Complex Systems **3** (1989) 295–311

Using Artificial Neural Nets for Statistical Discovery: Observations after Using Backpropogation, Expert Systems, and Multiple-Linear Regression on Clinical Trial Data

Erach A. Irani
John P. Matts
John M. Long
James R. Slagle
POSCH group
University of Minnesota, Minneapolis, MN 55455, USA

Abstract. Powerful new training algorithms developed for artificial neural networks hold the promise of identifying regularities in the training data and generalizing over the test data. The backpropagation algorithm is one such training algorithm that, with the use of hidden units, can learn functions such as exclusive-or. These functions can be learned by statistical techniques such as multiple-linear regression only by introducing additional parameters. We report experimental comparisons of the performance of backpropagation, multiple-linear regression, and an expert system. We conclude that, for the data studied here, backpropagation is unsuitable for discovering statistical relationships. It may be possible to customize neural-net algorithms for niche applications in discovery systems.

1. Introduction

Recent years have witnessed an explosive increase in neural-net research [10]. This increase has been fueled partly by the development of new training algorithms such as generalized backpropagation and the Boltzmann machine. These new algorithms learn functions, such as exclusive-or and parity-encoder, that require that a group of inputs be considered collectively rather than one at a time to determine the output [10].

At the University of Minnesota, we have been collaborating with the Program on Surgical Control of the Hyperlipidemias (POSCH), a large multicenter clinical trial. We have already developed two expert systems for doing expert evaluation of clinical trial data and are developing automated tools for doing the statistical analysis of the collected data [5–7,11]. We

are developing a discovery system for investigating the relationships among variables in the clinical trial database.

Neural nets were considered as one technique for use in a system that automatically discovers significant relationships in the database. Generalized backpropagation is one of the better known neural-net algorithms that can train nets with hidden units. It has the potential of using hidden units to discover regularities in the data. These regularities can be thought of as probabilistic relationships of higher than first order [10]. If neural nets can use hidden units to discover relationships of higher order with a minimum of human interaction, then they provide an important tool for discovering the structure of a set of data. Our intention was to compare backpropagation with other techniques to get a feel for how useful neural nets would be for statistical discovery over our data.

Before getting deeply involved with neural nets, we decided to get a feel for how backpropagation works with real-world data, in particular our clinical trial data. For this purpose, we trained a network using the backpropagation algorithm on a subset of the cases used in developing ESCA, an expert system developed here for assessment of change in heart disease [5], and tested it out on the remaining cases. Other datasets on which we tested backpropagation were peripheral disease data and computer-generated random data.

Section 2 describes the hypotheses we were testing in the experiments. Section 3 describes the datasets used and why they were chosen. The results of using backpropagation and other methods on the datasets are presented in section 4. Comments on the predictibility of the network's performance are given in section 5. Section 6 comments on the limitations of using backpropagation for discovery systems. Section 7 mentions the advantages of the Boltzmann machine for discovery systems. Section 8 gives conclusions.

2. Hypotheses tested in the experiments

We tested the hypothesis that a neural net trained using backpropagation on a large subset of the data could generalize to the remainder of the subset. This is the train-and-generalize paradigm. This generalization would be achieved if the net learned to configure itself so as to exploit the regularities in the data being presented. One test of the net having generalized "well" is that it could predict the output variables of the other set of data when the input variables were presented to it. If the net did not pass this test, it did not generalize well, and we do not know whether the net has learned the regularities in the data.

There have been attempts to obtain useful systems by using backpropagation for nets with and without hidden layers in the train-and-generalize paradigm outlined above. These systems can be used as *expert systems* [3,8]. These "expert systems" have the limitation that they cannot explain their decision process to the user as an expert system can. One of the motivations for the development of expert systems which use rules [13] or inference functions [12] derived from the knowledge of domain experts, is that the

domain experts feel more comfortable when the systems use domain knowledge in a manner they understand.

The uneasiness of experts with systems derived from a purely statistical basis was a major constraint in using these systems. With neural nets, the method of predicting outputs is even less understandable or explainable to a domain expert. Another limitation of neural nets is that it is hard to decipher the information encoded in the pattern of weights. If neural nets are used as a learning technique for discovering new knowledge, one has to be able to report the knowledge concisely so that people can understand it. Given these limitations, it is necessary to determine how well neural nets can learn the regularities of a domain. If neural nets do a substantially better job than multiple-linear regression or symbolic artificial intelligence techniques at capturing the regularities of a domain, then additional research can be done to determine how to use neural nets to automatically generate expert systems or acquire new knowledge. For backpropagation, the train-and-generalize paradigm is one way to compare its performance at capturing regularities with that of multiple-linear regression or expert systems.

