PRACE NAUKOWE AKADEMII EKONOMICZNEJ WE WROCŁAWIU
Nr 1121 — 2006
Pozyskiwanie wiedzy i zarządzanie wiedzą

Yevgeniy Bodyanskiy, Oleksandr Slipchenko

Kharkiv National University of Radioelectronics

ONTOGENIC NEURAL NETWORKS USING ORTHOGONAL ACTIVATION FUNCTIONS

1. Introduction

Artificial neural networks (ANNs) are widely applied to solving a variety of problems such as information processing, data analysis, system identification, control etc. under structural and parametric uncertainty [*Handbook*... 1997; Nelles 2001].

One of the most attractive properties of ANNs is the possibility to adapt their behavior to the changing characteristics of the modeled system. By adaptivity we understand not only the adjustment of parameters (synaptic weights), but also the possibility to adjust the architecture (the number of nodes). The goal of the present paper is the development of an algorithm for structural and synaptic adaptation of ANNs for nonlinear system modeling, capable of online operation, i.e. sequential information processing without re-training after structure modification.

The problem of optimization of neural network architecture has been studied for quite a long time. The algorithms that start their operation with simple architecture and gradually add new nodes during learning, are called 'constructive algorithms'. In contrast, destructive algorithms start their operation with an initially redundant network, and simplify it as learning proceeds. This process is called 'pruning'.

Radial basis function network (RBFN) is one of the most popular neural network architectures [Poggio, Girosi 1989]. One of the first constructive algorithms for such networks was proposed by Platt and named 'resource allocation' [Platt 1991]. By present time, a number of modifications of this procedure is known [Nag, Ghosh 1998; Yingwei et al. 1998]. One of the most known is the cascade-correlation architecture developed by Fahlman and Lebiere [Fahlman, Lebiere 1990].

Among the destructive algorithms, the most popular are the 'optimal brain damage' [Cun et al. 1990] and 'optimal brain surgeon' [Hassibi, Stork 1993]. In these methods, the significance of a node or a connection between nodes is determined by

the change in error function that its deletion incurs. For this purpose, the matrix of second derivatives of the optimized function with respect to the tunable parameters is analyzed. Both procedures are quite complex computationally. Besides that, an essential disadvantage is the need for re-training after the deletion of non-significant nodes. This, in turn, makes the real-time operation of these algorithms impossible. Other algorithms such as [Prechelt 1997] are heuristic and lack universality.

It should be noted that there is no universal and convenient algorithm, which could be used for the manipulation of the number of nodes and suitable for most problems and architectures. Many of the algorithms proposed so far lack theoretical justification as well as the predictability of the results of their application and the ability to operate in real time.

2. Network Architecture

Let's consider the network architecture, that implements the following nonlinear mapping

$$\hat{y}(k) = \hat{f}\left(x(k)\right) = \sum_{i=1}^{n} \sum_{j=1}^{h_i} w_{ji} \phi_{ji}\left(x_i(k)\right), \tag{1}$$

where: $k = 1, 2, \ldots$ – discrete time or ordinal number of sample in training set,

w_{ji} – tunable synaptic weights,

$\phi_{ji}(\cdot)$ – j-th activation function for i-th input variable,

h_i – number of activation functions for appropriate input variable,

$x_i(k)$ – value of i-th input signal at time moment k (or for k-th training sample).

Proposed architecture contains $h = \sum_{i=1}^{n} h_i$ tunable parameters and it can be readily seen that the this number is between the scatter-partitioned and grid-partitioned systems.

We propose the use of orthogonal polynomials of one variable for the basis functions. Particular system of functions can be chosen according to the specificity of the solved problem. If the input data are normalized on the hypercube $[-1, 1]^n$, the system of Legendre polynomials orthogonal on the interval $[-1, 1]$ with weight $\gamma(x) \equiv 1$ [Bateman, Erdelyi 1953] can be used:

$$P_n(x) = 2^{-n} \sum_{m=0}^{[n/2]} (-1)^m \frac{(2n-2m)!}{m!(n-m)!(n-2m)!} x^{n-2m}, \tag{2}$$

where $[\cdot]$ is the integer part of a number.

Among other possible choices for activation functions we should mention Chebyshev [Patra, Kot 2002; Bodyanskiy et al. 2004] and Hermite [Liying, Khorasani 2005, p. 821-833] polynomials as well as non-sinusoidal orthogonal systems proposed by Haar and Walsh.

