Time Based Experiential Learning on a Neural Network Classification System

Brad Morantz, Decision Science Dept, Georgia State University, Atlanta
Tushar Dave, Computer Science Dept, Georgia State University, Atlanta

Time Based Experiential Learning On a Neural Network Classification System

Brad Morantz Department of Decision Science, Georgia State University, Atlanta GA 30303 USA
Tushar Dave Department Of Computer Science, Georgia State University, Atlanta GA 30303

Abstract

Neural networks are used for classification on a variety of applications. They base these judgements on training, done most often with historical data. The world is dynamic as things are constantly changing. Therefore, it becomes necessary to adapt to these changes. Some rules become obsolete while new ones are introduced.

Using C++, a Boltzmann style artificial neural network was implemented. It supervised learning using back propagation. Several dynamically sized linked lists are used to store data. One has the training data, and another tracks past classification results in simulated real time. An additional database needed only for this research is used for tracking 'hit rate' accuracy.

The initial testing was done with test data, and then real world data was used. The update time was varied to understand its affect on the accuracy. This time value was not a constant, but rather a function of the data set. The results will be compared with those from a non-learning neural network system, and a standard regression package. The results will be compared using ANOVA to see if there are statistically significant differences resulting from improved classification with the learning system.

Keywords: Neural Networks, Machine Learning, and Classification

Background

The human brain is the model for artificial intelligence system design. [Anderson] [Tveter] We gain knowledge through personal observation and experience. [Fu] We try to learn how it works and then emulate its methods and techniques. The biological brain is massively parallel

and by utilizing large amounts of data it is able to recognize patterns. [Medsker] Because of its multiple layers, it is possible to represent nonlinear systems.

Current artificial neural networks learn either by supervised training with a data set that includes the answer, or by unsupervised organization of the data. [Kosko in Chester] [Haykin] Once they are trained, they will be used to make predictions, to classify, or to recognize patterns. [Dharr & Stein] Currently, a neural net does not improve its knowledge from experience over time.

Neuro-genetic hybrids have been implemented in such a manner as to improve their performance by using a genetic algorithm to optimize the weights. These systems are very effective, but still rely upon the original training data. [Stroud] [Williamson]

In a biological neural network, knowledge is often created by learning from mistakes and successes. One commonly makes a decision or prediction about an event to happen, and then afterwards looks back and proclaims either “next time I will know better” or “aha, I was right!” This is continuing to learn from experience. This is knowledge discovery, and a nontrivial chunk of knowledge has been acquired. Knowledge discovery is the learning of implicit or previously unknown knowledge from data and observations. [Fu]

The ability to discover this knowledge has become increasingly important both in life and in AI research. Applications range from business to military to science. The utilization of this could increase accuracy in medical diagnosis and hence save countless lives.

Introduction

Artificial Neural networks are commonly used for classification problems. Typically, they are trained on historical data and then presented with data for prediction or classification. Automated knowledge discovery has become a very important topic in the realm of AI research [Fu].

Life is dynamic, and new facts become truisms continuously, and at the same time, old facts or rules become obsolete. It is unwise to make judgments at one point in time based upon rules from another time. This set of rules, or knowledge base, must be as dynamic as life itself,

constantly changing, updating with new facts and deleting the old and now untrue (or no longer applicable). This is the goal of this project, to create an artificial neural network classification system that dynamically updates its knowledge base, adding the new and deleting the obsolete.

The simplest description is that this system should classify input data with as close to 100% accuracy as possible. In our research form, it also maintains a database for tracking 'hit' rate and tracking its accuracy. It does this by maintaining a database of every file upon which classification is requested. It will store a file number (as a key), a date originally processed, new data indicating accuracy of classification, and date acquired.

System Requirements

The code was written in C++, and compilations will be available for both Windows and Linux/Unix operating systems. Processor speed will affect the wait time for an answer, and will only be critical in real time applications. Because of the number of calculations, it would be unwise to run this program on anything less than a Pentium 100.

The amount of memory required will be a function of the number of variables and the size of the data set. Bare minimum for a Windows OS is 24 to 32 megabytes, but more would improve performance. If the data set is large, 64 megabytes is recommended. While Linux is not as memory hungry as Windows, the same amounts of memory would be wise. In light of the constantly attenuating prices in this realm, it should not present itself as a problem. In fact, these configurations are quite common.

An extension to this program will be a multi-processor version, to run on Linux or Unix platforms. A neural network is an inherently parallel process, and installation in this domain would result in tremendous gains in performance and processing speed. True parallel processing could also allow feedback neural configuration and simultaneous equation solution, providing faster and more accurate results. It could also emulate highly correlated variables and their effect. Because of the design of this program, a shared memory model machine must be used.

