Association for Information Systems
AIS Electronic Library (AISeL)

PACIS 1995 Proceedings

Pacific Asia Conference on Information Systems (PACIS)

December 1995

Evolving Pattern Detectors for Predicting Retention in Managing Student Admissions

Alvin Surkan
University of Nebraska-Lincoln

Alexei Skurikhin
Institute of Physics and Power Engineering

EVOLVING PATTERN DETECTORS FOR PREDICTING RETENTION IN MANAGING STUDENT ADMISSIONS

Alvin J. Surkan
Computer Science and Engineering Department
University of Nebraska-Lincoln, NE
68588-0115 USA Fax: 402-272-7767
Email: surkan@cse.unl.edu

Alexei N. Skurikhin
Mathematics Department
Institute of Physics and Power Engineering
249020 Obninsk, RUSSIA Fax: 7 95 230 2326
Email: alexei@cycoruss.obninsk.su

Abstract. *Student's responses to psychological questionnaires can be evaluated by classifiers designed to detect predictive patterns before new applicants are admitted to a school. Individual item responses and aggregate background scores provide information useful for predicting student's long-range performance. Historical records on previously admitted students and test performances can provide data for evolving of classifiers. The classifiers can be based on knowledge either in optimized population of data patterns or in the connection weights of neural network models. Expected accuracy in classification is monitored by splitting historical data into training and test sets. For management, the role played by these classifiers is to improve decision making to enhance the ratio of the graduating students to those who terminate before completion.*

Key words: retention, predict, admission, pattern, student.

1 The Student Retention Problem

There is a need for more effective criteria for screening students being accepted at academic institutions. Deciding to accept an applicant requires a valid assessment of the probability that a student will be graduated. To improve decision making, there are several methods for evolving both algorithmic and non-algorithmic classifiers.

There is now a growing basis for building computer systems that learn to classify and predict, using methods of statistics, neural networks, and expert systems [Weiss and Kulikowski 1987]. Classifiers can be derived from training data representative of students for whom performance has been established. Training data are collected from prospective students during the application process.

Classifiers may be modeled by neural networks as populations of patterns of weights or chromosomes that are improved by genetic operators. Several methods for evolving classifiers and evaluating their relative performance were tested on a data base of 200 patterns with 39, 40 and 41 variables.

2 Source of Classifier Knowledge

Typically, text-based psychological instruments designed for data collection have a few tens of items. In the available data, there were 39 items that were assigned binary values. The instrument items are designed to elicit responses which produce information necessary for building a classifier. Each response is individually scored. The scores are used in parallel to establish the potential of a particular school applicant to be retained through graduation. Evolving effective classifiers from psychological data can increase the proportion of students who complete their studies, allowing the administrators more selective in their admissions, and thereby solving the problem of retaining students.

3 Desiderata for Classifier Paradigms

It is important to develop a classifier that uses data in parallel. Typically, data variables have different sensitivities and may depend nonlinearly on the values of other variables in each instance of a pattern. The classifier must assign patterns to one of two classes of students (1) retained and (2) not-completing. To build a classifier from training data, the patterns from student application records are paired with historical logs of graduation or failure.

4 Problem and Data Description

Currently, many students begin their program of study and can not be retained to graduation. Costly waste of resources can be reduced if data available about potential students can be used successfully to favor admitting a greater proportion of students with a higher probability of being graduated. The retention problem requires a more refined application of the information available before admission. Minimizing the number of dropouts enhances the proportion being graduated.

Two main types of data are available from applicants. These data include the individual item scores and the aggregated scores. The later reflect an average over the collection of item scores or performance over a distribution of the same items spanning a period of time. One example of such an aggregate score is a student's high school grade point average.

5 Data Acquisition & Response Scoring

Psychological information is collected by a uniformly controlled presentation of a sequence of questions or items and recording the response to each. After the responses are recorded as text, the score for each item can be evaluated and assigned a binary value of zero or one, depending on whether or not an item satisfies prescribed, fixed criteria.

6 Partitioning of Training and Test Data

When a classifier is to be evolved and evaluated from an over-abundance of historical data, making decisions on how much of the data is sufficient for training is not difficult. One can simply use the largest training set sample that can be processed with the computer and available time. Then from the balance of the data, one can randomly select a sample of test patterns that is large enough to bring the confidence interval on the performance measure into an acceptable range.

