International Journal of Computer Applications (0975 – 8887)
Volume 63– No.5, February 2013

Core Porosity Estimation through Different Training Approaches for Neural Network: Back-Propagation Learning vs. Genetic Algorithm

Mojtaba Asoodeh
Islamic Azad University, Birjand Branch, Birjand, Iran

Parisa Bagheripour
Islamic Azad University, Birjand Branch, Birjand, Iran

ABSTRACT

Porosity of hydrocarbon bearing formations is a crucial parameter for reservoir characterization, reserve estimation, planning for completion, and geomechanical and geophysical studies. Accurate determination of porosity from laboratory core analysis is highly cost, time, and people intensive. Therefore, the quest for a rapid, cost-effective, and efficient method of determining porosity is inevitable. Conventional well log data are available in all wells and provide cheap continuous information. In this study, an improved strategy was followed to formulate conventional well log data (inputs) into core porosity (output) using the genetic optimized neural network (GONN). Firstly, back-propagation (BP) algorithm, the conventional learning method of neural network, was used to extract the formulation between inputs/output data space. Then, neural network was trained through the use of genetic algorithm (GA). Comparison between BP learning and GA demonstrated the effectiveness of GONN. It was deduced that GA enforces the performance function of neural network to converge to global minimum contrary to BP which frequently traps in local minima.

General Terms

Neural Network, Artificial Intelligence, Genetic Algorithm, Geosciences, Petroleum Engineering

Keywords

Back-Propagation, Genetic Algorithm, Genetic Optimized Neural Network, Core Porosity, Conventional Well Log Data

1. INTRODUCTION

Porosity refers to the void space portion of the rocks. Characterization of reservoir porosity is a very complex task due to its inherent heterogeneity [1]. However, its role in petroleum industry, especially in economic success of a reservoir development, makes the reservoir porosity of great attention. Core analysis is the oldest and still practiced technique for accurate measurement of porosity. However, it is highly cost, time, and people intensive. Therefore, this type of data is scars in hydrocarbon fields. On the contrary, conventional well log data are available in almost all wells. These logs contain invaluable implicit information about the hydrocarbon formations. Several researchers have tried to estimate reservoir properties from conventional well log data [2, 3]. In this study, neural network is employed to reveal what the conventional well log data are hiding, i.e. porosity. Associated weights and biases of neural network play a remarkable role in performance of the neural network. Back-propagation is the prevalent way of determining these parameters. This study proposes genetic optimized neural network in comparison with back-propagation neural network. Training efficiency of genetic algorithm (GA) is compared with back-propagation (BP). Results indicated the superiority of GA to BP in training the neural network. In this study, following three conventional well logs were chosen as inputs:

- RHOB log measures the bulk density of rocks. Records of RHOB are governed by lithology and porosity. As the porosity decreases, portion of solid grains increases and consequently the RHOB increases.
- Neutron porosity (NPHI) measures the hydrogen concentration in the formation. Since fluids filled the pore space of the rocks are the sole sources of hydrogen occurrence, NPHI has a relationship with core porosity.
- Sonic transit time (DT) is reciprocal of compressional wave velocity and is dominated by porosity, i.e. DT increases as porosity increases.

2. NEURAL NETWORK TRAINING: BACK-PROPAGATION LEARNING VS. GENETIC ALGORITHM

Neural network is a computational tool which emulates the biological cognition of human's brain for both classification and regression purposes. Assuming a three-layered neural network with *n* input layer nodes, *m* hidden layer nodes, and one output layer node; neural network utilizes the following mathematics for formulating inputs to the output.

$$IW_{m\times n} \times I_{n\times 1} + Ib_{m\times 1} = netHL_{m\times 1} \quad (1)$$

$$OHL_{m\times 1} = f_{HL}(netHL_{m\times 1}) \quad (2)$$

$$OW_{1\times m} \times OHL_{m\times 1} + b_{1\times 1} = netOL_{1\times 1} \quad (3)$$

$$ONN = f_{OL}(netOL_{1\times 1}) \quad (4)$$

Where, *IW, I, OW, Ib,* and *b* refer to initial weights, input vector, output layer weights, initial biases, and output layer bias, respectively. *netHL* and *netOL* are net input vectors of hidden layer and output layer, correspondingly. OHL and ONN are in turn output vectors of hidden layer and neural network's output. f_{HL} and f_{OL} refer to transfer functions of hidden layer and output layer, respectively. These functions are defined as:

