HONEST: A New High Order Feedforward Neural Network

Ashraf Abdelbar and Gene Tagliarini
Dept. of Computer Science, Clemson University,
Clemson, SC 29634-1906.
Tel: (864)656-5856, Fax: (864)656-0145,
email: abdelbar@cs.clemson.edu

ABSTRACT

A frequently voiced complaint regarding neural networks is that it is difficult to interpret the results of training in a meaningful way. The HONEST network is a new feedforward High Order Neural Network (HONN) which not only allows a fuller degree of adaptability in the form of the nonlinear mapping than the sigma-pi model, but has a structure that can make it easier to understand how the network inputs come to be mapped into the network outputs. This structure also makes it easier to use external expert knowledge of the domain to examine the validity of the HONEST network solution and possible to reject some solutions. This is particularly important for embedded, failure-critical systems such as life-support systems. We have applied the HONEST network to the problem of forecasting the onset of diabetes using eight physiological measurements and genetic factors. We obtained a successful classification rate of 83% compared to a 76% rate that had been obtained by previous researchers.

1. Introduction

A neuron in the Multi-Layer Perceptron (MLP) model combines its inputs according to

$$net_i = \sum_j w_{ij} o_j + \theta_i \ , \tag{1}$$

then computes its output according to

$$o_i = f(net_i) \ , \tag{2}$$

where f is some nonlinear, frequently sigmoidal, activation function. Zurada [16] classifies such networks as linear because they combine their inputs linearly. According to his classification, a High Order Neural Network (HONN) is one in which the neurons can combine their inputs nonlinearly.

While HONN's may be more expensive computationally, they have certain advantages over linear networks. According to DeClaris and Su [5], because high-order neurons combine their inputs in a nonlinear manner, they are better able to capture high-order correlations and form nonlinear mappings that would require a large number of hidden neurons in a linear network. In addition, HONN's may also have better generalization properties [12].

2. The HONEST Model

The High Order Network with Exponential SyanpTic links (HONEST) model [1] can be thought of as an extension of the sigma-pi model [14]. An HONEST network, as shown in Figure 1, is a three-layer feedforward network made up of two different neuron types. Neurons in the hidden layer are all high-order whereas output layer neurons are simple linear units.

The functionality of a hidden layer neuron i, as illustrated in Figure 2, is described by

$$o_i = \prod_j o_j^{p_{ij}} \ , \tag{3}$$

where p_{ij} is an exponential power associated with the synapse connecting unit input j to hidden neuron i.

The connections from the input layer to the hidden layer do not have an associated multiplicative weight; instead they have an associated adaptable exponential power. The output of a hidden layer neuron is the product of its inputs after each is raised to its associated power. The activation function for all neurons in an HONEST network is the identity function $f(x) = x$.

The output layer neurons are all simple linear units. The functionality of an output layer neuron i is described by

$$y_i = \sum_j w_{ij} o_j + \theta_i \, . \tag{4}$$

Each of the outputs, y_i, of an HONEST network can be expressed in terms of the network inputs by an expression of the form

$$y_i = \sum_{h=1}^{HIDDEN} w_{hi} \prod_{j=1}^{INPUTS} x_j^{p_{hj}} + \theta_i \, . \tag{5}$$

For example, the equation

$$y = 3.2 x_1^{2.5} x_2^{-1.3} - 0.5 x_1^2 x_2^{-3} + 2 x_1^4 - 4.3 \, , \tag{6}$$

is represented by the network shown in Figure 3.

The form of equation (5) is similar to the form of a polynomial with the number of hidden neurons corresponding to the number of terms in the polynomial. In fact, the only difference is that in a polynomial, the exponents are restricted to being non-negative integers. Therefore, the form of equation (5) can be seen as a generalization of the form of a polynomial. Because any continuous, real-valued function, on a closed and bounded interval, can be approximated to any desired degree of accuracy by some polynomial, it follows that a three-layer HONEST network with a sufficient number of hidden neurons is a universal approximator.

