The Hopfield network, the Boltzmann Machine, and a derivative known as the Mean-Field-Theorem machine have been used in applications such as image segmentation and restoration, combinatorial optimization (the Traveling Salesman Problem and graph partitioning), in addition to their use as content-addressable memory.

Multilayer perceptron network

The *perceptron* concept was one of the most exciting developments during the early days of pattern recognition.[2] It is a network of elementary processors (arranged in a manner reminiscent of biological neural nets) that can learn to recognize and classify patterns autonomously. The processors are simple elements arranged in one layer. This classical single-layer perceptron, given two classes of patterns, attempts to find a linear decision boundary separating the two classes. If the two sets of patterns are linearly separable, the perceptron algorithm is guaranteed to find a separating hyperplane in a finite number of steps. However, if the pattern space is not linearly separable, the perceptron fails.

A single-layer perceptron is obviously inadequate for situations with multiple classes and nonlinear separating boundaries—hence the invention of the multilayer perceptron network (MLP). Its layers are completely connected but lack internal node connection. The MLP uses supervised learning, implemented in two phases. In the forward phase, the network node output is computed; in the backward one, weights are adjusted to minimize the error between the observed and desired outputs. An MLP is believed to have generalization capability,[2] that is, it can produce correct (or nearly correct) output for input not used during training. The net can be viewed as a nonlinear input-output mapping, and the learning process can be seen as fitting a function to the given data set. An MLP performs interpolation well, since the continuous activation functions produce continuous output functions. Given a data set, however, an MLP can pick up one of many possible generalizations corresponding to different minima—and it may not be the desirable one. And because learning involves searching over a complex space, it is often time-consuming.

The MLP has been applied to many applications, ranging from classifier design, function approximations, and speech identification to scene analysis and military target identification.

Self-Organizing Feature Mapping (SOFM)

This unsupervised learning network[4] transforms p-dimensional input patterns to a q-dimensional (usually $q = 1$ or 2) discrete map in a topologically ordered fashion. Input points that are close in the p-dimension are also mapped closer on the q-dimensional lattice. Each lattice cell is represented by a neuron associated with a p-dimensional adaptable weight vector. The match between each weight vector is computed with every input. The best matching weight vector and some of its topological neighbors are then adjusted to better match the input points. Such networks have been used for generating semantic maps, clustering, phonetic typewriters, and graph bipartitioning.

In a special case of SOFM, the Learning Vector Quantization (LVQ) network,[4] only the weight vector associated with the winner node is updated with every data point. Such a learning scheme, where all nodes compete to win, is called *competitive*. It is essentially a clustering network that does not preserve topological order. Its main uses are for clustering and image data compression.

ANNs have become a technical folk legend. The market is flooded with new, increasingly technical software and hardware products.

Adaptive Resonance Theory network

In a competitive learning scheme, there is no guarantee that the clusters formed will be stable unless the learning rate gradually approaches zero with iteration. But when this happens, the network loses its plasticity. The Adaptive Resonance Theory (ART) net[2] overcomes this dilemma. In ART, a weight vector (prototype of a category) is adapted only when the input is sufficiently similar to that of the prototype, that is, when the input and a prototype resonate. When an input is not sufficiently similar to any prototype, a new category is formed using the input as the prototype. The condition "sufficiently similar" is checked by using a vigilance parameter. ART1 is designed for 0/1 input; ART2 is for continuous valued input.

Radial basis function network

In this network,[1] the output nodes linearly combine the basis functions computed by the hidden-layer nodes. Such a network is also known as a localized receptive field network because the hidden layer nodes produce localized responses to the input signals. The most commonly used basis function is the Gaussian kernel. Like the MLP, RBF networks can be used for both classifier design and function approximation, and they can make an arbitrary approximation to any continuous nonlinear mapping. The main difference between MLP and RBF nets lies in the basis functions used by the hidden-layer nodes in that RBF nets use Gaussian while MLP nets use sigmoidal functions.

Note that the choice between RBF and MLP depends on the problem at hand.

Cellular neural network

This network architecture[5] is similar to that of a cellular automata. It is an assembly of neurons or cells in which each node is connected only to its neighbor cell. A neighbor cell is not necessarily an adjacent cell; this depends on the definition of the neighborhood. A CNN has both output feedback and input control mechanisms. Asynchronous processing, continuous time dynamics, and local interactions between cells are some key features.

OTHER APPROACHES

Other new research areas have revealed that ANN-based models and systems can be improved by integrating their merits with those of other emerging theories and technologies, such as fuzzy logic and genetic algorithms.[1] For example, the time-delay network is good for modeling systems in which the output has a finite temporal dependency on the input. There are other families of networks for principal component analysis, mixed category perception, and character recognition. Integrating neural networks with other soft-computing tools such as fuzzy sets and genetic algorithms also provides a much more attractive and stronger computational paradigm for solving complex and computationally expensive problems. Two such emerging paradigms are called *neurofuzzy* and *neurogenetic*.