3. Synaptic Adaptation

The sum of squared errors will be used as the learning criterion:

$$E(k) = \sum_{p=1}^{p} e^2(k) = \sum_{p=1}^{p} \big(y(p)\big) - \sum_{i=1}^{n} \sum_{j=1}^{h_i} w_{ji} \phi_{ji} \big(x_i(p)\big)^2. \tag{3}$$

For the convenience of further notation, let us re-write the expression for the output of the neural network (1) in the form

$$\hat{y}(k+1) = \phi^T(k+1) W(k), \tag{4}$$

where $\phi(k) = \big(\phi_{11}(x(k)), \phi_{21}\big(x(k)\big), ..., \phi_{h_n n}\big(x(k)\big)\big)^T$ is a $(h \times 1)$ vector of the values of the basis functions for the k-th element of the training set (or at the instant k for sequential processing), $W(k) = (w_1(k), ..., w_h(k))^T$ is a $(h \times 1)$ vector of estimates of synaptic weights at the iteration k.

Since the output of the proposed neural network depends on the tuned parameters linearly, we can use the least squares procedure to estimate them. For sequential processing, e.g. in the case of online identification, we can use the recursive least squares method:

$$\begin{cases} W(k+1) = W(k) + \dfrac{P(k)\big(y(k+1) - W^T(k)\phi(k+1)\big)\phi(k+1)}{1 + \phi^T(k+1) P(k) \phi(k+1)}, \\ P(k+1) = P(k) - \dfrac{P(k)\phi(k+1)\phi^T(k+1)P(k)}{1 + \phi^T(k+1) P(k) \phi(k+1)}. \end{cases} \tag{5}$$

Because of the orthogonality of the basis functions, the matrix $P(k)$ will tend to diagonal form as $k \to \infty$. If the activation functions are orthonormal, $P(k)$ will tend to the unity matrix. Due to this property, the learning procedure will retain numerical stability with the increase of the number of samples in the training sequence.

4. Structure Adaptation

We consider sequential learning that minimizes (3). This leads to the estimate

$$W_h(k) = R_h^{-1}(k) F_h(k), \tag{6}$$

$$R_h^{-1}(k) = R_h^{-1}(k-1) - \frac{R_h^{-1}(k-1)\phi(k)\phi(k)^T R_h^{-1}(k-1)}{1+\phi(k)^T R_h^{-1}(k-1)\phi(k)}, \tag{7}$$

$$F_h(k) = F_h(k-1) + \phi(k)y(k). \tag{8}$$

The use of the recursive least squares (RLS) method and its modifications allows to obtain an accurate and well-interpretable measure of significance of each function in the mapping (1). This mapping can be considered as an expansion of an unknown reconstructed function in the basis $\{\phi_{ji}(.)\}$. Obviously, if the absolute value of any of the coefficients in this expansion is small, then the corresponding function can be excluded from the basis without significant loss of accuracy. The remaining synaptic weights does not need to be retrained if the weight of the excluded node is close to zero. Otherwise, the network should be retrained.

Assume that a vector of synaptic weights $W_h(k)$ of a network comprising h nodes was obtained at the instant k using the formula (6), where the index h determines the number of basis functions (the dimension of $\varphi(k)$). Also assume that the absolute value of the considered parameter $w_h(k)$ is small, and we want to exclude corresponding unit function from the expansion (1). The assumption about the insignificance of the activation h is not restrictive, because we always can renumber the basis functions. This will result only in the rearrangement of the rows and columns in the matrix $R_h(k)$ and in the change of ordering of the elements of the vector $F_h(k)$. However, the rearrangement of columns and/or rows of a matrix does not influence the subsequent matrix operations.

Taking into account the fact that the matrix $R_h(k)$ is symmetric, we obtain:

$$W_h(k) = R_h^{-1}(k)F_h(k) = \begin{pmatrix} R_{h-1}(k) & \beta_{h-1}(k) \\ \beta_{h-1}^T(k) & r_{hh}(k) \end{pmatrix}^{-1} \begin{pmatrix} F_{h-1}(k) \\ f_h(k) \end{pmatrix}, \tag{9}$$

where: $r_{ij}(k)$ is the element of the i-th row and j-th column of the matrix $R_h(k)$,

$\beta_{h-1}(k) = (r_{1h}(k),...,r_{h-1h}(k))^T = (r_{h1}(k),...,r_{hh-1}(k))^T$,

$f_i(k)$ is the i-th element of vector $F_h(k)$.