3. Advantages of the HONEST Network Structure

A frequently voiced complaint about neural networks is that once the training of a network is complete, it is very difficult to interpret the weights of the network in a meaningful way. In the case of a three-layer MLP network, the output of the network is expressed as a sigmoid function of a linear sum of sigmoid functions of linear sums of the network inputs. For non-trivial network sizes, it is practically impossible to get an intuitive understanding of how the network's inputs are mapped to the network's outputs. Thus, in applications where the network is to be embedded in a failure-critical system, such as a medical life-support system, it is difficult to guarantee the behavior of the network.

The structure of an HONEST network is more transparent and easier to interpret; each of the outputs of an HONEST network can be expressed in terms of the network inputs by a polynomial-like equation. It is thus easier to understand how the network's inputs come to be mapped to the network's outputs. For example, using the form of equation (5), it is possible to hold the values of the rest of the network inputs constant, and analytically investigate how two network inputs vary with each other for a certain value of the output. Further, it is possible to use external expert knowledge of the problem domain to examine the validity of the network solution. Alternatively, examining the HONEST network solution may serve to enrich one's understanding of the problem domain.

Another advantage of having the state of an HONEST network represented by an equation of the form of (5) arises from the fact that in some problem domains, a non-neural network solution may already exist. When an existing solution is in the form of a polynomial mapping, it is possible to use the polynomial to initialize the network. Since gradient-descent methods are generally prone to local minima traps, using an existing "good" solution as a starting point can be beneficial.

For example, we found this useful in our application of HONEST to a game-playing program (described in detail in [2]). The Othello program BILL [10, 11] won the 1989 North American Computer Othello Championship by using Bayesian learning [6], a statistical learning method, in its evaluation function. In BILL, Bayesian learning was used to learn from a training set, the "best" quadratic combination of feature evaluations. Our objective was to use HONEST to improve on the results of Bayesian learning.

Rather than commence training from a random starting point, we were able to start training with the HONEST network initialized to the mapping learned by Bayesian learning. Since the mapping produced by Bayesian learning was a quadratic polynomial, it was a simple matter to prescribe an HONEST network to represent it. This allowed us to use the existing good Bayesian learning solution as a launch-point for training and made it easier for us to find a better feature combination. Further, because of HONEST's transparent structure, we were able to examine the HONEST solution and make observations that could be useful to an Othello program designer even if he were not using machine learning.

Our use of existing knowledge to improve the learning process in the Othello problem has some similarily to learning with hints [3, 4]. In that method, prior knowledge (hints) about a problem domain is used to create additional training examples which are presented to the network throughout the learning process. However, since our approach only uses an existing solution to find a starting point for training, our approach does not rely on the prior knowledge being accurate. In a sense, in the worst case, starting training with "bad weights" is not likely to be worse than with random weights. Furthermore, since learning with hints is compatible with any descent technique [3], the potential exists for applying learning with hints to the HONEST network to attain the benefits of both approaches.

4. Learning Equations for HONEST

Both the weights in the hidden layer to output layer connections and the exponents in the hidden layer to output layer connections are learned through a generalization of the backward error propagation algorithm. Because the output layer units are simple linear neurons with an identity activation function, their learning equations are the same as for the MLP model:

$$\delta_i = -(y_i - o_i) \ , \quad \Delta w_{ij} = \eta \delta_i o_j \ , \quad \delta_j = \sum_i \delta_i w_{ij} \ . \tag{7}$$

where δ_i and δ_j are the error signals for an output layer neuron i and hidden layer neuron j, respectively, and Δw_{ij} is the requisite change in the weight w_{ij}.

