Adaptive Learning Expert System[1]

Sakchai Wiriyacoonkasem and Albert C. Esterline
Department of Computer Science and ACE Center
North Carolina A&T State University
Greensboro, NC 27411
Email: {sw981996, esterlin}@ncat.edu

Abstract

The purpose of this research is to improve the performance of an expert system through the use of a neural network, thus allowing the expert system to learn from experience. Even though the knowledge representation schemes used by expert systems allow them to succeed and proliferate, these schemes cause them to be brittle. Human experts usually use more knowledge to reason than expert systems do and often use experience in quantitative reasoning whereas expert systems cannot. Our study shows that a neural network can learn from an expert system's experience and guide the expert system when the expert system does not have enough knowledge to reason.

Introduction

The purpose of this research is to demonstrate that an expert system can be coupled with a neural net so that it may learn from past experience to select rules in a more efficient order and to ask appropriate questions. Expert systems have inherited a major problem from imperfect knowledge representations and their own principle of trying to match the performance of human experts on "narrowly" defined tasks [3, pp.257]. Since most domains cannot be captured with a reasonable number of rules, a knowledge engineer must judiciously select a small subset of the possible rules characterizing a domain. Facts can be thought of as the boundary conditions, which either initiate reasoning (in the case of forward chaining) or terminate reasoning (in the case of backward chaining); a small subset of the relevant facts also must be selected. As a consequence of this selection, many of the rules and facts in an expert system may not relate to each other through any direct chain of reasoning – the relations among these items remain hidden in the unarticulated substrate of the domain. This incomplete nature of an expert system's knowledge affects its reasoning since it must often blindly chose the next rule to apply. For the same reason, an expert system often asks the same sequence of questions regardless of the problem instance, asks many more questions than a human expert would, and asks irrelevant questions. Incomplete knowledge also makes an expert system brittle: when faced with a difficult problem, it can do nothing at all whereas a human expert could explore more subtle aspects of the domain. Thus, if an expert system could learn the hidden relations among domain elements, it would not only ask fewer questions but it could also make good guesses when faced with difficult problems.

Making a good guess means that we actually have some reason to support it. A human makes a guess by using qualitative reasoning applied to the instance of the problem or by trying to apply other knowledge believed to be helpful. However, a human also uses quantitative reasoning derived from experience to support decisions. In this case, using a neural network might be

[1] This research was supported by NASA Goddard Space Flight Center, grant NAG 5-4102, "Formal Foundations of Agents."

0-7803-6312-4/00/$10.00 ©2000 IEEE.

helpful since neural networks can identify the hidden relations among elements by learning from past experience [1,2]. Since the expert system interacts with a human, who in some way (if only implicitly) has access to the hidden relations, a neural network might learn the hidden relations from the interaction between the expert system and the human user.

Because an expert system is a qualitative model whereas a neural network is a quantitative model, there arises the question of how we can translate the information into quantitative form to be the input of neural network. There is also the question of how we can interpret the output.

Integration of an Expert System and a Neural Net

We used a back-propagation neural network, a popular supervised learning model. The intended system now is a dual system (see Figure 1). Firstly, whenever the expert system does not have a reason to select any given question, the expert system will let the neural network make the decision (see Figure 2). Thus, the expert system has to tell the current situation and the set of questions to the neural network. The current situation consists of all the values of the elements in the domain. The neural network has to test each question one by one and predict the consequence of each question. The neural network has to have some mechanism to evaluate which question is the best to select.