7 Artificial Neural Net Algorithms for Parallel Distributed Processing PDP

Neural networks are tolerant of uncertainty and can operate with online observations to identify the class of sample patterns. These networks can learn the equivalent of rule information from examples of training patterns of many variables. Explicit coding of process knowledge as a set of symbolic rules is unnecessary.

The most common type of layered neural network model, the [Rummelhart 1986], has an input processing layer which feeds the first one or more hidden layer(s). The final active layer processes intermediate signals from the second-to-last layer to activate the output.

Fully connected, recurrent neural networks include backwards directed and/or lateral feedback connection weights. They can take the form of memory neural networks or MNNs [Unikrishnan et al 1991, 1993]. In MNNs each node has attached one or more memory neurons to produce lengthy response delays. In the training phase, the cumulative or net effect of the time-delay neurons is to learn or encode the responses that incorporate multiple time delays.

8 Alopex: Correlation-based Simulated Annealing

Simulated annealing using the Boltzmann distribution originated with the Metropolis algorithm for function optimization. Alopex [Unikrishnan et al 1991] is the name given to an algorithm that is capable of evolving improved neural network models by stochastically selected modification of weights. Alopex differs fundamentally in principle from other algorithms for systematically improving a solution model on the basis of input patterns and corresponding known outputs. Because it is more dependent on random excursions taken in solution space, Alopex has more of an evolutionary flavor than other deterministic hill-climbing approaches to model building. One architecture for an Alopex algorithm optimized network model has one or more fully connected hidden layers with node that also includes self-loops. The exponential term in this distribution depends on a measure of the correlation in the error changes and the weight change over the previous time period.

One specific example of a successful architectural structure has one output, two hidden layers of 5 neurons each, and these are fed by the 39 binary inputs that correspond with values of scores assigned to the 39 responses to items of a psychological questionnaire.

9 Memory Neural Network (MNN)

The architecture of a memory neuron network is multi-layer with recursion and internal time delays supported by feedback paths associated with one or more memory neurons attached to each ordinary network neuron. Each input node and every ordinary neuron of the hidden layers has attached a memory neuron through a time delay. Also, the inserted memory neurons all have self-loops with a time delay. The output layer neurons can have delays attached to more than one memory neuron in series and each delay is introduces in parallel to the host network neurons to which they are linked. There is no connection between network neurons within layers. The internally added memory neurons all have an output to every node in the following layers. Optimization of the weights connected to all networks on memory neurons can be accomplished by a modification of error backpropagation algorithms or evolution with a genetic algorithm.

9 Cascade Correlation (CASCOR)

The architecture of CASCOR [Fahlman 1988, 1990] has several advantages over that of the BACKPROP and QUICKPROP algorithms. CASCOR automatically determines the size and topology of the network. This method also preserves the structure it builds even after subsequent changes in the training set, and requires no back-propagation of error signals through the connections of the network. CASCOR is a supervised learning method that builds a significantly minimal, multi-layer, network topology during the training. Initially, the network contains only inputs, output units, and the connections between them.

10 General Description of LVQ Methods

Learning vector quantization or LVQ [Kohonen 1989] is a classical method for producing an approximation to a continuous probability density function p(x) of the vector input variable $\mathbf{x}$. This density function uses a finite number of codebook vectors $\mathbf{m}_i$, i = 1, 2, ... k. On specifying the "codebook", approximation the output for the input $\mathbf{x}$ vector requires finding the reference vector $\mathbf{m}_c$ closest to $\mathbf{x}$. An optimal placement of the $\mathbf{m}_i$ minimizes E, the expected value for r^{th} power of the reconstruction error:

$$E = \int_{\mathbf{x}} |\mathbf{x}-\mathbf{m}_c|^r \cdot p(\mathbf{x})\, d\mathbf{x}, \quad |\mathbf{x}-\mathbf{m}_c| = \min_i \{|\mathbf{x}-\mathbf{m}_i|\} \quad (5)$$

In general, no closed-form solution is possible for optimally placing $\mathbf{m}_i$. Consequently, one must resort to

resort to iterative approximation schemes. The LVQ algorithm uses vector quantization to define directly the class borders with the nearest-neighbor rule.

When the LVQ algorithms are used, the classification accuracy achievable and the learning time depends on the following factors: (a) approximately the optimal number of codebook vectors assigned to each class and their initial values and (b) the detailed algorithm, its learning rate schedule during training steps and some appropriate stopping criterion.