3. Data used in the experiments

Experiments have been carried out on three sets of data. We can be more confident of the results because three different sets of data are used. The backpropagation model was used for all the experiments. Tests were done with different numbers of layers and a different number of units in each layer. The algorithm described in Chapter 8 of [10] was used. Values of 0.3 for "eta" and 0.9 for "alpha" were used. The units in all layers were completely connected with units in their immediate preceding and succeeding layers.

The rationale behind using completely connected layers is as follows:

> If we are to establish the topology of the layers, then we are supplying the net with information about how units influence each other. Considering that the networks use a distributed encoding for representation of information which is hard for humans to understand, it would be difficult for people to supply this information in a real task.
>
> One has to preset the number of layers and the number of units in each layer. Even supplying this information results in a large number of experiments being necessary to determine the best performance. In addition, supplying the topology increases the number of experiments that should be carried out. While sparsely connected topologies can reduce computer training time, the researcher has to spend more time supplying the topologies and comparing the results unless he is supported by a neural-net experimentation environment. Computer training time is an important consideration when carrying out neural-net experiments with real world data. For our experiments, it took more

than 800 minutes of SUN III[1] workstation time to train a particular network configuration over 700 iterations.

3.1 Experimenting with computer-generated data

Input fields are computer generated and consist of random numbers in the range 0.1 to 0.9. The output is computed by first mapping the input fields to intermediate scores when they lie in a pre-specified range and then adding these intermediate scores together. This weighted addition seems common in our domain; it has been used for expert systems we have developed and for computing the indexes of peripheral disease. It is essential for our domain that the network be capable of learning these mapped scoring functions.

46 input fields and one output field are generated. In Experiment (A), the input fields are obtained by generating random numbers between 0.1 and 0.9. A score is computed for each field given by

$$\text{score} = 0.0, \text{ if } 0.1 \leq \text{field} \leq 0.4$$

$$\text{score} = 0.01, \text{ if } 0.4 \leq \text{field} \leq 0.7$$

$$\text{score} = 0.03, \text{ if } 0.7 \leq \text{field} \leq 0.9$$

The scores for the various fields are added to give a total score. The output is computed as follows:

$$\text{output} = 0.1, \text{ if } 0.0 \leq \text{total score} \leq 0.3$$

$$\text{output} = 0.2, \text{ if } 0.3 \leq \text{total score} \leq 0.5$$

$$\text{output} = 0.3, \text{ if } 0.5 \leq \text{total score} \leq 0.9$$

$$\text{output} = 0.8, \text{ if } 0.9 \leq \text{total score} \leq 0.94$$

3.2 Experimenting with expert-system data

The data used for testing ESCA, an expert system developed for POSCH to evaluate data collected on angiograms taken for the same patient some years apart was used for this experiment [5]. Angiograms are photos taken of the heart after the coronary arteries are injected with radiographic dye to increase the contrast. This experiment compares the performance of backpropagation with an expert system doing the same task. Multiple-linear regression was

[1]SUN and SUN III are trademarks of SUN Microsystems, Inc.

	NA1-Tr	NA1-Ts	NA2-Tr	NA2-Ts
Exact	82.67	64.0	90.67	52.0
Within one	100.0	100.0	100.0	100.0

Table 1: Results of computer-generated data. NA1 has 0 hidden layers. NA2 has two hidden layers with three hidden units in each layer. Tr = training; ts = test.

used over the training data to determine the relationship between the output variable and the input variables. The coefficients obtained from the regression were inserted into an equation. This equation was then used to predict the output variable from the input variables from the test data.

3.3 Experimenting with peripheral disease data

Data on peripheral disease collected on a sample of POSCH patients was used in this experiment. Three different indexes of peripheral disease are mathematically computed from the percentages of stenoses (constrictions due to lipids deposited on the walls of the arteries) in the peripheral arteries. These indexes based on mathematical scoring can be expected in most real-world applications. Other data related to peripheral disease, such as the patient's history, cholesterol levels, and noise variables such as height and age, are included. Such noise variables have to be considered when doing a statistical analysis. The performance of backpropagation to predict one of these indexes in the presence of irrelevant variables is tested.

4. The experiments and their results

700 training iterations were used in each experiment. It was observed that there was little difference in the net's predictions at the 600th and 700th iteration. The net's predictions before the 500th iteration on the training set showed little change with those of the 600th and 700th iteration. To determine whether training the network further would improve results, additional experiments were carried out. The results of these experiments and whether training the network further would improve performance are discussed in section 5.