The system is designed totally modular or object oriented. Fortran 90 or HPF are both functionally adequate to accomplish this task. While they are more powerful mathematically, C++

forces strict object oriented programming. Additionally, C++ is the language of choice of our programming team. Therefore this project was programmed in C++.

The neural network is an off the shelf freeware program, modified for this application. It is a feed forward design, and learns via supervised back propagation. An off the shelf program was chosen because it saved considerable programming time. Eventually, if this project is successful and continues, a custom version that will better interface with the other modules will be implemented.

There are databases for the following: training data, cases supplied for classification, and hit rate to track accuracy. This last database is only for research purposes, and will be optional to the commercial customer (where management might wish to track performance). All of these databases will be implemented using linked lists.

A main control system monitors the entire process and controls operation. This controls the update process, the retraining, the statistics section, and the Input/Output. The former is the process of receiving new information about past classifications. Retraining is using this new information to improve the knowledge base. Statistics are purely for research purposes, to prove that this method is superior to other classification methods. A management team might wish to have this data to track departmental performance.

A statistics module will be needed to analyze this data. There are countless ones available as freeware from universities all over the globe. It will save much time to find one of these that is written in C++. With the Internet, this should be readily available.

The goal is to utilize as much freeware or open source material as possible, and modifying where necessary. The contribution of this project is not any fantastically new techniques, but rather a unique combination that will allow dynamic learning of an artificial neural network for improved classification accuracy.

Test Methods

This program was designed with a single predicted output, and therefore can be compared easily to mean square regression. The output is defined as follows: zero represents bad credit risk; and one, good credit risk. While neural networks are adept at a non-linear

response, we will accept 0.2 or less as bad credit, and 0.8 and greater as a good credit risk. The values between these two will be considered equivocal, requiring further investigation.

Some credit grantors might consider a value close, but still not in the accepted range, as a high-risk customer and assign a higher interest rate. We will consider this option in our data and allow for just such an occurrence. These fuzzy gray areas might possess good information for our test methods.

This regression method will be used to test the predictions. The regression coefficients will be calculated from the data in the initial training sets. The multivariate equation will then be employed to forecast the unknown ratings. The 'News' about any previous prediction will not be utilized for forecasting.

The predictions will be compared against the later information, and a hit rate table will be created. This will be done for several data sets, to eliminate the possibility that any one set of data might be more continuous or monotonic, and possibly better suited for regression techniques. A mean hit rate for each set can be calculated.

The next data set will be obtained by turning off the learning function of the neural net program , so that it will operate as an ordinary artificial neural network, and only learn from the original training data. The forecasts will be made, as in the regression model, and the hit rates calculated. This will be done for the same series of data sets, and average hit rates for each set will be computed.

This same procedure will be followed a third time, but this time with the learning neural network classifier. A third table of average value hit rates will be obtained.

Having three (3) sets of average values, from two (2) known and respected methods, and a third method under test lends itself to analysis of variance (ANOVA). SPSS or SAS will be used to compare these three sets of hit rates and determine conclusively if there exists statistical difference between the three methods of prediction. Tukey's multiple comparison procedure can then identify the group(s) that has a different mean hit rate.

Time Base

Another area of research, closely related to this testing, is determining the time base for updating the knowledge base. One train of thought is to keep it small, as any rule that is still current will re-present itself continually. Keeping it too short might remove too much knowledge or experience, and excessively long might attempt to enforce obsolete rules. The volume of data and cases supplied, and the particular area of prediction will affect the time interval selected.

For a chosen area of interest, we will try the same data sets with various update times. Again, we will record the average hit rates and then employ ANOVA to determine the differences, if any. Tukey Multiple Comparison Procedure can then identify which update time base produced the most accurate results, in terms of highest hit rate.

Application for Testing

Credit granting is an area where these methods of prediction have been both utilized and scrutinized for years. The impact of a wrong prediction can expensive, either due to lost reimbursement or opportunity loss. It is possible to search for patterns or relationships that might indicate future financial repayment trends. [Williamson] Because of the financial magnitude involved, continuous improvement is sought. Additionally, a large volume of data exists in this realm.

The initial input variables are:

1) Credit score i.e Fair Isaac Beacon score or Kaufman credit inquiry score
2) Number of years on the job
3) Annual income in U.S dollars
4) Total debt in U.S. dollars
5) SSN or a key number to maintain the file
6) Outcome (good or bad)
7) Date

All of the above are required for training, and all but the outcome are required for prediction. The neural net only needs the first four (4) variables, as the others are required for the dynamic learning function.