Fuzzy sets

Fuzzy sets[6] model human reasoning and/or thinking in that they can handle imprecise information. On the other hand, ANNs attempt to model how the brain works. A judicious integration of the two approaches may thus result in more intelligent systems (in terms of parallelism, fault tolerance, adaptivity, and uncertainty management) that function more the way humans do. Fuzzification can be implemented at the input, output-decision, learning-strategy, and neuronal levels of networks. Fuzzy algorithms can also be integrated.

Genetic algorithms

Genetic algorithms,[7] another biologically inspired evolutionary process, provide an adaptive, robust, parallel, and randomized searching technique in which a population of solutions evolves over a sequence of generations to a globally optimal solution. Based on a fitness function, good solutions are selected for reproduction using two genetic recombination operators: crossover and mutation,

In a neurogenetic system, the two components may interact in many ways to overcome a neural network limitation.[8,9] For example, a genetic algorithm can be used to select the cloning templates of a cellular neural network. It can also help avoid the tedious back-propagation algorithm for an MLP.

These two hybrid paradigms will not only continue to remain prominent research areas for the coming decade but will also play a key role in the development of future technology, including sixth-generation computing systems.

IN THIS ISSUE

This theme issue of *Computer* seeks to provide adequate information about the status of theory and applications to readers who have little or no knowledge of neural networks. We present six articles covering various aspects—but definitely not all facets—of neural networks. Readers interested in the more computational aspects can consult the Spring 1996 issue of *IEEE Computational Science & Engineering* (see the sidebar).

The tutorial article by Jain et al. introduces various popular neural network models (computations, architectures, and learning algorithms) along with the motivation behind developing neural networks. It also discusses applying neural networks to character recognition. The authors have provided some interesting tables and diagrams to bring out the networks' characteristics and distinguishing features.

Supervised learning of weights in a feed-forward neural network can be viewed as a nonlinear optimization problem. The article by Shang and Wah studies various global optimization methods for supervised learning. They discuss the merits and demerits of these methods, and they propose NOVEL, a global minimization method for general nonlinear optimization problems.

Sequential machines are not suitable for real-life applications of neural networks. General-purpose parallel machines and neurocomputers that implement a particular model directly in hardware are much better platforms for neural network applications. Šerbedžija's article discusses the value of both approaches to neural simulations, clearly distinguishing between various parallelization techniques as they relate to nodes, training schemes, and weights.

Tan et al. propose a neural logic network named Neulonet for modeling the human decision-making process. Apart from the parallel and adaptive nature of the connectionist architecture, Neulonet distinguishes itself by expressing various logical operations. The authors have also developed a hybrid system named the Neulonet shell that combines neural and rule-based systems. The shell's usefulness and superiority (over pure rule-based or pure neural network systems) have been established for financial applications.

Setiono and Liu describe the Neurorule algorithm for generating symbolic rules from a trained three-layer feed-forward network. The trained net is pruned by removing redundant weights and then the activation of the hidden units is discretized by clustering. These discretized activation values are used to generate symbolic classification rules. Neurorule performance not only compares favorably with that of decision trees but also provides better comprehension of the activities in a feed-forward network.

Spert-II, a vector microprocessor system proposed by Wawrzynek et al., is an Sbus card for Sun-compatible workstations. The authors have integrated multiple fixed-point data paths with a high-bandwidth memory system, resulting in cost-effective network training. Spert-II provides an environment closely resembling a conventional workstation. The authors have developed many vector library routines for several applications.

SINCE THERE IS NO ARTICLE ON HYBRIDIZATION of artificial neural networks, we have included key references in the area of neurofuzzy and neurogenetic computing for the convenience of readers. ■

Fuzzy sets and genetic algorithms will play a key role in the development of future technology, including sixth-generation computing systems..

Acknowledgments

We are grateful to the former and current editors-in-chief of *Computer*, Ted Lewis and Ed Parrish, respectively, for their guidance through the whole process. We thank the more than 50 groups of authors who submitted manuscripts for this theme issue, and we gratefully acknowledge the efforts of the more than 100 reviewers who spent their valuable time to provide timely reviews. Nikhil R. Pal's and Indranil Dutta's assistance in completing the editing process is also gratefully acknowledged.