10 Results of Experimental Trials of Classifiers Produced from Training Data Using Alternative Methods

BACKPROP, QUICKPROP and ALOPEX were used for learning a classifier for predicting the binary or integer output values used for designating class in the set {0,1} or in the interval [0-8700]. ALN (adaptive logic networks) and LVQ were used to produce classifiers for the binary output. Results were obtained for the data by the six methods: (1) BACKPROP, (2) QUICKPROP, (3) ALN, (4) ALOPEX, (5) LVQ1 and (6) LVQ3. Results from the first three methods were inferior to those of the LVQ algorithm.

The best results were obtained using the LVQ3 algorithm. The accuracy, in terms of the number of correct classifications, obtained with a test set of 100 patterns is 61%. This level of performance was observed by using either 39 or 40 input variables, independent of whether the output classes being predicted were binary or integer. Two versions of the LVQ algorithm derived acceptably performing classifiers from a set of training samples. Unlike the BACKPROP, QUICKPROP, ALN and ALOPEX, which gave highly nonuniform classification results, the accuracies of LVQ results for the two classes were similar at 57 to 64 %, respectively.

For predicting binary output classes, the LVQ accuracy is superior to that of BACKPROP, QUICKPROP, ALN and ALOPEX. These four learning algorithms tend to get trapped at local minima. The undesirable tendency characterizes the behavior of these gradient and random search methods, namely: BACKPROP, QUICKPROP, ALN and ALOPEX, to produce spurious results.
In a series of experiments, with values of ϵ between 0.1 and 0.5 and values of α between 0.01 and 0.03, a window size between 0.1 and 0.4 was found effective. The optimal value of ϵ depends on the size of the window. A smaller window size is more optimal. After the definition of the codebook vectors, finding acceptable values of the parameters ϵ, α, and window size are formed through an iterative process.

The 100 training patterns had an equal number in each of the two classes. The test set of 100 patterns was split between the two classes as 33 to 67. The number of codebook vectors varied over the range 20 to 50. The number of training iterations ranged from 4,000 to 10,000. Since the level of performance obtained on the training set was 78 and 88 %, respectively, for the two classes with an average of 83%, it appears probable that the psychological questionnaire data does not have enough discriminating information to support better classifier performance.

11 Conclusions

The difficult problem of automatically creating a classifier from training samples of psychological data has explored by making a comparison of a number of evolutionary algorithms. These algorithms capture the equivalent of classification rule information from training examples. It is concluded the most reliably performing classifier was obtained by iterative generation of variants of LVQ, the learning vector quantization method.

12 References

[1] Davis, Lawrence 1991 "Handbook of Genetic Algorithms" edited by Lawrence Davis (Van Nostrand Reinhold)

[2] Fahlman, S.E. (1988) "An Empirical Study of Learning Speed in Backpropagation Networks", June 1988, Technical Report, CMU-CS-88-162. Computer Science Department, Carnegie Mellon University, Pittsburgh, PA.

[3] Fahlman, S.E., Lebiere, C. (1990) "The Cascade-Correlation Learning Architecture", February 1990, Technical Report, CMU-CS-90-100. Computer Science Department, Carnegie Mellon University, Pittsburgh, PA.

[4] Rumelhart, D.E., Hinton, G.E., and Williams, R.J. (1986) "Learning Internal Representations by Error Propagation", In: Parallel Distributed Processing, MIT Press, Vol.1, pp.318-362.

[5] Unnikrishnan, K. P. and Venugopal, K. P. (1993) "ALOPEX: A correlation-based learning algorithm for feed-forward and recurrent neural networks" R & D Publication GMR-7919, GM Technical Center, MI. 48090-9055

[6] Unnikrishnan, K. P. and Venugopal, K. P. (1992) "Learning in connectionist networks using the ALOPEX algorithm" Proceedings of the International Joint Conference on Neural Networks IJCNN Part I, pp. 926-931.

[7] Unnikrishnan, K. P., Hopfield, J. J. and Tank, D. W. (1991) "Connected-digit speaker-dependent speech recognition using a neural network with time-delayed connections" IEEE Transactions on Signal rocessing. 39, pp. 698-713.

[8] Wiess, S. M., and Kulikowski, C. A. (1987) "Computer Systems that Learn Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning and Expert Systems". Morgan Kaufmann, 1997.