유전자 알고리즘을 이용한 신경망 설계

박 정 선†

요 약

본 연구는 보험 회사의 파산 예측을 위하여 신경회로망이 사용되는데 이를 최적화하기 위하여 유전자 알고리즘이 사용된다. 유전자 알고리즘은 최적의 네트워크 구조와 매개변수들을 제시해 준다. 유전자 알고리즘에 의해 설계된 신경회로망은 파산 예측을 함에 있어 discriminant analysis, logistic regression, ID3, CART 등과 비교되는데 가장 좋은 성능을 보여준다.

Designing Neural Network Using Genetic Algorithm

Jeongsun Park†

ABSTRACT

The study introduces a neural network to predict the bankruptcy of insurance companies. As a method to optimize the network, a genetic algorithm suggests optimal structure and network parameters. The neural network designed by genetic algorithm is compared with discriminant analysis, logistic regression, ID3, and CART. The robust neural network model shows the best performance among those models compared.

1. Introduction

Recently, neural networks have shown good performance in solving a variety of problems. The neural networks are known to be useful in applications which require robust and non-linear models. Recent developments in hardware and software made the neural networks partially overcome problems (e.g. computational inefficiency, difficulty to optimize network structure, and less persuasiveness to management), so that they are even more useful.

The strong points of non-linearity, adaptability, and robustness of neural networks all make the analysis of optimal network structure and parameters difficult. Advances in the theory cannot provide enough knowledge to adjust the structure and the parameters. We usually get some rules of thumb from empirical studies. Some previous works point out the problem in using neural networks. A paper [8] says that "we have performed some exploratory experiments to decide on the configuration of the neural nets." Similar remarks are also made by another paper [7] "The major limitation encountered when using NeuralWorks Explore for this application was its inability to support the numerous experiments required to find a satisfactory combination of network architecture and set of learning parameters."

In this study, we present a heuristic optimization

※본 연구는 명지대학교 산업기술연구소의 '97 교내 연구비 지원으로 수행된 결과임.
† 정 회 원: 명지대학교 산업공학과
논문접수: 1996년 3월 26일, 심사완료: 1997년 7월 30일

technique of the network : a genetic algorithm. The genetic algorithm is suggested because : It is successful without extensive knowledge of the domain in accumulating good solutions and rejecting poor ones. Even for highly complex problems, the genetic algorithm can be quickly prototyped once an acceptable representation and a performance measure are developed.

There are recent studies which synthesize genetic algorithms and neural networks. Harp et al. [4] use a genetic algorithm to optimize the design of the neural network for sine function. Powell et al. [5] try to optimize the design of the network for engineering tasks. The above studies show that genetic algorithms are helpful in the designing the neural networks. In this study, we will apply the genetic algorithm to the design of the optimal neural network for the bankruptcy prediction of insurance companies.

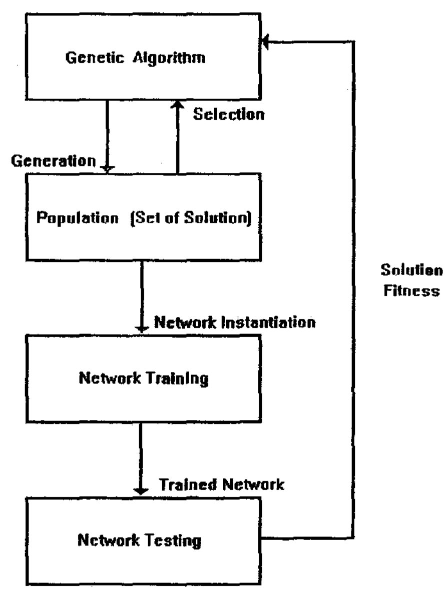
The purposes of this study are : 1)application of the neural network to the business domain of insurance 2)exploration of the design of the network using the genetic algorithm in the domain of insurance 3)demonstration of the superior robustness of the neural network combined with the genetic algorithm as compared with the other models in the bankruptcy prediction of insurance companies.

2. Design of the Neural Network Using the Genetic Algorithm

In most applications using neural networks, it has been observed that the performance of a neural network depends on the structure and the parameters of the network [4]. Our study will attempt to optimize the neural network by using the genetic algorithm to find a local optimal network structure and values for the parameters.

Figure 1. shows the procedure of how the neural network is optimized by the genetic algorithm. A network is represented by a solution of a population. Each solution represents a number of characteristics of the network such as the number of hidden units for a layer and learning rate. When a solution is generated by the genetic algorithm, it does not have a fitness value. The generated solution is used to construct a network. The instantiated network is then trained by backpropagation method using a training set of observations. After training, the network is tested by a test set of observations. The output of the test will be the fitness of the network. This process of instantiation, training, and testing is repeated for each generated solution of the population. When the number of the solutions in the population reaches the predetermined size, the genetic algorithm stops generating a new solution.

