ADAPTIVE DIMENSIONAL NEURAL-FUZZY TECHNIQUE FOR SPATIAL DATA

T.D. Gedeon[1], P.M. Wong[2], Y. Huang[1,2] and C. Chan[1]

[1]School of Computer Science and Engineering / [2]Centre for Petroleum Engineering
The University of New South Wales
Sydney NSW 2052 Australia
Tel: +61-2-93853965
Email: tom@cse.unsw.edu.au

ABSTRACT

In this paper, we propose a neural-fuzzy technique for spatial data interpolation. The methodology involves the use of neural networks to approximate functions through existing data points. The trained networks are used as fuzzy rules. The results of these rules are interpolated based on the relative location of the point-of-interest. We also demonstrate the use of this methodology in petroleum reservoir modelling where properties are estimated between two oil wells. The end result is a simple and computationally-cheap method for spatial data analysis.

KEYWORDS

Neural, Fuzzy, Interpolation, Approximation, Petroleum Reservoir, Oil Wells.

INTRODUCTION

In spatial data analysis, we often need to estimate values of the variables of interest at locations where samples are not available. These locations are usually referred to as the "unsampled" locations. We usually obtain these values by interpolating neighboring measurements (or hard data) with or without the help of secondary (or soft) information at the unsampled locations.

In recent years, artificial intelligence (AI) techniques have been successfully applied in many engineering disciplines. In many cases, these applications have been shown to provide significant improvements over conventional methods. AI techniques, such as neural nets and fuzzy logic, are widely used in pattern classification and function approximation. They are particularly useful in cases where the data is large in both amount and types of variables. Their applications to spatial data analysis, however, are few.

The objective of this paper is to introduce the integration of neural nets and fuzzy logic in spatial data interpolation. We will first review some basics in neural nets and fuzzy logic, followed by introducing our new methodology in data interpolation. We will particularly discuss the advantages of using multiple, adaptive number of inputs (or adaptive dimensional) nets, instead of the conventional single, multi-dimensional net approach. In the later sections, we will demonstrate the use of this technique in petroleum reservoir modelling. Data from two oil wells were interpolated to the location of a third well where output (or target) data were withheld from the interpolation algorithm. Performance of the technique was then evaluated by comparing the predictions and the target data in the third well.

NEURAL NETS

An artificial neural net is a computer model which attempts to mimic some of the workings of the human brain. It can learn from examples or experience, and is extremely useful in solving pattern classification and mapping problems. The following will briefly review the basic structure of a typical neural net, the mathematical algorithm to reduce the errors during training, and the issue on over-learning. More details can be found in Dayhoff (1990).

Basic Architecture

A typical neural net is composed of three kinds of layers: input, hidden and output layers. Figure 1 shows a schematic diagram of a neural net. Each layer consists of a number of processing elements or neurons. In this example, we have two input neurons, three hidden neurons and one output neuron which can be represented as a "2-3-1" network. All the neurons in a layer are fully connected, via weighted links, to the neurons in the preceding layer.

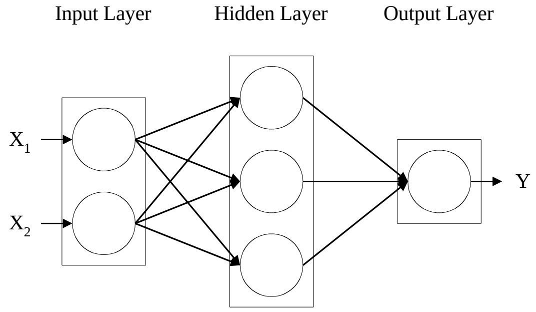

Figure 1. A schematic diagram of a neural network.

The number of neurons present in input and output layers is usually straightforward and can be determined by the particular application based on the data available and the nature of the task. The decision of the number of neurons (*H*) which should be present in the hidden layer, however, is difficult to determine *a priori* and is usually determined by trial and error.

Learning Algorithm

The backpropagation algorithm is the most widely used learning procedure for neural nets. This technique requires pre-existing training patterns, and involves a forward-propagation step followed by a backward-propagation step. The forward-propagation step begins by sending the input signals through the neurons of each layer until the output values are calculated. The backward-propagation step calculates the errors by comparing the actual and target outputs. New sets of weights are re-iteratively calculated based on these error values until a minimum overall error is obtained. The total sum of error squares (TSS) is a typical measure of the overall error of the trained neural net. The neural nets which use the backpropagation algorithm to minimise error are commonly known as backpropagation neural nets or BPNNs. After the training stage, the weights on all connections are fixed and the network is ready for solving the real problem it has been trained for.