4.1 Results with computer-generated data

Seventy-five training cases and 25 test cases were used. The network with no hidden layers and the net with 2 hidden layers with 3 units in each layer had the best overall predictive performance. The performance of these two network configurations is given in table 1.

The network learned to predict the output exactly or within one category of the exact answer for both the training and the test sets. These results

	SP	ES	MLR-Tr	MLR-Ts
Exact	38	46	57	44
Within one	92	93	98	93

Table 2: Results of techniques other than backpropagation with unsubtracted expert system data. SP is comparison of subpanels with subpanels. ES is comparison of expert system with subpanel. MLR compares multiple-linear regression with subpanel. MLR-tr is results of multiple-linear regression on the training set; MLR-ts is results of multiple-linear regression on the test set.

indicate that the net should be able to predict correctly when mapped and summed inputs are present. The generalization performance on this "toy" data is poorer than the generalization performance reported for other tasks in the literature [2]. In reference [2], backpropagation was tested with different tasks performed on binary patterns. Our patterns require a continuous encoding, one possible cause of the poorer performance.

4.2 Results with expert system data

The data used was from comparisons of 200 pairs of angiograms made by subpanels of human experts. Experts record their observations of the visibility of the angiograms, the morphology, stenoses, and ratings for change for arteries viewed. The expert system used the information on stenoses of 23 pairs of arteries. The experts must give a global assessment of change of atherosclerotic disease on an 8-point scale from -3 to $+3$, with -3 being much worse and $+3$ being much better, -0 being imperceptibly worse and $+0$ being imperceptibly better.

The expert system was designed to come up with this global assessment of change. It agreed exactly with the panel of experts on the test set of 200 cases for 46% of the cases and within one for 93% of the cases. During quality control studies, it has been discovered that subpanels of experts agree exactly with each other 38% of the time and within one 92% of the time. The subpanels make their assessments based on film, rather than the information encoded from the films, and this introduces one more factor of uncertainty for the subpanels. The results for subpanel versus subpanel, expert system versus subpanel, and multiple-linear regression versus subpanel are given in table 2. They provide an alternative with which to compare the neural net results.

The neural net was trained on the output assessment given the percentage stenoses observed for the arteries in the angiograms. These angiograms are taken at different times for the same patient. In the first experiment, for each artery the net was given the absolute values of both the percentage stenoses. Various configurations of the network were used. The results are summarized in figure 1.

The results for the best configuration are given in table 3. In the second

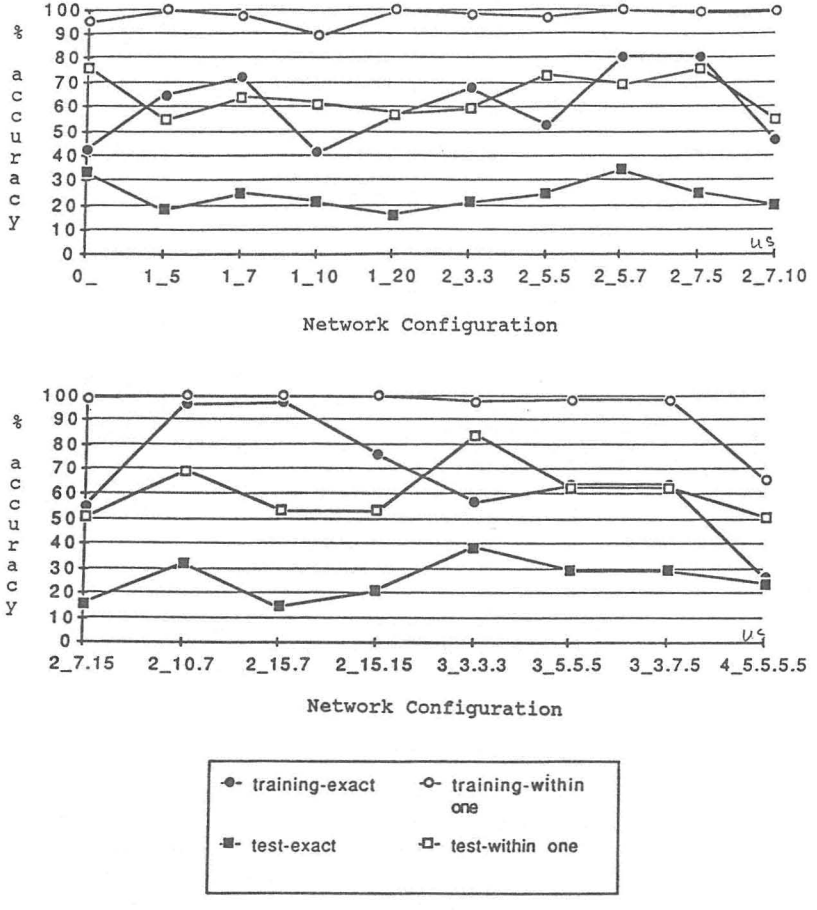

Figure 1: Training backpropagation on unsubtracted expert system data for different configurations. Percentage accuracy of net trained on unsubtracted data after 700 iterations for various configurations of the network. 0_ refers to the net with no hidden layers; 2_10.7 refers to the net with two hidden layers having ten units in the first hidden layer and seven in the second hidden layer.