When a population finishes being evaluated, a new population is produced. To make the new population, two solutions of the old population are selected in proportional to their fitness. The two selected solutions are altered by using two genetic operators. The most prominent operator crossover is used to make two offspring from two parent solutions of the population by splicing parents. The other operator is a mutation



(Fig. 1) Design of the Neural Network Using the Genetic Algorithm which is used for generating a new solution by randomly changing one bit in an existing solution.

Each created solution is used to construct a new network. The new network is then trained and tested to make a fitness value for the solution. Whenever one solution is evaluated, the overall process checks stop criteria. If a criterion is satisfied, the process stops. Otherwise, the process returns to the procedure to make a new population. The process repeats until a stop criterion is satisfied. In this model, backpropagation method [6] is used for the training of the network and the genetic algorithm [3] is used for the optimization of the network.

To optimize the number of hidden nodes, the learning rate, and the momentum, we use the genetic algorithm. We want to change the number of hidden nodes from 2 to 9 based on the rule of thumb that the number of hidden nodes should be 75 percent of the number of input nodes [7]. Generally speaking, too many hidden nodes may reduce the ability to generalize the network; it may show good performance in a training data set, but week performance in a test data set. For the optimization of the number of hidden nodes, three binary digits are required.

The optimal number of learning rate is the maximal number which does not cause oscillation [6]. In this study, we use eight values between 0.1 and 1.5: 0.1, 0.3, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5. Harp et al. [4] suggest values between 0.1 and 12.8. However, some prior experiments showed oscillation when the learning rate has a high value. To code these eight values for the genetic algorithm, three binary digits are necessary.

The purpose of the momentum is to give a stability to the weight space of the network. We use four values for the momentum: 0.1, 0.3, 0.5, and 0.7. We need two digits for this purpose. Therefore, we need eight binary digits to optimize the neural network. In this study, we use a neural network with one hidden layer, changing number of hidden nodes, changing learning rate, changing momentum, and full connection between adjacent layers. The input variables are nine financial indices and the output variable is the default prediction of insurance companies.

3. Experimental Design

We have two prediction periods: one-year ahead prediction (one-year) and two-year ahead prediction (two-year). The methods for comparing the models are the training sample method and the independent test sample method. To test the model, the training sample method uses the same data that were used to train the model. The independent test sample method uses a separate test sample which is different from the training sample. In each experiment that uses one-year data or two-year data, 56 failed and 56 non-failed insurance companies are used for training, and 36 failed and 36 non-failed for testing.

Other aspects we should consider except the three aspects (number of hidden units, learning rate, and momentum) for the performance of a network are the number of cycles and initial weights. A cycle is defined as presenting all the observations in the data set to the network. Depending on the number of cycles, the accuracy of a network may change. If the accuracy of a network increases as the number of cycles, a higher number of cycles may give a better performance of the network. Sometimes, more cycles may decrease the performance of the network. To determine the number of cycles, we did some prototype experiments. Because we found that the error values become stable around 7000 cycles, we use that figure as the number of cycles. The other factor affecting the performance of the network is the initial weights. We have decided that it is not appropriate to manipulate the initial weights; because there are several ways to assign initial weights and we can't predict the results of them. By randomization of the initial weights, we have avoided a bias due to the wrong assignment of initial weights.

4. Empirical Results

In this experiment[1], the five models are compared: DA, CART, ID3, LR, and NN (designed by genetic algorithm). Of the five models, only two models (DA and CART) can adopt priors and misclassification costs. Therefore, the type I and II error rates and the average rate are calculated as measures.

4.1 Neural Network

In this experiment, the size of the population pool is set at 20 and the number of trials at 10. Therefore, the first 20 strings are generated randomly and not affected by the genetic operators. The next 10 strings are generated using the genetic operators.

We use the sum of I(test) and II(test) as the performance measure. The performances of the networks are compared with those of logistic regression. Logistic regression shows 21 type I or II errors in one-year and 26 errors in two-year.

〈Table 1〉 Performance Procedence Rate of NN over LR

	randomization	genetic operators
one-year data	25%	30%
two-year data	45%	60%

In Table 1, 5 times out of 20 (25%) show that the networks perform better than or equal to logistic regression in one-year data and randomization method; 3 times out of 10 (30%) in one-year data and genetic operators method; 9 times out of 20 (45%) in two-year data and randomization method; 6 out of 10 (60%) in two-year data and genetic operators method. The precedence rates by the genetic operators (30% in one-year and 60% in two-year) are little higher than those generated by randomization (25% in one-year and 45% in two-year). These results show that the genetic operators are helpful in the design of the neural networks.

4.2 Comparisons of the Five Models

When training sample method using the one-year is used, CART shows the best performance (0) and is followed by ID3 (0.036), NN (0.116), DA (0.264), and LR (0.277), sequentially. When test sample using the one-year is used, NN shows the best performance (0.179) and is followed by CART (0.278), LR (0.292), DA (0.347), and ID3 (0.403) by sequence. The above results are described in Table 2.

