Knowledge Discovery Based on Neural Networks

The intelligence emerging from interactions among numerous self-organizing processing elements can be trained to discover the knowledge embedded in data.

LiMin Fu

Traditionally, human experts have derived their knowledge from their own personal observation and experience. With advancing computer technology, automated knowledge discovery has become an important AI research topic, as well as a practical business application in an increasing number of organizations. Knowledge discovery can be defined as the learning of implicit and previously unknown nontrivial knowledge from data or observations.

ANGELA WILCZYNSKI/WIZZLE.SIMPLENET.COM/FRACTALS/FRACTALINTRO.HTM

In symbolic AI, learning is often viewed as searching in a defined hypothesis space that typically grows exponentially with the size of the problem to be solved. A practical learning method has to rely on heuristics and bias to avoid an exhaustive and impractical search in large problem domains. As heuristics often lack a solid mathematical basis, the solution found is not guaranteed to be the best one possible. The symbolic approach is also weak in dealing with data noise, inconsistency, and uncertainty.

The neural network approach has emerged as a promising alternative for knowledge discovery applications. Inspired by biological neural networks, it assumes that intelligence emerges from interactions among a large number of simple processing elements. Biological neurons transmit electrochemical signals through neural pathways. Each neuron receives signals from other neurons through special junctions called synapses. Some inputs tend to excite the neurons, others to inhibit them. When the cumulative effect exceeds a threshold, the neuron fires and sends a signal to other neurons. An artificial neuron created by AI software engineers models these simple biological characteristics. Each artificial neuron receives a set of inputs; each input is multiplied by a weight analogous to a synaptic strength.

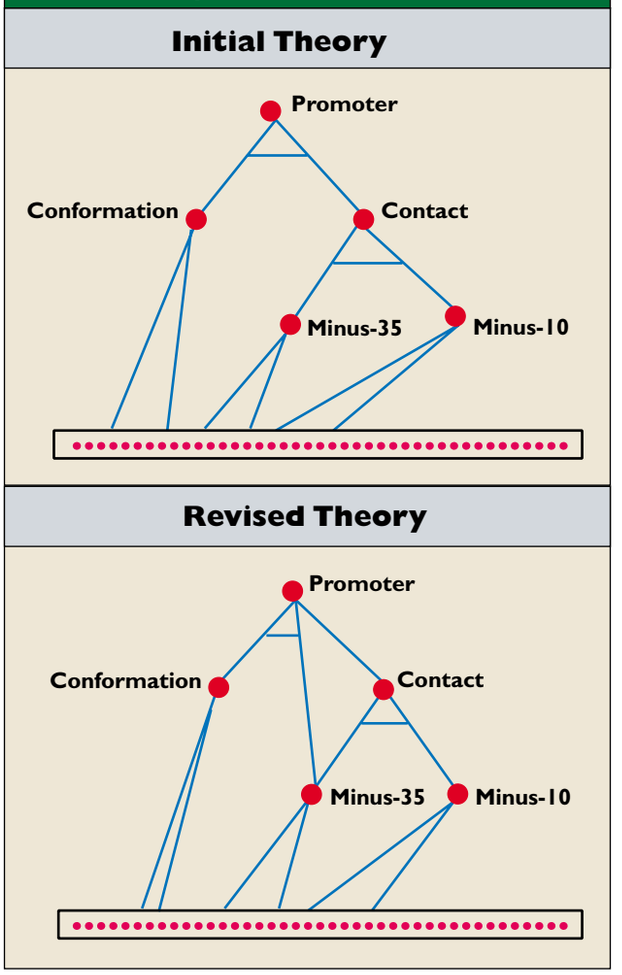
An artificial neural network is represented by a set of nodes and arrows. A node corresponds to a neuron; an arrow corresponds to a connection, along with the direction of signal flow between neurons. Some nodes are designated as input units, others as output units. An artificial neural network is "trained" empirically on a data set by adjusting its weights, so a given input can be mapped to a target output. In this way, the network learns, or discovers, the knowledge embedded in the data. Its power stems from the network's own internal ability to adapt and self-organize.

The neural network approach has a good theoretical foundation and effectively addresses the weakness of the symbolic AI approach. Although many real-world applications attest to its viability, it also has some disadvantages, such as the fact that neural network knowledge is cryptically coded as a large number of weights, and their semantics (in terms of the problem to be solved) are not explicit. Two issues are important for getting past these disadvantages: how to represent and how to extract neural network knowledge.