$$f_{HL}(net) = \frac{1}{1+\exp\ (-net)} \quad (5)$$

$$f_{OL}(net) = net \quad (6)$$

In above equations, all weights and biases are unknown. During the training process, neural network would be capable of learning from training data and adjusting its weights and biases. This process enables the neural network to extract a satisfying formulation between inputs and output. Back-propagation algorithm is a popular method which is routinely used for training of neural network. The simplest implementation of back-propagation learning updates weights and biases in the direction in which the performance function (half of mean square error (MSE) of prediction) decreases most rapidly [4]. The following mathematics justify that performance function decreases most rapidly in direction of the negative of gradient. Supposing t_j is the target value of jth node, o_j is the real value jth node, and w_{ij} is connection weight and bias from ith node to jth node; performance function is defined as following equation.

$$E = \frac{1}{2}\sum_j (t_j - o_j)^2 \quad (7)$$

Net input of jth node is equal to:

$$net_j = \sum_i w_{ij} o_i \quad (8)$$

Assuming f_j is transfer function of jth node, the real output of jth node is defined as:

$$o_j = f_j(net_j) \quad (9)$$

Use of the chain rule for derivative results in the following equation:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial net_j}\frac{\partial net_j}{\partial w_{ij}} \quad (10)$$

By substituting equation 8 into the second term of equation 10, the following equation is deduced:

$$\frac{\partial net_j}{\partial w_{ij}} = \frac{\partial}{\partial w_{jj}}\sum_k w_{kj} o_k = \sum_u \frac{\partial w_{ik}}{\partial w_{ij}} o_k = o_i \quad (11)$$

Since $\frac{\partial w_{ik}}{\partial w_{ij}}$ is always equal to zero except for i=k.

$-\frac{\partial E}{\partial net_j}$ is defined as gradient of performance function (δ_j). Thus:

$$-\frac{\partial E}{\partial w_{ij}} = \delta_j o_i \quad (12)$$

This equation means that modification of weights and biases in direction proportional to negative of gradient leads to decreasing of performance function. Generally, modification of weights and biases follows the succeeding equation:

$$w(t+1) = w(t) - \alpha_t g_t \quad (13)$$

Where, w (t+1) and w (t) refer to weights and biases of (t+1) th and tth iterations, respectively. α_t and g_t are learning rate and gradient in tth iteration, correspondingly. Up to now, several methodologies have been presented by some researchers for implementing equation 13. They tried to propose fast and accurate ways of assigning and calculating learning rate and gradient. These attempts lead to several training functions, including Bayesian regularization (BR), Levenberg-Marquardt (LM), BFGS Quasi-Newton (BFG), Resilient Backpropagation (RP), Conjugate Gradient with Powell/Beale Restarts (CGB), Scaled Conjugate Gradient (SCG), Fletcher-Powell Conjugate Gradient (CGF), Polak-Ribiére Conjugate Gradient (CGP), One Step secant (OSS), and Variable Learning Rate Backpropagation (GDX). More details about these training functions can be found in a number of papers and reviews [5-13].

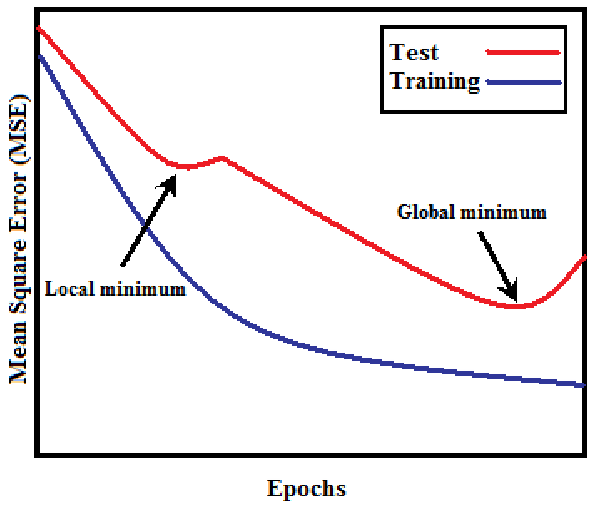

Figure 1: Graph showing the difference between local minimum and global minimum.

According to equation 13, in local minima where gradient approaches to zero weights and biases tend to remain unchanged. Based on this equation, back-propagation algorithm cannot distinguish between local minima and global minimum (difference between local minimum and global minimum is illustrated in Figure 1). Therefore, it’s a common trouble for neural networks to be trapped in local minima. Consequently, the extracted formulation between inputs and output is not the optimal one. It means all constructed neural network associated with uncertainty. To eliminate this flaw, it is necessary to find another way of assigning connection weights and biases for neural network which is capable of escaping from local minima.