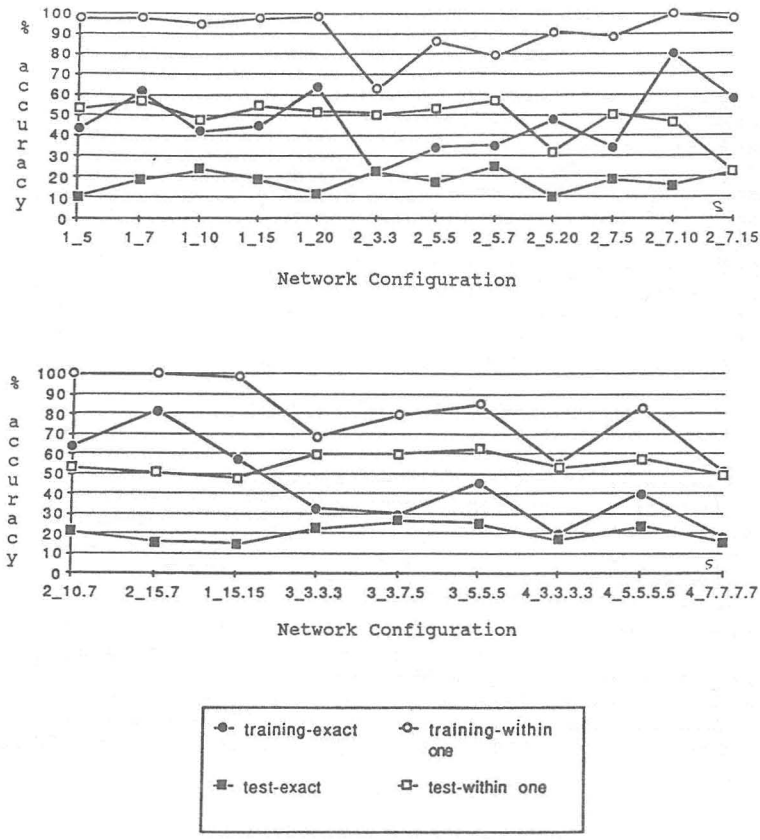

Figure 2: Training backpropagation on subtracted expert system data for different configurations. Percentage accuracy of net trained on subtracted data after 700 iterations for various configurations of the network. 1_5 refers to the net with one hidden layer and five units in the first hidden layer; 2_10.7 refers to the net with two hidden layers having ten units in the first hidden layer and seven in the second hidden layer.

experiment, the net was given the percentage stenoses observed in the first angiogram, and the difference in stenoses between the first angiogram and the second. Figure 2 gives the results. The best results are given in table 4.

In both experiments, the neural net was trained on 125 cases from the data used to test the expert system and tested on the remaining 75 cases. Different numbers of hidden layers with different numbers of units in each hidden layer

	NB1-tr	NB1-ts	NB2-tr	NB2-ts
Exact	42	33	57	39
Within one	95	76	98	84

Table 3: Results of backpropagation with unsubtracted expert system data. NB1 is neural net with no hidden layers compared to subpanel. NB2 is neural net with three hidden layers with three units in each hidden layer.

	NC1tr	NC1ts	NC2tr	NC2ts
Exact	80	16	32.8	22.67
Within one	100	46.7	62.8	60.0
	NC3tr	NC3ts	NC4tr	NC4ts
Exact	45.6	25.3	43.2	10.7
Within one	84.8	62.6	97.6	53.34

Table 4: Results of backpropagation with subtracted expert system data. NC1 is neural net with two hidden layers. The first hidden layer has seven units; the second has ten units. NC2 has three hidden layers with three units in each hidden layer. NC3 has three hidden layers with five units in each hidden layer. NC4 has one hidden layer with five units. NC1tr refers to training performance of NC1; NC1ts refers to test performance of NC1.

were used. The performance of the net with no hidden layers compared favorably with the best performance obtained. The best performance at predicting the data during both test and training in terms of the number of assessments accurately predicted were observed for the net trained with the absolute percentages (figure 1). That particular net had three hidden layers with three units in each layer. The results for the subpanels, the expert system, and backpropagation are summarized in table 3. The neural net performance on the training set is better than the experts. On the test set it is poorer than the experts, the expert system, and multiple-linear regression.