Knowledge Representation

The human mind has difficulty comprehending the knowledge of a neural network, as determined by its connection pattern and weights. Lack of comprehension causes concern about the credibility of the result when neural networks are applied to risky domains, such as patient care and financial investment. In a broad sense, we say a neural network has discovered knowledge if it deals adequately with the problem at hand. In a strict sense, however, we also say a neural network has discovered knowledge only if that knowledge can be put in a human-understandable form. Numerous research papers describe interesting real-world applications of neural networks, though only a few have addressed how to explicitly represent neural network knowledge. The following examples involve a distinct formalism for knowledge representation.

Figure 1. Revision of a promoter theory by a neural network.



Theory formation in bioinformatics. A knowledge-based neural network refines the theory of molecular biology based on the conformation hypothesis [6] for predicting the presence of a "promoter" in a DNA sequence [4, 7]. The promoter is a region in DNA that indicates to a cellular mechanism the presence of a gene up ahead. The theory is first mapped into the neural network. The network is then trained using a

learning procedure called "backpropagation" [4]. Finally, the knowledge of the trained network is decoded, representing a revised version of the prior theory. In the revised theory, the conformation hypothesis about promoters is de-emphasized (see Figure 1).

In both initial theory and revised theory, knowledge is represented as a set of if-then rules, which are easily converted into a computer program in the logic programming language Prolog.

Grammatical inference in languages. A second-order recurrent neural network has been applied to learn regular grammars in natural languages [5]. Unlike a "feedforward" neural network, a recurrent neural network has feedback connections. A second-order connection combines inputs from two nodes, often through multiplication. The network was trained on 1,000 strings using a special gradient-descent algorithm adapted to the recurrent structure; the grammar learned was then extracted from the network using a special procedure called "dynamic state partition." The knowledge—the grammar—discovered was represented as a transition graph, classifying all 65,535 test strings with no error. The neural network had evidently discovered the correct grammar.

Prediction and synthesis in chemical reactions. The Kohonen neural network, a kind of artificial neural network, was used to classify chemical reactions with a common reaction center described by physicochemical properties [2]. The Kohonen network uses a special self-organization mechanism to model its environmental inputs as a map analogous to, for instance, the visual or auditory map in the human brain. Not only were the chemical reactions categorized in a way consistent with predetermined reaction types, their relationships were also explored. The knowledge discovered was represented by a 2D Kohonen map (landscape) revealing the consequence of various influences in a chemical reaction and suggesting different levels of similarities of the reactions under consideration. Note too that the 2D landscape provided by a Kohonen network transcends what can be found through traditional clustering methods.

Knowledge Extraction

Since computer programs make discoveries by seeking "regularities" from observations, neural networks endowed naturally with such ability show promise for this task. Various neural network architectures intended for this purpose all share the same idea—that neural networks discover regularities [4]. However, heuristic guidance is still needed, since neural networks today are not intelligent enough to learn everything from scratch.

An important AI research topic is knowledge extraction from a trained neural network. In general, it proceeds as follows: An artificial neural network is trained for modeling the knowledge embedded in the data by adaptation, or self-organization. The network knowledge is then extracted or interpreted. Symbolic knowledge can be extracted from a trained neural network through two approaches:

Decompositional. Each internal element in the neural network is examined. The knowledge extracted at this level is then combined to form the knowledge base of the entire neural network.

Pedagogical. Only the network input/output behavior is observed. It can be viewed as a learning task in which the target concept is the function computed by the neural network.

The decompositional approach is also called an "open-box" approach; its pedagogical counterpart is a "closed-box" approach. In general, the first approach is more analytical, the second one more empirical [1].

The decompositional approach is represented by two important developments: the Knowledgetron (KT) method [4] and the so-called M-of-N method [7]. The pedagogical approach is illustrated by the RULENEG program [1].

KT. This method heuristically searches through the rule space expanded in terms of combinations of attributes, distinguished into pos-atts, which, for concept *C*, refers to an attribute, and neg-atts, which also refers to an attribute of concept *C* but in terms of a negative weight, depending on the concept for which the rules were formed. We define a pos-att for the concept *C* to be an attribute designating a node that connects directly to the node corresponding to *C* with a positive connection weight. We define a neg-att for *C* in the same way, except that the connection weight is negative. Rules are sought for each processing node in the network before they can be combined into input-output rules without involving hidden concepts.

To extract conforming rules, KT first explores combinations of pos-atts, then uses negated neg-atts in conjunction to strengthen the positive combinations. Specifically, for each hidden unit and each output unit, KT searches for combinations of pos-atts whose summed weights exceed the threshold on the unit; for each such combination, KT then tries to form a valid rule, often by coupling with some negated neg-atts.