Table.1. Statistics of Datasets used in this study.

	Min	Max	Mean	sigma	R^2 with output
NPHI	0.0046	0.3140	0.1138	0.0642	0.5511
RHOB	2.1071	2.8280	2.6084	0.1155	0.4054
DT	40.7304	88.2162	60.5129	7.9745	0.3597
Porosity	0.007	0.333	0.1028	0.0534	1

Genetic algorithm is a method for solving widespread optimization problems. Genetic algorithm produces the global minimum of introduced fitness function (function that its global minimum is sought). Therefore, a function meant to be solved should be rearranged such that the global minimum of rearranged function and sought point of original function are coinciding. Genetic algorithm follows an interesting strategy for finding the global minimum. It randomly gathers a population of probable solutions (chromosomes) and asses each based on fitness function. The assessment assigns a score to each chromosome which is used for their ranking. Top-ranked chromosomes are then selected and genetic operations are applied to them for generating chromosomes of next population. Genetic operators include cross-over, mutation, inversion and elite preservation. All chromosomes are codded into binary strings. If all zeros converted to ones and vice versa, it is called inversion operator. If two chromosomes randomly break from one or more parts and switch their brocken parts, they will produce two new chromosomes. This procedure is called cross-over. Mutation refers to stochastic change of one or more digits of chromosomes. Elites are those chromosomes with the lowest value of fitness function (or the highest score). The aforementioned genetic operators provide a stochastic search capability for genetic algorithm. Mentioned process is repeated in successive iterations evolving toward optimal chromosome. Selection of initial population and generation of succeeding populations all exploit a stochastic nature. In other words, final result of genetic algorithm is independent of the points that genetic algorithm starts from. In order to train neural network with genetic algorithm, the performance function of neural network (equation 7) should be introduced into genetic algorithm. It can be stated in following form, too.

$$MSE = \frac{1}{N}\sum_{\ell=1}^{N}(ONN_\ell - t_\ell)^2 \quad (14)$$

Genetic algorithm would be able to extract all associated weights and biases for neural network through the stochastic optimization of equation 14. By use of genetic algorithm instead of back-propagation algorithm, risk of sticking in local minima will be eliminated.

3. RESULTS & DISCUSSION

At the first stage of this study, quality of datasets was checked and bad hole intervals were removed by processing of conventional well log data. Introducing noisy data make confused the neural network. Therefore, it was necessary to remove noisy data by removing bad hole intervals. Table 1 indicates the statistics of inputs/output datasets used in this study. The underlying dependency between inputs and output is mention in Table 1 using the concept of coefficient of determination (R-Square). In next step, a three-layered back-propagation neural network was constructed for estimating porosity from conventional well log data. Framework of neural network was composed of three input nodes, five hidden nodes, and one output nodes. TANSIG and PURELIN transfer functions were used for hidden layer and output layer, respectively. TANSIG transfer function is mathematically equivalent to hyperbolic tangent function, and PURELIN transfer function is equivalent to function “f(x) =x”. The constructed neural network was trained using the Levenberg-Marquardt training function (one of the popular implementation of back-propagation algorithm).

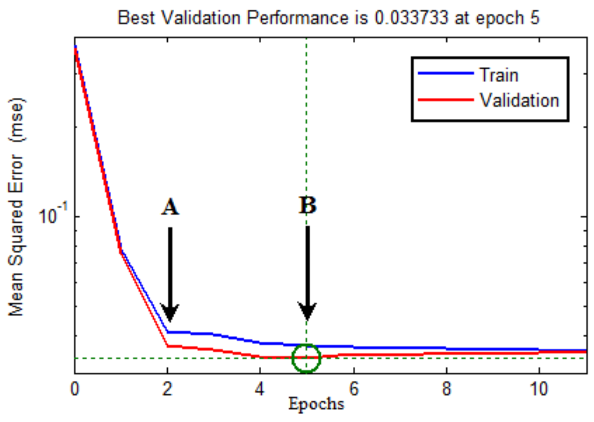

Figure 2: Graph showing mean square of error for training and test data versus epochs. Back-propagation algorithm defines points A and B as local minimum and global minimum, respectively.