When the training sample using the two-year is used, CART shows the best performance (0) and is followed by, ID3 (0.107), NN(0.188), LR (0.268), and DA (0.385) sequentially. When the test sample using the two-year is used, NN shows the best performance (0.223) and is followed by LR (0.361), DA (0.375), CART(0.375), and ID3 (0.403) by sequence. The above results are described in Table 3.

If we look at the results using the test sample method (tables of 2 and 3), NN shows the best per-

〈Table 2〉 Misclassification Rates Using One-Year

model	type I (train)	type II (train)	average (train)	type I (test)	type II (test)	average(test)
DA	0.438	0.089	0.264	0.500	0.194	0.347
CART	0	0	0	0.194	0.361	0.278
ID3	0.071	0	0.036	0.528.	0.278	0.403
LR	0.339	0.214	0.277	0.361	0.222	0.292
NN	0.125	0.107	0.116	0.278	0.083	0.179

1 CART and ID3 are written in Pascal and run on an Encore Multiprocessor machine. DA and LR are run by SAS on an IBM machine. NN and GA packages are run on a Sun4 workstation.

〈Table 3〉 Misclassification Rates Using Two-Year

model	type I (train)	type II (train)	average (train)	type I (test)	type II (test)	average(test)
DA	0.357	0.413	0.385	0.694	0.056	0.375
CART	0	0	0	0.528	0.222	0.375
ID3	0.071	0.143	0.107	0.306	0.500	0.403
LR	0.339	0.196	0.268	0.639	0.083	0.361
NN	0.304	0.071	0.188	0.306	0.139	0.223

formance over the other models. LR shows relatively good performance: third in one-year and second in two-year. LR is better than DA in both years, confirming the previous results that LR has a better performance than DA. CART gives a good performance (second) in one-year but a fair performance in two-year (third). DA shows a fair performance of fourth in one-year and third in two-year. ID3 shows a relatively poor performance of fifth in both one-year and two-year. Generally, three methods (NN, LR, and CART) using the sigmoidal function or the pruning method show good performances; one method (DA) with the assumptions gives a fair performance; the other method (ID3) which builds trees recursively but which does not prune them shows weak results.

5. Conclusions

In this study, we introduced the neural network model in the bankruptcy prediction of insurance companies. The robustness of the neural network motivated us to apply the model in the bankruptcy prediction. In this study, the five models were compared. Previous works explained the problem using neural network to be the difficulty in optimizing the structure and the parameters. To overcome this problem, we introduced the genetic algorithm. The results showed that neural network has the best performance among the models. Additionally, the use of the genetic algorithm was shown to be helpful in finding the optimal structure and parameters of a neural network.

References

[1] T.B. Bell, G.S. Ribar, and J.R. Verchio, "Neural Nets vs. Logistic Regression: A Comparison of Each Model's Ability To Predict Commercial Bank Failures," 1990 Deloitte & Touche/University of Kansas Auditing Symposium, May, 1990.

[2] E. Collins, S. Ghosh, and C. Scofield, "An Application of a Multiple Neural Network Learning System to Emulation of Mortgage Underwriting Judgements," Proc. of the IEEE Int. Conf. on Neural Networks, 2, 1988, 459-466.

[3] D. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning," Addison-Wesley, Reading, MA, 1989.

[4] S.A. Harp, T. Samad, and A. Guha, "Designing Application-Specific Neural Networks Using the Genetic Algorithm," Advances in Neural Information Processing Systems, 2, 1990, 447-454.

[5] D.J. Powell, S.S. Tong, and M.M. Skolnick, "ENGENEOUS: Domain Independent, Machine Learning for Design Optimization," Proc. Third Int. Conf. on Genetic Algorithms, 1989, 151-159.

[6] D.E. Rumelhart, G. Hinton, and R. Williams, "Learning Representation by Back-propagation Errors," Nature, 323, 9, 1986, 533-536.

[7] L.M. Salchenberger, E. M. Cinar, and N. A. Lash, "Neural Networks: A New Tool for Predicting

Thrift Failures," Decision Sciences, 23, 1992, 899-915.

[8] K.Y. Tam and M. Kiang, "Neural Net Applications in Bank Failure Predictions," Manegement Science, 38, 1992, 926-947.

박 정 선

1983년 서울대학교 산업공학과 졸업(학사)
1985년 한국과학기술원 경영과학과 졸업(석사)
1993년 미국 Univ. of Texas at Austin 경영정보시스템 전공(박사)
1985년~1988년 금성소프트웨어 근무
1988년~1989년 한국유니시스 근무
1993년~1995년 한국전산원 근무
1995년~현재 명지대학교 산업공학과 조교수
관심분야:AI/Expert System, CALS, WWW 응용