Now, it remains to derive Δp_{ij} for the exponent p_{ij} associated with the connection from input layer unit j to hidden layer neuron i.

$$\frac{\partial o_i}{\partial p_{ij}} = \frac{\partial}{\partial p_{ij}} \left(o_j^{p_{ij}} \prod_{k \neq j} o_k^{p_{ik}} \right) = \left(\prod_{k \neq j} o_k^{p_{ij}} \right) \frac{\partial o_j^{p_{ij}}}{\partial p_{ij}} = o_j^{p_{ij}} \ln(o_j) \prod_{k \neq j} o_k^{p_{ik}} = o_i \ln(o_j) \ . \tag{8}$$

Therefore,

$$\Delta p_{ij} = -\eta \frac{\partial E}{\partial p_{ij}} = \eta \delta_i o_i \ln(o_j) \ . \tag{9}$$

As is clear from equation (9), we need to take the log of the network inputs as part of the backward error propagation process. However, we do not need to take the log of hidden neuron outputs. Therefore, this potential limitation can be overcome by scaling the network inputs to be positive.

5. Comparison to Other High Order Neural Networks

5.1. The Sigma-Pi Model

The sigma-pi model was first developed by Rumelhart *et al.* [14] and has been studied by other researchers including DeClaris and Su [5] and Kavuri and Venkatsubramanian [9]. The inputs to a sigma-pi neuron are grouped into sets called conjuncts. The output of the neuron is the result of applying an activation function f to the weighted sum of the products of the members of each conjunct. In other words, the inputs to a neuron i are combined according to

$$net_i = \sum_{j=1}^{N_i} w_{ij} \prod_{k \in A_j} x_k \ , \tag{10}$$

where N_i is the number of conjuncts associated with neuron i, and A_j represents conjunct set number j.

The weights w_{ij} are learned through backward error propagation but the number of conjunct sets for each neuron and the membership of each set are fixed at network configuration time and are not adaptable. Narayan [14] observes that this predetermination may be appropriate for applications where *a priori* knowledge of the problem being modeled is available, but in general this lack of adaptability is a significant limitation.

The HONEST model on the other hand does not require assumptions about the form of the mapping. By allowing the exponents to be trained, the HONEST model allows full adaptation of the form of the nonlinear mapping.

5.2. The ExpoNet Model

ExpoNet is a HONN which also allows for adaptation of exponents. In ExpoNet, every synapse has an associated weight w_{ij} and associated exponent p_{ij}. Each neuron combines its inputs through a weighted summation of its inputs after each input raised to its associated exponent and then adds its self-bias. In other words,

$$net_i = \sum_j w_{ij} o_j^{p_{ij}} + \theta_i \, . \tag{11}$$

Both the weights and exponents are learned through backward error propagation; however, ExpoNet cannot directly represent an arbitrary polynomial. For example, it is not possible to prescribe an ExpoNet network to represent the equation represented by the HONEST network in Figure 3. In the Othello application described earlier, we were able to use the quadratic mapping produced by Bayesian learning to prescribe an HONEST network. This would not be possible with ExpoNet because the cross-terms, *e.g.* $x_1 x_2$, cannot be directly represented in an ExpoNet network. It would be possible with the sigma-pi model to prescribe a network representing a quadratic mapping but it would not be possible to adapt the exponents.

5.3. The GMDH Model

The General Method of Data Handling (GMDH) is an early polynomial neural network developed by Ivaknenko [8] and studied by others [7]. Adaptation in this HONN is not based on backward error propagation; instead, it learns using a method based on linear regression.

Like HONEST, the state of a GMDH network can always be represented by a polynomial equation. Further, GMDH allows full adaptation of the form of the polynomial and does not require *a priori* knowledge. However, unlike HONEST, the mapping learned by GMDH is always strictly a polynomial. It does not allow negative or even non-integer exponents. However, since GMDH learning is not based on gradient-descent, it is possible to use it as pre-processing for an HONEST network. In other words, first a polynomial mapping can be learned by a GMDH network as a good starting approximation. This polynomial mapping can then be used as a launch-point for training an HONEST network.