Similarly, to extract disconforming rules, KT first explores combinations of neg-atts, then uses negated pos-atts in conjunction. The distinction between these two kinds of attributes—pos-atts and neg-atts—

reduces the size of the search space considerably. An important idea in KT is to search for valid rules. The validity condition is defined as: Whenever the rule's premise holds, its conclusion also holds, irrespective of any combination of the values of attributes not referenced by the rule. A weakness of this method stems from the combinatorial problem of rule formation.

M-of-N. This method explicitly searches for rules of the following form: "If *M* of the following *N* antecedents are true, then …" For each hidden unit and output unit, M-of-N forms groups of similarly weighted links by clustering, sets the link weights of all group members to the average of the group, removes the groups that do not significantly affect whether the unit is active or inactive, keeps all link weights constant, and optimizes the biases of all hidden and output units, using the backpropagation procedure.

For each hidden and output unit, a single rule is generated consisting of a threshold given by the bias and weighted antecedents specified by the remaining links. Where possible, rules are simplified to eliminate superfluous weights and thresholds. In this approach, the extracted rules can be viewed as a simplified version of the neural network in that they are often associated with weights and thresholds. Such rules look more like discriminant formulae than what we call "rules" in traditional knowledge-based systems. Moreover, these rules cannot be run by a pure symbolic pattern matcher.

RULENEG. This program generates rules based on individual patterns (instances) in the training set of given data. To generate a rule on a given instance, RULENEG tests the importance of each attribute value in the instance. An attribute value is considered by the program if its negation leads to a classification change. A rule is formed by collecting such attribute values. RULENEG is also sensitive to data noise. For example, if the neural network overfits the data, the rules extracted are likely to be overly specific for the problem domain being considered.

Future Directions

Future research on neural networks for knowledge discovery is likely to take two directions. One is to seek a better general theory for knowledge extraction. The other is to develop a special neural network whose knowledge can be decoded faithfully. Progress in the first direction seems to have reached a plateau; the second, however, has lots of potential. A significant development is the Certainty Factor Network (CFNet) described in [3]. This special network refers to the feedforward multilayer neural network in which the network activation function is based on the certainty factor model, a model for uncertainty management in traditional expert systems. It overcomes the traditional disadvantage—having a black-box nature—in the neural network approach, allowing its knowledge to be decoded precisely. The CFNet can discover the domain concepts from a small fraction of domain instances with accuracy significantly better than C4.5, which today is the best representative rule-learning system available.

A knowledge discovery system has to be able to deal with domain complexity and data noise. In meeting these needs, the neural network approach seems to hold the promise of providing the ultimate solution for knowledge discovery. However, delivering this promise depends on how neural network knowledge is understood in a human sense. Two issues are therefore critical: knowledge representation and knowledge extraction. Future research will seek a better general theory for knowledge extraction and a way to develop a special neural network whose knowledge can be decoded faithfully. As a sign of the potential of the second approach, CFNet can accurately discover the domain concepts and significantly outperform the popular rule-learning tool C4.5. C

References

1. Andrews, R., Diederich, J., and Tickle, A. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowl.-Based Syst. 8,* 6 (1995), 373–389.
2. Chen, L. and Gasteiger, J. Knowledge discovery in reaction databases: Landscaping organic reactions by a self-organizing neural network. *J. Am. Chem. Soc. 119,* 17 (1997), 4033–4042.
3. Fu, L.M. A neural network model for learning domain rules based on its activation function characteristics. *IEEE Trans. Neur. Nets. 9,* 5 (1998), 787–795; see also www.cise.u.edu/fu.
4. Fu, L.M. *Neural Networks in Computer Intelligence.* McGraw Hill, New York, 1994.
5. Giles, C., Miller, C., Chen, D., Sun, G., Chen, H., and Lee, Y. Extracting and learning an unknown grammar with recurrent neural networks. In *Advances in Neural Information Processing Systems 4.* Morgan Kaufmann, San Mateo, Calif., 1992.
6. Koudelka, G., Harrison, S., and Ptashne, M. Effect of non-contacted bases on the affinity of 434 operator for 434 repressor and Cro. *Nature* 326 (1987), 886–888.
7. Towell, G. and Shavlik, J. Knowledge-based artificial neural networks. *Artif. Intel. 70,* 1-2 (1994), 119–165.

LiMin Fu (fu@cise.ufl.edu) is an associate professor of computer and information science and engineering in the Computer and Information Sciences and Engineering Department at the University of Florida in Gainesville, Fla.