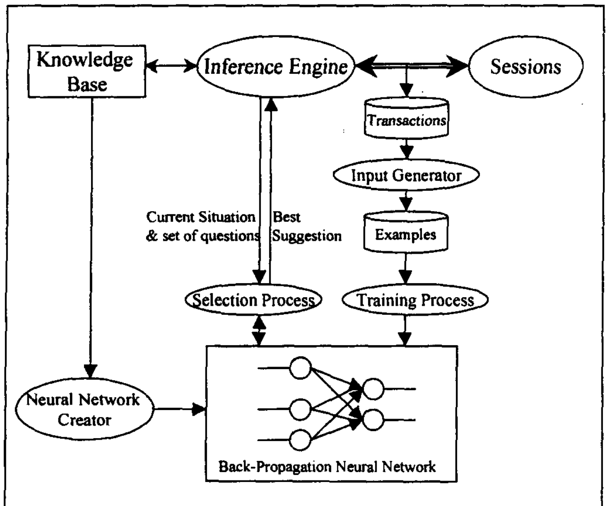


Figure 1. Adaptive Learning Expert System Model

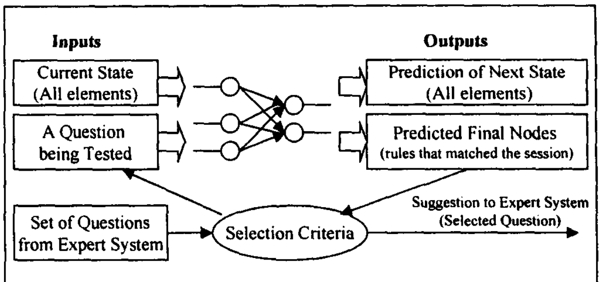


Figure 2. Selection Process

Secondly, and at the same time, the transactions between the expert system and the subjects are collected. Whenever a subject finishes a session, the input generator examines the transactions and generates a set of examples for the neural network (see Figure 3). The neural network starts training when the number of examples is large enough (about twice the number of elements), and it starts training again when the same number of additional examples are added. We predicted that the average square error would be large because, at any one time, there are many possible outcomes. As a result, some examples can have the same inputs but different outputs. Therefore, the neural network would never be able to reduce the average square error to near zero. Moreover, we use the neural network only for ordering the questions. Thus, the average error from the neural net is not critical. For this reason, we have the neural network stop training when the maximum number of iterations is reached, not when some error threshold is reached.

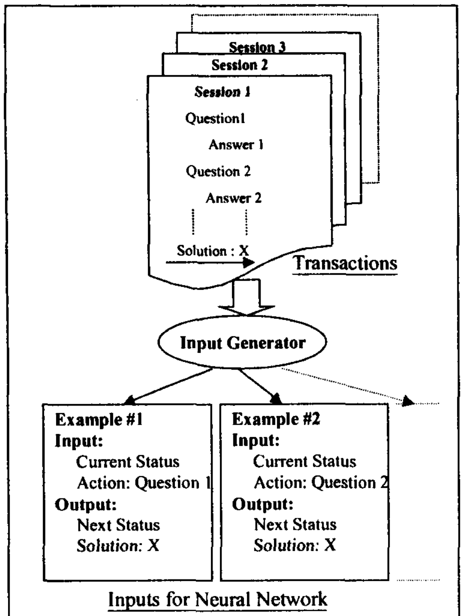


Figure 3. Input Generator

We add a set of special, *final* nodes at the output layer to help evaluate each question tested (see Figure 4). Each *final* node represents the final answer or solution of a session. (This is the main reason the input generator can generate the examples only after each subject finishes a session.) The neural network should select the question that causes the greatest ratios of the maximum value of a final node for the predicted case (rule) to the values of the other final nodes for that case. Thus, the neural network simply calculates the ratio by dividing the maximum node value by the average node value. The maximum ratio indicates the question most likely to cause the system to reach the conclusion fastest.

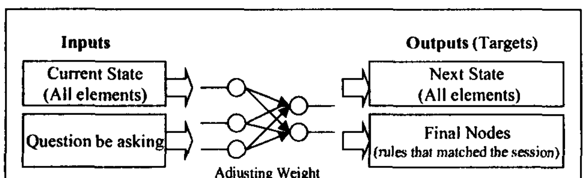


Figure 4. Training Process

The neural network has to be flexible to handle sudden changes of pattern because new examples added to the system might be totally different from those that the neural network used for learning. This is the other reason that we assign a small maximum number of iterations for each training session.