6. Applying HONEST to Diabetes Forecasting

The HONEST model was applied to the problem of forecasting the onset of diabetes (within a five year period) in a population of Pima Indians living on a reservation near Phoenix, Arizona [1]. Because of their high incidence rate of diabetes, this population has been under continuous study since 1965 by the National Institute of Diabetes and Digestive and Kidney Diseases. We used the same dataset (consisting of 576 training cases and 192 test cases) that had been used in a previous study by Smith *et al.* [15]. Smith *et al.* used a neural network algorithm called ADAP and the following eight features for their forecasting: number of previous pregnancies, plasma glucose concentration after 2 hours in a Glucose Tolerance Test (GTT), diastolic blood pressure, triceps skin fold thickness, 2-hour serum insulin, body mass index (weight divided by the square of the height), a diabetes pedigree function which evaluates the subject's genetic family history of diabetes, and age.

With the ADAP neural network, which contained more than 100,000 elements, Smith *et al.* obtained a successful classification rate of 76%. With an HONEST network started with random weights, and the same eight features, we were able to obtain a successful classification rate of 83%. Our network consisted of 8 input units, 40 hidden neurons, and a single output neuron. This is a total of 49 units compared to the more than 100,000 units used in the ADAP network.

Training the HONEST network was not entirely simple. The network found local minima three times and it was necessary to re-start the learning process. More than 100,000 training epochs were required, which really is not a very large number of training epochs considering the size of the training set. The learning rate for the multiplicative weights in the hidden layer to output layer connections was 0.01, and, because of the high level of sensitivity of the exponential weights, the learning rate for the input layer to hidden layer connections was 0.0000001.

Although achieving a correct classification rate of 83% required a significant amount of training, slightly lower classification rates were surprisingly easy to achieve. Out of 10 HONEST networks started with random weights and allowed to run for only 1000 epochs, five networks failed to achieve even 70% correct classification. However, the remaining five networks were able to achieve successful classification rates higher than 81% and one of the five achieved a rate of 82% in only 30 epochs.

References

[1] Abdelbar, A., "HONEST: A New High Order Neural Network Architecture," Master's Thesis, Dept. of Computer Science, Clemson University, 1994.

[2] Abdelbar, A., "Teaching the HONEST Neural Network to Judge the Strength of Othello Positions," Technical Report 96-100, Department of Computer Science, Clemson University, 1996.

[3] Abu-Mostafa, Y.S., "An Algorithm for Learning with Hints," *Proceedings of 1993 International Conference on Neural Networks*, Vol. 2, pp. 1653-1656.

[4] Abu-Mostafa, Y.S., "Machines that Learn from Hints," *Scientific American,* Vol. 272, No. 4, pp. 68-73, April 1995.

[5] DeClaris, N., and Su, M., "A novel class of neural networks with quadratic junctions," *Proceedings of IEEE Conference on Systems Man and Cybernetics*, Charlottesville, Virginia, pp. 1557-62, 1991.

[6] Duda, R., and Hart, P., *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.

[7] Farlow, S., *Self-Organizing Methods in Modeling: GMDH Type Algorithm,* Dekker, New York, 1984.

[8] Ivakhnenko, A.G., "Polynomial theory of complex systems," *IEEE Transactions on Systems, Man and Cybernetics,* **12** (1971) pp. 364-378.

[9] Kavuri, S.N., and Venkatsubramanian, V., "Solving the hidden node problem in networks with ellipsoidal units and related issues," *Proceedings of the International Joint Conference on Neural Networks*, **III**, pp. 689-692, Baltimore, 1992.

[10] Lee, K.-F., and Mahajan, S.,"A Pattern Classification Approach to Evaluation Function Learning,"*Artificial Intelligence* **36** (1988) 1-25.

[11] Lee, K.-F., and Mahajan, S.,"The Development of a World Class Othello Program," *Artificial Intelligence* **43** (1990) 21-36.

[12] Maxwell, T., Giles, C.L., and Lee, Y.C., "Generalization in Neural Networks: The Contiguity Problem," *Proceedings of the IEEE International Conference on Neural Networks*, **II**, pp. 41-46, San Diego, 1987.

[13] Narayan, Sridhar, "ExpoNet: A Generalization of the Multi-layer Perceptron Model," *Proceedings of the World Congress on Neural Networks*, **III**, pp. 494-497, Portland, Oregon, July 1993.