The net with subtracted data did not perform as well as the net with unsubtracted data. For subtracted data, a network configuration that would do well on the training set would do poorly on the test set (figure 2). For unsubtracted data, there are configurations that do well on both the training set and the test set (figure 1).

Several factors can explain the difference.

Percentage changes were supplied to the net with subtracted data. Percent changes vary more (from -100 to $+100$) than absolute percentages (0 to 100). In mapping percent changes to a floating point number between 0 and 1, the dynamic range is reduced more than when mapping absolute percentages. When backpropagation was tested with exclusive-or, with 0 encoded as .005 and 1 encoded as .095, the network was unable to learn the function.

It may well be impossible to predict the performance of backpropagation in individual configurations. Another configuration of the network with percentage changes might do as well as the network trained on the absolute percentages.

Experiments have shown that single-layer networks using backpropagation can explain human learning mechanisms [4]. It is possible that human experts view data in terms of the absolute percentages rather than a first measurement and a change. In that case, the network that used absolute percentages was a more faithful representation of the decision-making process.

4.3 Results with peripheral disease data

The peripheral disease data has several features that potentially make it more difficult for a neural network to predict outputs both over the training set and the test set.

1. The neural net is asked to predict an index of peripheral disease mathematically computed from some of the input fields. Some of the other input fields are potential causes for the mathematically relevant fields, causing interference for the network.

2. Much data that is possibly irrelevant, such as patient height, is mixed with data that measures causes of peripheral disease (such as cholesterol, ratios of HDL/LDL cholesterol, smoking history) and data used to compute the indices of disease (the percent stenoses in each artery). By comparison, in the angiogram experiment only the data that were determined most useful by the experts to arrive at the global assessment was input to the net.

3. Some of the information has non-numeric connotation and was mapped to real numbers (e.g., sex was mapped to either 0.4 or 0.8, type of smoker cigar/cigarette/pipe or a combination was mapped to numbers from 0.1 to 0.5).

4. The dynamic range of the data was much larger. While stenoses range from 0 to 100 with peripheral disease data there are inputs such as weight that could be above 200 pounds, and cholesterol measurements in milligrams that could range up to 500 milligrams.

Table 5 gives the results of the neural net on peripheral disease.

5. Predictability of the net's performance

The net's performance cannot be predicted. There is no guarantee that increasing the number of layers or increasing the number of units in a layer or both will improve either test or training performance. The performance

	NC-Tr	NC-Ts
Exact	83	64
Within one	100	100

Table 5: Results of peripheral disease data. NC is neural net with no hidden layers. Tr = training; Ts = test.

on the training set is not a good indicator of the performance on the test set for subtracted or unsubtracted data.

To determine the effect of additional training iterations on the same network configuration, additional experiments were conducted with the configurations having the best test performance for 1, 2, and 3 hidden layers (figure 3). These configurations were trained for 2,250 iterations of the entire training set. The absolute fractional changes for all weights for each connection for an iteration was calculated. These fractional weight changes were then added together to get the total absolute fractional weight change. The total absolute fractional weight changes occurring over the last 250 iterations is plotted for every 250th iteration, along with the information on the accuracy of the networks predictions for the training and test sets (figure 4). A further run of the net with three hidden layers was done. During this run, the floating-point precision for representing weights was reduced to six digits after the decimal point by introducing random noise after the sixth digit. The results are shown in figures 5 and 6. This reduction of precision is necessary when running the simulations on the CRAY-II.[2] Floating-point precision on the CRAY-II is so high that the network fails to predict simple functions like exclusive-or during training. Reducing precision seems to introduce enough random noise to keep the network from getting stuck in a local minimum.

The performance over the test set for the three network configurations worsened when the number of iterations was increased over 1000. The performance of the training set worsened over 1000 iterations for the network with three hidden layers. The performance of the training set for the other two network configurations improved with increasing iterations. We carried out tests with nets with more than three hidden layers; the results were no better than for the nets with fewer hidden layers. The deterioration in performance with extended training runs of 2,250 iterations seems to indicate that even three layers may be too many. It has been previously reported in [2] that multilayer architectures do no better than those with two hidden layers.

The possibility of the network getting stuck in a local minimum in rare cases is discussed in reference [10]. The unpredictability discussed here is of a different nature. It refers to the lack of features in the network that can be used to determine a good network configuration or when to stop training.

[2]Cray and Cray-II are trademarks of Cray Research, Inc.