Over-learning

One of the important issues in neural training is over-learning or over-fitting. Too many iterations will over-fit the training data and fail to generalise the data trend. Therefore, proper termination of training is required before the network generalisation capability degrades. One way to avoid over-learning is to validate the network during training by using a separate data set, namely a validation set. Thus, we terminate training when the minimum error on the validation set, instead of the training set, is reached.

FUZZY LOGIC

The major significance of the introduction of fuzzy logic and fuzzy sets is to simulate human ways of thinking in a formal manner better than previous logical tools of mathematics. Fuzzy logic provides a completely new means of modelling very complex or not very well defined systems. The combination of the classical discrete IF-THEN rule models applied in classical AI and the fuzzy logic for describing inexact statements concerning systems, leads to a new class of approximative models with an inherent interpolative nature that allows the evaluation of situations that are not identical with any of the antecedents in the knowledge base. This type of model was first proposed by (Zadeh, 1973). Computationally well tractable algorithms were proposed soon after (Mamdani and Assilian, 1975), and an alternative model fitting especially well to piecewise quasilinear systems was proposed (Sugeno and Takagi, 1985). The rule interpolation model and algorithm (Kóczy and Hirota, 1990, 1993) are suitable for treating sparse models with insufficient information, as well. This is a crucial difference from symbolic rule-based models, which use a complete cover of the universe of discourse, and the antecedents give an exhaustive list of the possible situations. This is of course why these systems are so expensive to produce and maintain.

Fuzzy rule bases usually refer to continuous universes, and contain knowledge that states something exactly for a certain domain of the input and output spaces, and states something that has less and less degree of truth farther and farther from the region of exact knowledge. This is usually expressed in terms of the membership function which ranges from 0 to 1 representing exact knowledge at the highest value, and lower values represent partial knowledge.

NEURAL-FUZZY TECHNIQUE

We use a number of simple separate fuzzy rule-bases at each well location which are therefore readily created and maintained, largely automatically using statistical techniques. The rule-bases are interpolated between existing well sites, producing new rule-bases for chosen (nearby) locations (Gedeon, 1996a). The collection of simple interpolated rule-bases is then used to produce the prediction of the petrophysical properties. The evidence from each rule is combined using the conclusions of each individually unreliable information source (Gedeon et al., 1995). Thus, we end up with a prediction of properties, and an explanation for each property value predicted based on the significance of each rule in the rule-base, and which can be related back to the real well rules we interpolated from.

The proposed neural-fuzzy technique for data interpolation consists of three steps: 1) pattern selection for neural learning; 2) extraction of fuzzy rules from neural nets; 3) fuzzy rule interpolation to obtain the final estimate.

Pattern Selection

Conventional neural training uses the number of available input variables to decide the number of neurons in the input layer. If there are *m* input variables, there will be a single network with *m* input neurons. This approach, however, may fail in practice as some input variables or combinations of variables may not contribute to, or may even reduce, the network performance (Wong et al., 1995).

In this paper, we use adaptive networks with various input variables in such a way that the bad combinations of variables are penalised (see later sections). All different combinations of inputs are considered. Hence, instead of a single multi-dimensional network in the conventional method, there will be C_n^m *n*-dimensional networks. Each of these will be trained independently.

To avoid over-learning, we will use a validation set to terminate training as discussed previously. For the purpose of simplicity, data from only two locations are considered (see later sections). Data from one location is used for training, and the other for validation. In order to reduce bias associated with the predictions, we will also swap the usage of the data for training and validation. Results are combined via fuzzy rule interpolation (see below).

Fuzzy Rule Extraction

Fuzzy rules are extracted from each of the adaptive-dimensional networks. Each location consists of *m* variables, resulting in a total of 2 times C_n^m IF-THEN fuzzy rules. Each of the fuzzy rules can be expressed as follows:

$$\text{Rule}_{ij}\text{: IF } x_{ij} = (x_1,\ldots,x_k) \text{ is } A_{ij_k} \text{ THEN } Y_{ij} = Nn_{ij}(x_1,..,x_k) \quad (1)$$

where i = 1 or 2 represents the location identity of the data, $j = 1,\ldots,C_n^m$ represents the j^{th} fuzzy rule, $(x_1,\ldots,x_k)$ is the input vector with $k = 1,\ldots,n$, A_{ij_k} is the fuzzy set of the k^{th} partitioned rule spaces, and Y is the output from the trained neural network, *NN*.

Fuzzy Rule Interpolation

When the training of all the networks are terminated, the extracted fuzzy rules can be interpolated to the location of interest. In order to take into account the bad combinations of variables, the minimum overall errors on the validation set (i.e. where training is terminated) are used in the interpolation algorithm based on centered-defuzzification (Masters, 1993):

$$Y = \frac{\sum_i \sum_j \frac{NN_{ij}}{e_{ij} d_i}}{\sum_i \sum_j \frac{1}{e_{ij} d_i}} \quad (2)$$

where Y is the final estimate, d_i is the distance between the location of the i^{th} data set and the location of prediction, and e_{ij} is the minimum error on the validation set of j^{th} rule of the i^{th} data set.