[14] Rumelhart, D.E., Hinton, G.E., and McClelland, J.L., "A General Framework for Parallel Distributed Processing," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, eds. D.E. Rumelhart, J.L. McClelland, and the PDP Research Group, Bradford, Cambridge, 1986.

[15] Smith, J., Everhart, J.E., Dickson, W.C., Knowler, W.C., Johannes, R.S., "Using the ADAP Learning Algorithm to Forecast the Onset of Diabetes Mellitus," *Proceedings of the Twelfth Annual IEEE Symposium on Computer Applications in Medical Care*, November 1988.

[16] Zurada, Jacek, *Introduction to Artificial Neural Systems*, West Publishing Company, St. Paul, 1992.

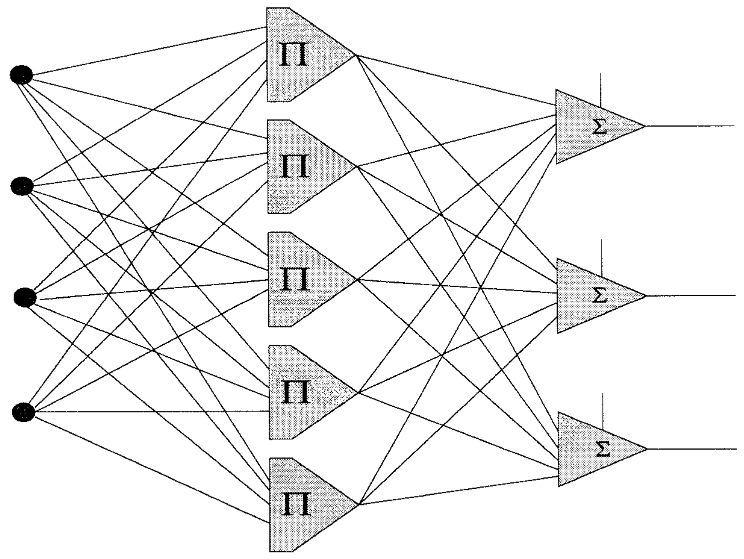

Fig. 1: The HONEST Network

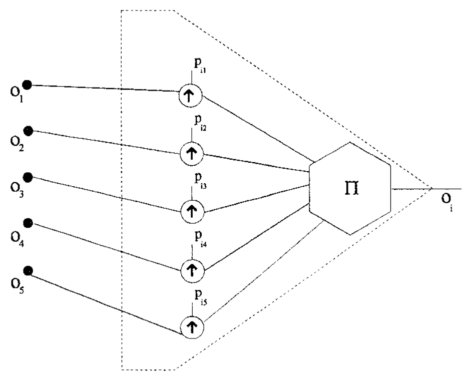

Fig. 2: A Hidden Layer Neuron in the HONEST Architecture

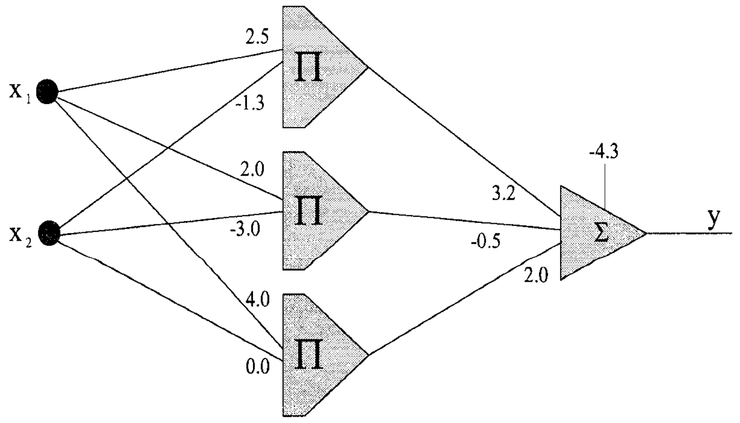

Fig. 3: Example of an HONEST Network