When information is received, at some point in time after the prediction, this is considered to be 'news'. This will either confirm a correct prediction, or indicate an incorrect one. In either case, it is a learning experience. This data needs to be introduced to the system:

1) The SSN or key number indicating the file
2) The date that the news occurred
3) The true outcome

The system will then find the original file from which the original prediction was made and then update it with this 'news', adding the date and the true outcome. Since it is no longer merely a prediction file, but now historical data, it will be moved form the classification file to the training file of historical data.

For this project, we have chosen zero (0) to represent 'bad' credit; and one (1), 'good' credit. Because the output of the net varies over the range of zero to one, we have decided to accept a prediction in the range of 0.0 to 0.2 as 'bad; and 0.8 to 1.0, as 'good'. The values between 0.2 and 0.8 are considered to be equivocal, and a definitive answer cannot be given without more information.

In our example, it is possible that some creditors might wish to extend limited credit, at a possibly higher interest rate, to the group with a predicted score in the range of 0.6 to 0.8. It is expected that much can be learned from these 'gray area' predictions.

Once a day, the dates in the training file can be examined, and those that are out of date can be deleted or moved to a history file for legal reasons. The recently modified files have been placed in this file as 'news' was received, and the system is now ready to re-train on this new and more up-to-date data base.

Future

The next logical step would be to implement a genetic algorithm for optimizing architecture and training, creating a neuro-genetic hybrid. [Williamson] [Sundarashen and

Condarcure] These types of system have shown to remove design responsibility from the operator and improve overall performance. [Whitley et al] Further, they outperform the manually adjusted system, and require less human intervention. [Williamson]

Application of parallel processing on appropriate hardware opens up a whole new world of possibilities. Expected improvements would be in both speed and accuracy. The concept of simultaneous solution of structural equations promises to allow more complex relationships and increased variable sets.

The knowledge base in the machine pertains to the current time and has been improved from the original rule base. Further extension of this research could be realized in utilizing rule extraction. The latest knowledge contained in the trained neural network contains the best and most up to date knowledge. It has also been shown that human intervention on the variable set can improve the performance of a neural network. [Gim]

In many instances, i.e. credit granting, government regulations require explanation of a decision. Rule extraction would provide this needed information. These rules can also provide simple heuristics and screening questions for first level interrogation. They could also be used as guides for human decision making.

References

Anderson, John R; *Cognitive Psychology and its Implications*, 2nd Edition, W H Freeman and Company, 1985: New York

Haykin, Simon; *Neural Networks A Comprehensive Foundation*, IEEE Press, Macmillan College Publishing Company, 1994:New York

Medsker, Larry R; *Microcomputer Applications of Hybrid Intelligent Systems*, Journal of Network and Computer Applications, No 19, 1996, pp 213-234

Medsker, Larry, and Liebowitz, Jay; *Design and Development of Expert Systems and Neural Networks*, Macmillan Publishing, 1993:New York

Pao, Y H; *Knowledge in the form of Patterns and Neural Network Computing*, in *Knowledge Engineering Vol I, Fundamentals*, edited by Hojat Adeli, McGraw-Hill Publishing Co, 1990: New York

Ruiz,A, Owens, D H, Townely, S; *Existence, Learning, and Replication of Periodic Motions in Recurrent Neural Networks,* v 9, no. 4, July 1998, pp 651-661

Schalkoff, Robert J; *Artificial Intelligence:an Engineering Approach*, McGraw-Hill Publishing, 1990: New York

Snelbecker, Glenn E; *Learning Theory, Instructional Theory, and Psychoeducational Design*, McGraw-Hill book Company, 1974: New York

Squire, Larry r; *Memory and Brain*, Oxford University Press, 1987: Oxford

Sundareshan, Malur K, and Condarcure, Thomas A; *Recurrent Neural Network Training by a Learning Automaton Approach for Trajectory Learning and Control System Design*, IEEE Transactions on Neural Networks, v 9, no. 3, May 1998, pp 354-368

Tveter, Donald R; *The Pattern Recognition Basis of Artificial Intelligence*, IEEE Press, 1998: Los Alamitos CA

White, Barbara Y, and Frederiksen, John R; *Causal Models as intelligent Learning Environments for Science and Engineering Education*, in *Causal AI Models Steps Toward Applications*, edited by Werner Horn, Hemisphere Publishing Corp., 1990: NewYork

Williamson, A G; *Refining a Neural Network Credit Application Vetting system with a Genetic Algorithm,* Journal of Microcomputer Applications, v 18, no. 3, July 1995, pp 261-277