The above formula is based on the notion that if the error is small, the uncertainty of the rule is less, and hence a high weighting is put on that rule. Similarly, if the separation distance between the location of the i^{th} data and the location of prediction is small, a higher weighting is put on that set of data.

EXAMPLE

Background

The proposed technique is applied to petroleum reservoir modelling. In reservoir modelling, we often drill only a few holes at different locations. Reservoir properties are measured at the laboratory from the rock samples retrieved at different depths. One of these properties is permeability which is a measure of fluid conductivity of the rock sample. Unlike well logs (a series of digital measurements at different depths) which are available in every hole, retrieving rock samples is an expensive process and hence permeability (or logarithm of permeability) has to be estimated from well logs alone.

Data Description

In this example, we had data from two oil wells, namely Well 1 and Well 2. They are separated by approximately 2.5 kilometres. The data was obtained from the North West Shelf, offshore Australia. Wells 1 and 2 recorded 173 and 225 data respectively at various depths, together with the corresponding permeability measurements. There are eight different well logs (m = 8) run in these wells. The rock types are identified by visual inspection of the samples. Nine classifications are obtained. In order to improve the performance of neural nets, each rock classification uses its own set of training and validation data.

Objective

The objective of this example is to predict the permeability values of another well, namely Well 3, using the model developed from wells 1 and 2 data. This well is location approximately half-way between wells 1 and 2. We will compare the predictions using the conventional single 8-dimensional (8D) network. Actual permeability values (157 of them) are available at this well for performance evaluation.

Network Setup

Since there are 9 rock classifications, 9 of the 8D networks were utilised. For the adaptive dimensional networks, C_n^m combinations of inputs with a total of 9 of each of the C_n^m networks were used. We first used the data from Well 1 for training, and the data from Well 2 for validation. Then we swapped the data as mentioned previously. All the input data was normalised in the range of 0 and 1, and the output data (logarithm of permeability) was in the range of 0.1 and 0.9 to avoid the asymptotic ends of the sigmoid transfer function. Different numbers of hidden neurons (*H*) were used.

Preliminary experiment

To determine the appropriate range of hidden neurons to use, we performed two experiments with 2-*H*-1 and 8-*H*-1 networks with a number of values of *H*, which gave the following results.

H	TSS on Well 3 data	
	9 runs x 2 inputs x 28 combinations of pairs	9 runs x 8 inputs x 1 combination of all 8
2	1.404	**1.407**
3	**1.395**	1.982
4	1.437	2.287
5	1.512	*3.349*

Table 1. Total sum of error squares (TSS).

The best result for 2-*H*-1 networks was for *H* = 3, while for 8-*H*-1 networks the best result was for *H* = 2. Thus, the range of value for *H* was chosen to be 2 to 5. In this range, the worst result was for the 8-5-1 network. As a final check to ensure that the TSS difference is significant in terms of the values we are trying to predict, the following diagram compares the two networks for predicting permeability.

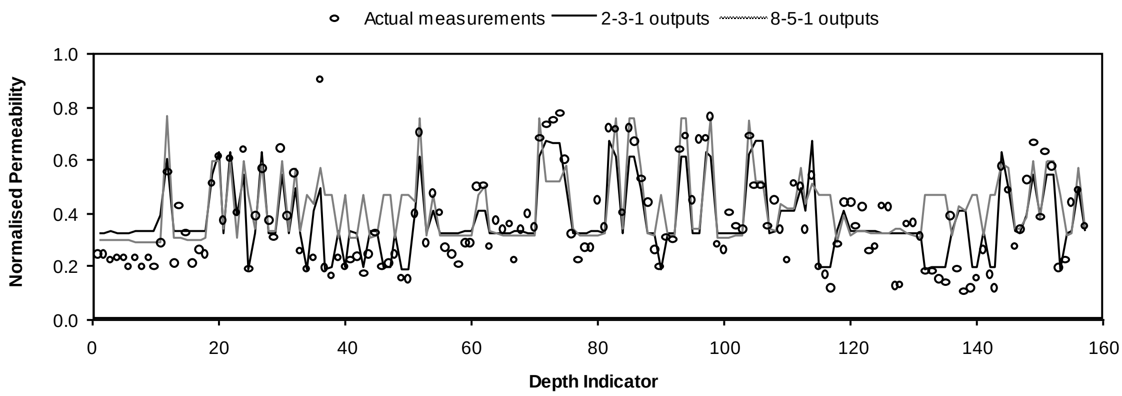

Figure 2. Core permeability versus 2 network predictions.