The training time for each network configuration is large. We can try to reduce the training time by training the network in the early iterations on a randomly selected subset of the training data. An alternative is to select a different network learning algorithm that does not require experimenting with different configurations.

6. Limitations of using backpropagation for discovery systems

The limitations of using backpropagation for discovery systems are as follows:

1. A completely connected network has too many weights to analyze. If we are supposed to make a prior assumption about the layout of the network, we are making assumptions about the flow. If we are going to make assumptions about the flow, we might as well use standard statistical techniques. Besides, in backpropagation, the interaction between weights is complex and distributed. It is not realistic to expect a researcher to supply the topology that would best represent the flow of influence.

2. Compared to backpropagation, standard statistical techniques give results that are easier to interpret and take a significantly smaller amount of computer time. The backpropagation simulations take more than a day to run on a SUN III. Even on a CRAY, considering the scheduling of other jobs on the system, a batch of 20 backpropagation training runs for our experiment took days; ultimately, the runs were done over several SUNs.

3. Another difficulty with backpropagation is that it does not produce concise models of what happened in the world. In discovery systems, deriving the numerical relationships of the flow of influence between variables is tedious. Extracting symbolic information from the weights in backpropagation is an even more difficult task at this stage because the information encoded in the weights is distributed in nature.

4. Backpropagation's generalization performance to the test set is unpredictable for numeric data or for numeric data with non-numeric connotation. Our application and several real-world applications where statistical discovery is used consist of either numeric or symbolic data. In addition, statistical techniques seem better able to predict data and deliver easily interpretable results.

5. The necessity of running several configurations of the network makes backpropagation more difficult to use. A huge amount of computer time is needed to run the different configurations. Running several configurations makes the research frustrating, as comparisons have to be made to determine the best configurations. Neural net programs have not reached the state of maturity of statistical packages in terms of flexibility of supplying input, reporting results, and so on.

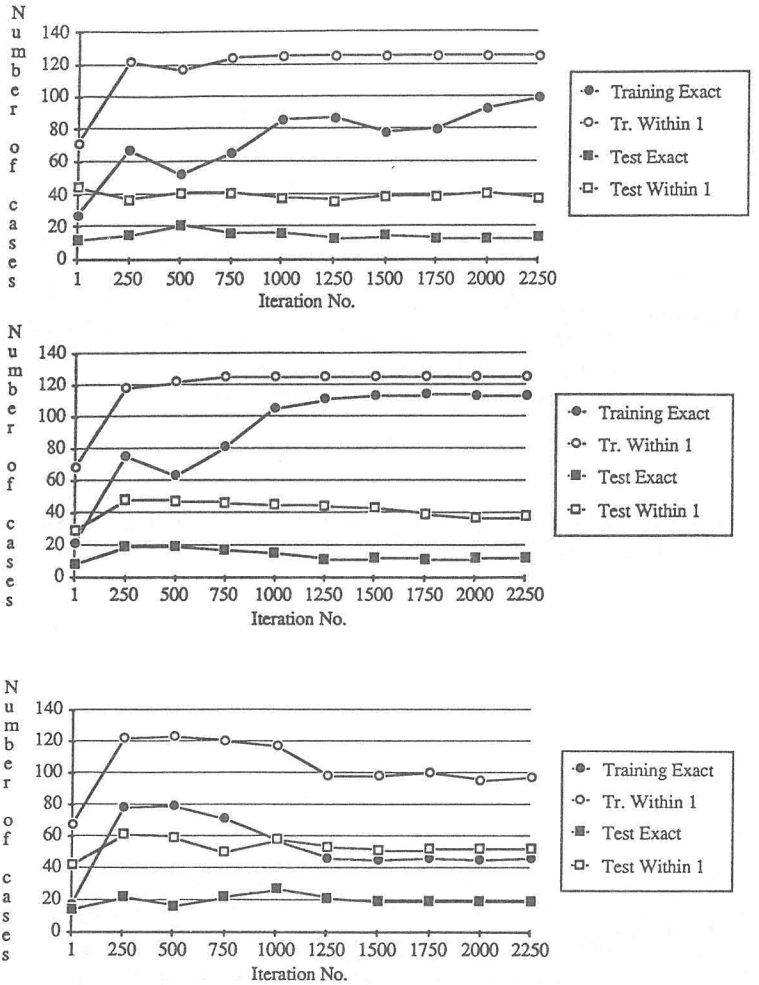

Figure 3: Accuracy observed in test and training the best nets with 1, 2, and 3 hidden layers for 2,250 iterations. (a) Training the net with one hidden layer containing seven hidden units. There are 125 cases in the training set and 75 cases in the test set. (b) Training the net with two hidden layers having seven units in the first hidden layer and five in the second hidden layer. There are 125 cases in the training set and 75 cases in the test set. (c) Accuracy observed in test and training the best nets with 1, 2, and 3 hidden layers for 2,250 iterations. Training the net with three hidden layers and three units in each hidden layer. There are 125 cases in the training set and 75 cases in the test set.