We noticed that, when we assigned 1.0 as the TRUE value and 0.0 as the FALSE value, the range in node weights was so extreme that the neural network usually could not learn a new pattern when new examples were added. To alleviate this problem, we assigned 0.9 as the TRUE value and 0.1 as the FALSE value to avoid over-training and to balance the weights. The UNKNOWN value remained the same at 0.5. As a result, the improvement this system exhibited over the bare expert system increased by nearly 50%.

Experimental Results

To test the utility of this insight, we set up an experimental situation by developing a simple expert system that contains four rules for four kinds of problems with twelve elements. The sessions were generated with different probabilities. The original, unaided expert system was found to select rules 30% worse than the optimal selection. When coupled with the neural network, the expert system defers to the neural network when it cannot decide which rule is most relevant. Initially, the neural network's selections are random, and we found that its selections were 45% worse than those of the original expert system (see Figure 5). Improvement was noted after three sessions were processed. Substantial improvement was noted after 12 sessions, and no significant improvement was noted after 60 sessions. In the steady state, the system performed 19.4% better on average than the original. The upper bound is 30.6% and the lower bound is 9.5%. Moreover, the system is more likely to select relevant questions.

We set up the second experiment by doubling the knowledge base size to eight rules 24 elements. The relations between the elements are more complex. The original, unaided expert system performed about 28% worse than the optimal selection. When coupled with the neural network, the system started at about 40% worse than the original. Substantial improvement was noted after about 50 sessions. In the steady state, the system performed about 6.7% better on average than the original. The upper bound is 19.8%. The lower bound is –7.4%. The system is still likely to select relevant questions.

Figure 5. Performance after embedding Neural Network with small knowledge base

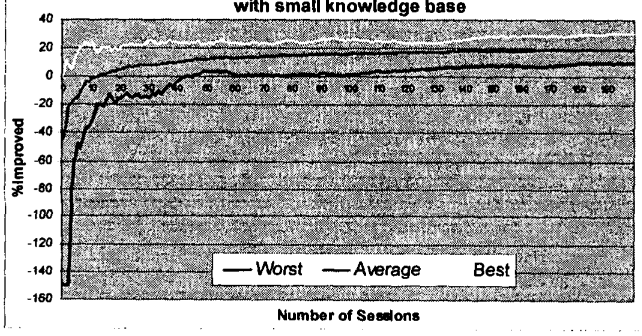
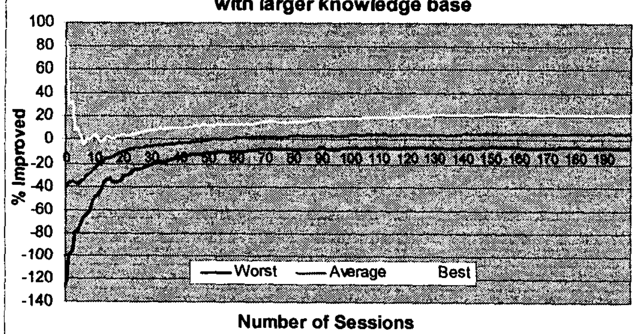


Figure 6. Performance after embeding Neural Network with larger knowledge base



Conclusion

This evidence indicates that the neural network can retain the probabilities of events and can find relations among elements. As a result, the neural network improves the expert system. Because the neural network does not modify the knowledge base, it is easy to embed the neural network module into existing expert systems. If we analyze how the weight of each element contributes to those of the others, we might find some logical relations among them or hidden rules. At least this could suggest guidelines or good guesses as to the areas of the domain where the system needs to be enhanced.

References

[1] Anderson, James A. (1995). An Introduction to Neural Networks, Cambridge,Massachusetts, London, England: The MIT Press, 1995.

[2] Neapolitan, R. E. (1990). Probabilistic Reasoning in Expert Systems: Theory and Algorithms. New York: Wiley, 1990.

[3] Russell, Stuart., Norvig, Peter. (1995). Artificial Intelligence A Modern Approach. Upper Saddle River, New Jersey: Prentice Hall, 1995.