Figure 2 illustrates the mean square error of training data and test data during the learning procedure of neural network. It shows the mean square error (MSE) of constructed model is equal to 0.03373 for normalized data. The learning process ceased in point B and back-propagation algorithm defined the point B as global minimum. This figure indicates that BP could escape from the point A, which is a local minimum. After training the neural network using the back-propagation algorithm, performance of the constructed model was evaluated by introducing test data. Figure 3 shows the quality of estimation using the concept of correlation coefficient. This figure indicates the neural network with back-propagation algorithm was capable of satisfyingly estimate core porosity from conventional well log data. The deviation between fitted line and ideal line in Figure 3 is related to high-valued core porosity. In other words, lack of sufficient representative high porosity data cause this error.

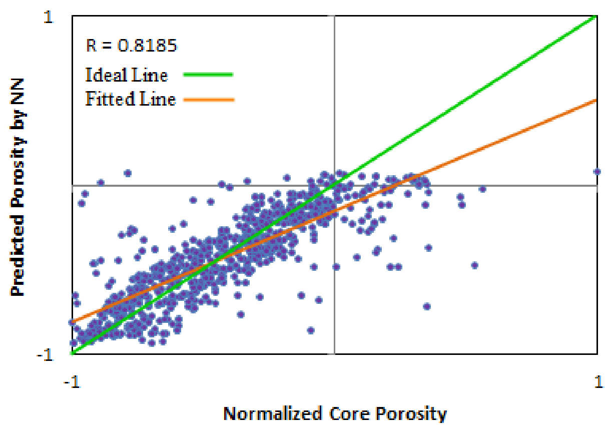

Figure 3: Crossplot showing the correlation coefficient between core porosity and BPNN predicted values in normalized scale.

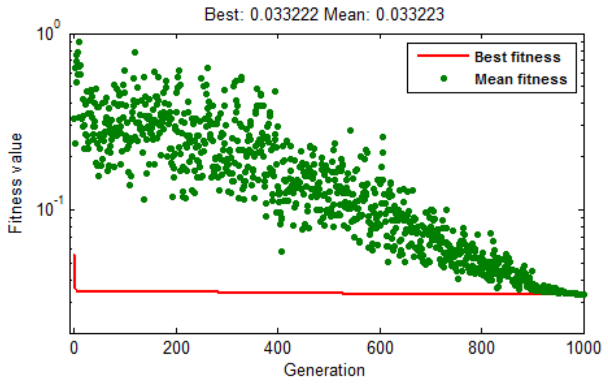

Figure 4: Best and mean values of fitness function for GONN model during the 1000 generations.

In the latter stage, an unfilled skull of neural network (equation 14) was introduced to genetic algorithm. Genetic algorithm was capable of filling this skull through its stochastic search capability, i.e. all weights and biases of neural network were extracted by virtue of genetic algorithm. This strategy is called genetic optimized neural network (GONN). To construct the GONN, equation 14 was defined as fitness function. A population of twenty random chromosomes was employed to initiate the evolutionary process through the use of genetic operators, including cross-over, mutation, inversion and elite preservation. Figure 4 illustrates the best and mean fitness function scores for each population of chromosomes during the 1000 generations of genetic algorithm. It indicates the mean square error of GONN for normalized data is equal to 0.033222. Figure 5 indicates the correlation coefficient between measured core porosity and GONN predicted values. It is observed that GONN significantly improved the accuracy of final prediction and fitted line approached to ideal line. Figure 6 illustrates the error (Core Porosity-GONN Predicted Porosity) of prediction in normalized scale versus sample. It indicates most of the data points are in good agreement with reality. Error for sixty eight percent of samples located in range of mu±sigma, where mu and sigma are mean and standard deviation of error distribution of GONN.

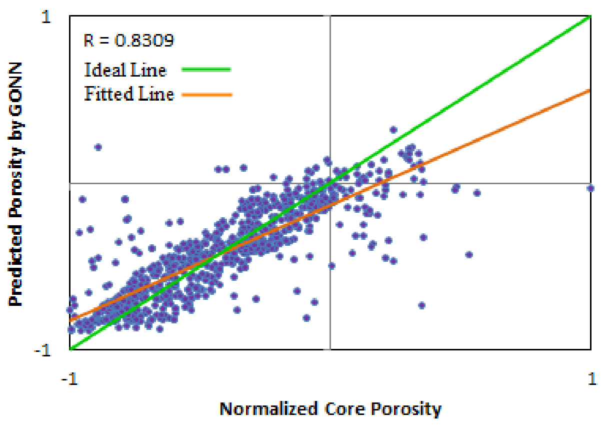

Figure 5: Crossplot showing the correlation coefficient between core porosity and GONN predicted values in normalized scale.