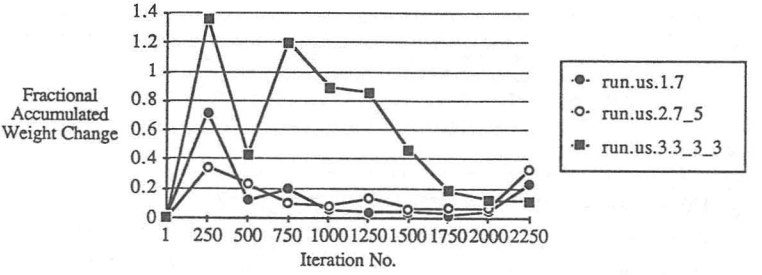

Figure 4: Fractional weight changes observed when training the best nets with 1, 2, and 3 hidden layers for 2,250 iterations.

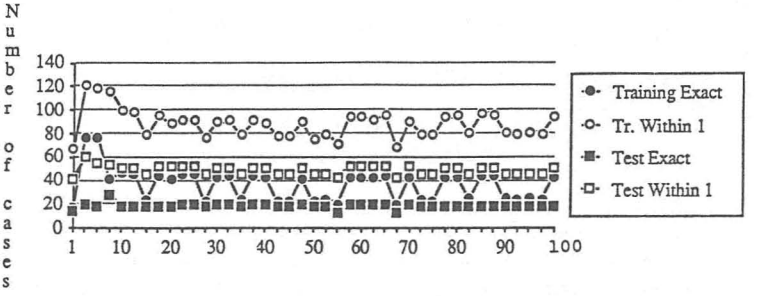

Figure 5: Training the best net with 3 hidden layers for 10,000 iterations. One hundred twenty-five cases are used in the training set and 75 cases are used in the test set. The floating-point accuracy of representing weights has been reduced to six digits after the decimal point.

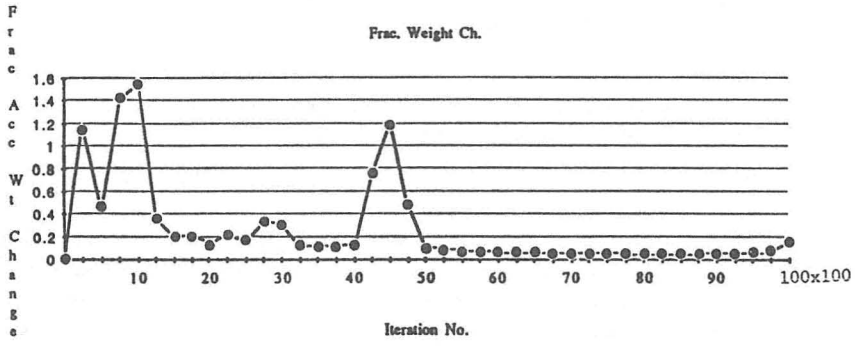

Figure 6: Fractional Weight changes observed when training the best net with 3 hidden layers for 10,000 iterations. Fractional weight changes are calculated by summing the absolute fractional weight changes over all connections in the network for the last 250 iterations.

6. The lack of a network-based criteria of when the network has converged to a stable state. How does one decide when to stop training? The answer to that question is not clear with generalized backpropagation.

7. The inability of the backpropagation algorithm to deal with missing data. Backpropagation expects all input fields to be present. This is unrealistic in the real world. Statistical packages allow the user to decide whether to ignore missing fields or supply the average. Other neural net algorithms can deal with missing fields and may be better suited for the real world.

7. Further neural net research for discovery systems

Further work involves testing the Boltzmann machine paradigm for the same sets of data. The Boltzmann machine paradigm seems to be more applicable to discovery systems than backpropagation for reasons given below. Another reason we are experimenting with the Boltzmann machine and the mean field theory approximation for the Boltzmann machine is that they may be useful for tasks such as robotic control where the input information is more perceptual in nature. In this case, coarse coding as used in reference [8] can help. Testing on the clinical data is a way of gaining familiarity with the Boltzmann machine algorithm.

The Boltzmann machine seems more applicable to real world tasks than backpropagation for several reasons.

1. An estimate of stochastic relations using the mean field theory speeds

up the settling algorithm considerably and yields qualitatively better results [9].