References

1. D.R. Hush and B.G. Horne, "Progress in Supervised Neural Networks," *IEEE Signal Processing Magazine*, Vol. 10, No. 1, Jan. 1993, pp. 8-38.
2. J. Hertz, A. Krogh, and R.G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Reading, Mass., 1991.
3. J.A. Anderson and E. Rosenfeld, eds., *Neuro Computing Foundations of Research*, MIT Press, Cambridge, Mass., 1988.
4. T. Kohonen, *Self-Organization and Associative Memory*, Springer-Verlag, Berlin, 1989.
5. L.O. Chua and L. Yang, "Cellular Neural Networks: Theory," *IEEE Trans. Circuits and Systems*, Vol. 35, No. 10, Oct. 1988, pp. 1,257-1,272.
6. G.J. Klir and B.Yuan, *Fuzzy Sets and Fuzzy Logic, Theory and Applications*, Prentice-Hall, Englewood Cliffs, N.J., 1995.
7. D.E. Goldberg, *Genetic Algorithms: Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, Mass., 1989.
8. H. Muhlenbein, "Limitations of Multilayer Perceptron Networks—Steps Towards Genetic Neural Networks," *Parallel Computing*, Vol. 14, 1990, pp. 249-260.
9. D. Murray, "Tuning Neural Networks with Genetic Algorithms," *AI Expert*, June 1994, pp. 27-31.

Further reading

Basak, J., and S.K. Pal, "X-tron: An Incremental Connectionist Model for Category Perceptron," *IEEE Trans. Neural Networks*, Vol. 6, No. 5, 1995, pp. 1,091-1,108.

Berenji, H.R., and P. Khedkar, "Learning and Tuning Fuzzy Logic Controllers through Reinforcements," *IEEE Trans. Neural Networks*, Vol. 3, No. 5, May 1992, pp. 724-740.

Bezdek, J.C., and N.R. Pal, "A Note on Self-Organizing Semantic Map," *IEEE Trans. Neural Networks*, Vol. 6, No. 6, pp. 1,029-1,036.

Fukushima, K., "Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position," *Biological Cybernetics*, Vol. 36, No. 4, 1980, pp. 193-202.

Gallant, S.I., "Connectionist Expert Systems," *Comm. ACM*, Vol. 31, No. 2, Feb. 1988, pp. 152-169.

Ghosh, A., N.R. Pal, and S.K. Pal, "Self-Organization for Object Extraction Using Multilayer Neural Network and Fuzziness Measures," *IEEE Trans. Fuzzy Systems*, Vol. 1, No. 1, Feb. 1993, pp. 54-68.

Kosko, B., *Neural Networks and Fuzzy Systems*, Prentice-Hall, Englewood Cliffs, N.J.,1992.

Keller, J.M., and H. Tahani, "Implementation of Conjunctive and Disjunctive Fuzzy Logic Rules with Neural Networks," *Int'l J. Approximate Reasoning*," Vol. 6, No. 2, 1992, pp. 221-240.

Lang, K.J., A.H. Waibel, and G.E. Hinton, "A Time-Delay Neural Network Architecture from Isolated Word Recognition," *Neural Networks*, Vol. 3, No. 1, 1990, pp. 23-44.

Mitra, S., and S.K. Pal, "Neuro-Fuzzy Expert Systems: Overview with a Case Study," in *Fuzzy Reasoning in Information, Decision, and Control Systems*, S.C. Tzafestas and A.N. Venetsanopoulos., eds., Kluwer Academic Publishers, Boston/ Dordrecht, 1994.

Pal, S.K., and S. Mitra, "Multilayer Perceptron, Fuzzy Sets, and Classification," *IEEE Trans. Neural Networks*, Vol. 3, No. 5, May 1992, pp. 683-697.

Special Issue on Fuzzy Systems, *IEEE Trans. Neural Networks*, Vol. 3, No. 5, 1992.

Sankar K. Pal *is a professor and the founding head of the Machine Intelligence Unit of the Indian Statistical Institute in Calcutta. His research interests include pattern recognition, image processing, neural nets, genetic algorithms, and fuzzy sets and systems.*

Pal is a member of the editorial board of IEEE Transactions on Neural Nets, IEEE Transactions on Fuzzy Systems, International Journal of Approximate Reasoning, *and* Far-East Journal of Mathematical Sciences. *He is coauthor and coeditor of two books on fuzzy systems.*

Pal is a fellow of the IEEE; the Indian National Science Academy; the National Academy of Sciences, India; IETE; and the Institute of Engineers, India.

Pradip K. Srimani *is a professor in the Department of Computer Science at Colorado State University in Ft. Collins, where he also serves as acting chair. His research interests include distributed computing, parallel algorithms, fault-tolerant computing, networks, and graph theory applications.*

Srimani is the associate editor-in-chief of IEEE Computer Society Press and is a member of the editorial boards of IEEE Software *and* International Journal of Simulation. *He also serves as Technical Committee Chair of the IEEE Computer Society Education Activities Board. He is coeditor of two books on software reliability and distributed mutual exclusion algorithms. He is a senior member of IEEE and a member of ACM.*

Readers can reach Pradip K. Srimani at the Department of Computer Science, Colorado State University, Ft. Collins, CO 80523; e-mail srimani@cs.colostate.edu.