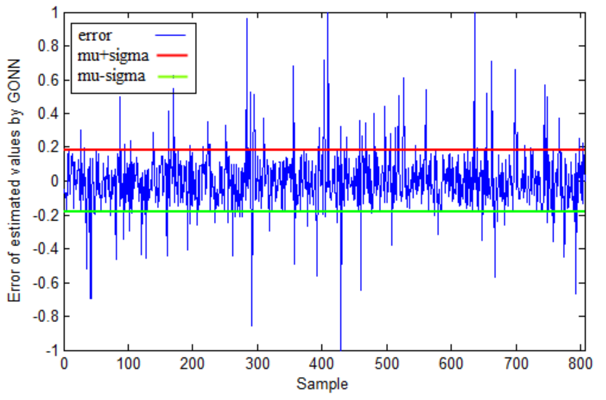

Figure 6: Graph showing error of estimation versus sample. Error of 68% of samples locates in range of mu±sigma.

4. CONCLUSIONS

This study followed a sophisticated approach to construct a quantitative formulation between conventional well log data and core porosity. Results indicated that back-propagation neural network (BPNN) is capable of producing precise estimation of porosity from conventional well log data. However, use of genetic algorithm for optimizing the neural network, which is called genetic optimized neural network (GONN), can significantly enhance the accuracy of final prediction. Correlation coefficient and MSE are two witness criteria for this claim. MSE for BPNN and GONN was equal to 0.03373 and 0.03322, respectively. Furthermore, the correlation coefficient for BPNN and GONN models was equal to 0.8185 and 0.8309, correspondingly. Use of GONN instead of BPNN undoubtedly eliminates the risk of sticking in local minima. Accuracies of both models were shrinkaged in high porosity values, which are attributed to lack of sufficient representative data for high porosities. This confirms the need for a comprehensive datasets in modeling by neural networks. Eventually, it is deduced obviously that implementation of the proposed strategy can considerably reduce cost of reservoir characterization and saves time.

5. REFERENCES

[1] Al-Qahtani, F. A. 2000. Porosity Distribution Prediction Using Artificial Neural Networks. College of Engineering and Mining Recourses, West Virginia University, M.Sc. Thesis, 1-3.

[2] Asoodeh, M., Bagheripour, P. 2012. Prediction of Compressional, Shear, and Stoneley Wave Velocities from Conventional Well Log Data Using a Committee Machine with Intelligent Systems. Journal of Rock Mechanics and Rock Engineering 45, 45-63.

[3] Mohaghegh, S. D. 2000. Virtual-intelligence applications in petroleum engineering: part 1-Artificial Neural Network. Distinguished Author Series articles, SPE 58046.

[4] MATLAB user's guide. 2011. Fuzzy logic, Neural Network & GA and Direct Search Toolboxes, MATLAB CD-ROM, by the Mathworks, Inc.

[5] Asoodeh, M., Bagheripour, P. 2012. Estimation of Bubble Point Pressure from PVT Data Using a Power-Law Committee with Intelligent Systems. Journal of Petroleum Science and Engineering 90-91, 1-11.

[6] Kononen, V. 2005. Gradient descent for symmetric and asymmetric multiagent reinforcement learning. Web Intell. Agent Syst. 3, 17–30.

[7] Burney, S. M. A., Jilani, T. A., Ardil, C. 2004. Levenberg–Marquardt Algorithm for Karachi Sock Exchange Share Rates Forecasting. Trans. Eng. Comput. Technol. 3, 1305–1313.

[8] Demuth, H., Beale, M. 2002. Neural Network Toolbox, User's Guide (Version 4). The Mathworks Inc.

[9] Baird, L., Moore, A. 1999. Gradient Descent for General Reinforcement Learning. Adv. Neural Inf. Process Syst. 11, 968–974.

[10] Bishop, C. M. 1995. Neural Networks for Pattern Recognition. Clarendon Press, Oxford, p. 670.

[11] Riedmiller, M., Braun, H. 1993. A direct adaptive method for faster back-propagation learning: the RPROP algorithm. Proc. IEEE Conf. on Neural Networks.

[12] MacKay, D. J. C. 1992. A practical Bayesian framework for back-propagation networks. Neural Comput 4, 448–472.

[13] Battiti, R. 1992. First and second order methods for learning: between steepest descent and Newton's method. Neural Comput 4 (2), 141–166.