2. It is not necessary to determine how many units to have in each layer. This makes designing the network easier.

3. The connection strengths between units are adjusted based on estimates of correlational statistics. This may yield connection strengths more suitable for human interpretation.

4. The Boltzmann machine does not make a distinction between input and output units. It will predict values for all units whose values are not supplied. Values can be missing when data was not collected for a particular patient or was not possible to record. For instance, in recording stenoses for angiograms, the stenoses in arteries might not be recorded because they were poorly visualized or because an upstream stenosis in an artery completely blocked the flow of dye.

5. It should be possible to develop subroutines for a Boltzmann machine that would disable a connection once the weight falls below a certain threshold. In this fashion, a structure of flow of influence can be obtained.

8. Conclusions

The present state of maturity of neural net research requires extensive experimentation on the part of the researcher. In our experience, backpropagation has consistently done worse than statistical techniques or expert systems on both training and test sets. There may be applications where a simple training scheme that delivers sub-optimal results can be used in an automatic system, but the large training times required by backpropagation to find an optimum configuration makes it unsuitable. The Boltzmann machine with the mean field approximation can reduce overall training times and may yield more interpretable connection strengths; testing its performance is a possible future task. Neural net algorithms need to be customized further if they are to be used for discovery of statistical relationships in data where there may be used as specialized niche techniques, for instance over nonsymbolic data.

Acknowledgments

Paul Clarkson's assistance at the POSCH Coordinating Center for retrieving the data and converting it to other formats is invaluable. Maria Gini and Daniel Boley at the University of Minnesota secured supercomputer time from a grant by the Minnesota Supercomputer Institute. This material is based partly on work supported by NHLBI Grant 2R10HL15265 and by the Microelectronics and Information Sciences Center of the University of Minnesota. Erach enjoyed the support of his colleagues at the Honeywell Systems and Research Center, Minneapolis, while doing this research.

References

[1] C.K. Chow and C.N. Liu. "Approximating discrete probability distributions with dependence trees," *IEEE Trans. Inf. Theory*, **IT-14** (1968) 773–791.

[2] F. Fogelman Soulie, P. Gallinari, Y. Le Cun, and S. Thiria. "Evaluation of network architectures on test learning tasks," In *IEEE First International Conference on Neural Networks*, (1987) 653–660.

[3] Stephen I. Gallant. "Automated generation of connectionist expert systems for problems involving noise and redundancy," In *AAAI Workshop on Uncertainity*, Seattle (1987).

[4] Mark A. Gluck and Gordon H. Bower. "Evaluating an adaptive network model of human learning," *Journal of Memory and Language*, **27** (1988) 166–195.

[5] J.M. Long, E.A. Irani, D.W. Hunter, J.R. Slagle, J.P. Matts, et al. "Using a symbiotic man/machine approach to evaluating visual clinical research data," *Journal of Medical Systems*, **12** (1988) 327–339.

[6] J.M. Long, J.R. Slagle, A.S. Leon, M.W. Wick, J.P. Matts, J.N. Karnegis, J.K. Bissett, H.S. Sawin, and J.P. Stevenson. "An example of expert systems applied to clinical trials: Analysis of serial graded ECG test data," *Controlled Clinical Trials*, **8** (1987) 136–145.

[7] J.M. Long, J.R. Slagle, M. Wick, E. Irani, J. Matts, and the POSCH group. "Use of expert systems in medical research data analysis: The POSCH AI project. In *Proceedings of the National Computer Conference*, **56** (1987) 769–776.

[8] Michael C. Mozer. "RAMBOT: A connectionist expert system that learns by example," In *IEEE First International Conference on Neural Networks, Vol. II*, (1987) 693–700.

[9] Carsten Peterson and James R. Anderson. "A mean field theory learning algorithm for neural networks," *Complex Systems*, **1** (1987) 995–1019.

[10] D.E. Rumelhart, J.L. McClelland, and the PDP Research Group. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition Vols. I and II* (MIT Press, 1986).

[11] J.R. Slagle, J.M. Long, E.A. Irani, J.P. Matts, D.W. Hunter, and M.G. San-Marco. "ESCA: A success story of an expert system for evaluating visually-derived data," *A.I. Memo* UM-CS-AI87-6, University of Minnesota, Computer Science Department, Minneapolis (1987).

[12] James R. Slagle, Michael R. Wick, and Marius O. Poliac. "AGNESS: A generalized network-based expert system shell," In *Proceedings of the NCAI* (AAAI, 1986) 996–1002.

[13] Donald A. Waterman. *A Guide To Expert Systems*. Addison-Wesley, Reading, MA, 1986).