The prediction by the 8-5-1 network is clearly worse than the 2-3-1 network, hence the difference is sufficient. Thus, the rest of the results for the complete set of combinations of input values was done, as reported in the next section.

Results

Table 2 shows the averaged results for the rule interpolation using various *H* and *n* values. Clearly, this is straightforward, The 7-3-1 network is the best overall. The worst results are from the 1-*H*-1 networks, and include the worst best result with the 1-4-1 network. The column for 0 inputs is clearly fitting the best line to the outputs. That the TSS is so competitive is a measure of the complexity of the prediction task.

H	TSS on Well 3 data with different numbers of inputs								
	0	1	2	3	4	5	6	7	8
2	1.376	1.765	1.404	1.381	1.381	1.381	1.375	1.394	1.407
3	1.373	2.949	1.395	1.385	1.371	1.389	1.385	**1.364**	1.982
4	1.374	*1.506*	1.437	1.386	1.382	1.382	1.389	1.371	2.287
5	1.372	5.571	1.512	1.384	1.384	1.383	1.377	1.375	3.349

Table 2. Total sum of error squares (TSS) for different numbers of hidden neurons (*H*).

It is important to note that most TSS values obtained from the 8-*H*-1 networks were higher than the other networks. This is particulary significant when we consider that the best networks with 3, 4 or 5 hidden neurons are with 7 inputs. The most plausible explanation is that the 8th input interferes with the internal representation formed by the neural network and reduces its performance. In this case there must be many such interactions as all of the 7 input networks are improved.

CONCLUSIONS

We have presented a new technique to integrate spatial data. This technique uses multiple, adaptive dimensional neural nets, followed by interpolation of the derived fuzzy rules to the point-of-interest. The technique was applied to petroleum reservoir modelling where properties are estimated between two oil wells. Our technique uses separation distances in the interpolation routine which is an essential input in spatial data analysis. It gives us in general better results than standard multi-dimensional neural nets over a range of network sizes.

The greater number of smaller networks needed by our method use more connection weights overall, which would require some extra benefit over the standard techniques (in the worst case for our method, the result was no better than for the multi-dimensional network). This extra benefit comes in the ability to read off the relative significance of each input for each of the output rock classes, from the individual networks. This is a non-trivial task otherwise (Gedeon, 1996b), hence this is a significant benefit.

Future work will be required to investigate the relative weighting of bad combinations using the evidence combination notions from possibility theory, and to investigate the use of the relative significance information in pruning inputs from the multi-dimensional networks.

REFERENCES

Dayhoff, J.E. (1990), Neural Network Architectures: An Introduction, New York: Van Nostrand Reinhold.

Gedeon, T.D. (1996a), "Fuzzy-Neural Interpolation of Numeric Data," Proc. World Automation Congress (WAC'96), Montpellier, 237-242.

Gedeon, T.D. (1996b), “Indicators of Input Contributions: Comparing Functional Measures,” Proc. of IEEE Intl. Conference on Digital Signal Processing (TENCON'96), Perth, 520-525.

Gedeon, T.D., Ramer, A., Padet, C. and Padet, J. (1995), “Abstracting Uncertain Knowledge: Case for Neural Nets Applications,” Intl. J. of Intelligent Automation and Soft Computing, 1: 361-373.

Kóczy, L.T. and Hirota, K. (1990), “Fuzzy Inference by Compact Rules,” Proc. of Int. Conference on Fuzzy Logic & Neural Networks IIZUKA '90, Iizuka Fukuoka, 307-310.

Kóczy, L.T. and Hirota, K. (1993), “Approximate Reasoning by Linear Rule Interpolation and General Approximation,” Intl. J. of Approximate Reasoning, 9: 197-225.

Mamdani, E.H. and Assilian, S. (1975), “An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller,” Intl. J. of Man Machine Studies, 7: 1-13.

Masters, C. (1993), Practical Neural Network Recipes in C++, London: Academic Press, Inc.

Sugeno, M. and Takagi, T. (1985), “A New Approach to Design of Fuzzy Controller,” in Wang, P.P. ed., Advances in Fuzzy Sets, Possibility Theory, and Applications, New York: Plenum, 325-344.

Wong, P.M., Gedeon, T.D. and Taggart, I.J. (1995), “An Improved Technique in Porosity Prediction: A Neural Network Approach,” IEEE Trans. on Geoscience and Remote Sensing, 33: 971-980.

Zadeh, L.A. (1973), “Outline of a New Approach to the Analysis of Complex Systems,” IEEE Trans. on Sys. Man and Cybernetics, 1: pp. 28-44.