After simple transformations of (9) we obtain the expression

$$W_h(k) = \begin{pmatrix} W_{h-1}(k) - R_{h-1}^{-1}(k)\beta_{h-1}(k)w_h(k) \\ w_h(k) \end{pmatrix} \tag{10}$$

that enables us to exclude the function from (1) and obtain the corrected estimates of the remaining parameters of the ANN. For this operation, we use only the information accumulated in the matrix $R_h(k)$ and vector $F_h(k)$.

Using the same technique as above, we can obtain a procedure that can be used to add a new function to the existing basis. Direct application of the Frobenius formula [Gantmacher 1990] leads to the algorithm

$$W_{h+1}(k) = R_{h+1}^{-1}(k)F_{h+1}(k) = \begin{pmatrix} R_h(k) & \beta_h(k) \\ \beta_h^T(k) & r_{h+1,h+1}(k) \end{pmatrix}^{-1} \begin{pmatrix} F_h(k) \\ f_{h+1}(k) \end{pmatrix} =$$

$$\begin{pmatrix} W_h(k) + R_h^{-1}(k)\beta_h(k)\dfrac{\beta_h^T(k)W_h(k) - f_{h+1}(k)}{r_{h+1h+1}(k) - \beta_h^T(k)R_h^{-1}(k)\beta_h(k)} \\ \dfrac{-\beta^T(k)W_h(k) + f_{h+1}(k)}{r_{h+1,h+1}(k) - \beta_h^T(k)R_h^{-1}(k)\beta_h(k)} \end{pmatrix}, \tag{11}$$

where $\beta_h(k) = \left(r_{1h+1}(k),...,r_{hh+1}(k)\right)^T = \left(r_{h+11}(k),...,r_{h+1h}(k)\right)^T$.

Thus, with the help of equation (11) we can add a new function (neuron) to the model (1), and exclude an existing function using the formula (10) without retraining remaining weights. In order to perform these operations in real time, it is necessary to accumulate the information about a larger number of basis functions than currently being used. E.g., we can initially introduce a redundant number of basis functions H and accumulate information in the matrix $R_H(k)$ and vector $F_H(k)$ as new data arrive, with only $h < H$ basis functions being used for the description of the unknown mapping. The complexity of the model can be either reduced or increased as required.

Analysis of equations (6), (10), and (11) shows that the efficiency of the proposed learning algorithm is directly related to the condition number of the matrix $R_h(k)$. This matrix will be non-singular if the functions $\{\varphi_i(.)\}_{i=1}^h$ used in the expansion (1) are linear-independent. The best situation is when the function system $\{\varphi_i(.)\}_{i=1}^h$ is orthogonal. In this case, the matrix $R_h(k)$ becomes diagonal, the formulas (6), (10), and (11) being greatly simplified because

$$diag(a_1,..,a_n)^{-1} = diag\left(\frac{1}{a_1},...,\frac{1}{a_n}\right), \tag{12}$$

where $diag(a_1,..,a_n)$ is an $(n \times n)$ matrix with non-zero elements $a_1,..,a_n$ only on the main diagonal.

5. Simulation Results

We have applied the proposed ontogenic network with orthogonal activation functions to online identification of a rat's (*Ratus Norvegius Vistar*) brain activity during sleeping phase.

The signal was measured with frequency of 64 Hz. We took a fragment of signal containing 3200 points (50 second of measuring), that was typical for sleeping phase of rat's life activity. Two neural networks of type (1) were trained in real-time. Each network had 10 inputs – delayed signal values ($y(k), y(k-1),\ldots, y(k-9)$) and was trained to output one-step ahead value of the process – $y(k+1)$. First network utilized synaptic adaptation algorithm (6) while second one also involved the structure adaptation technique (10), (11). Initially both ANNs had 5 activation functions per input, the one with synaptic adaptation only retained all 50 tunable parameters during it's work while ANN with structure adaptation mechanism had only 25 fired functions (the most significant ones chosen in real-time). For the results comparing purpose we also trained multilayer perceptron (further referred as MLP) with the same structure of inputs and training signal, having 5 units in the 1st and 4 in the 2nd hidden layers (that totals to 74 tunable parameters). As MLP is not capable of real-time data processing, all samples are used as training set and test criteria are calculated on the same data points. MLP was trained during 250 epochs with Levenberg-Marquardt algorithm. Our research showed that this is enough to achieve precision comparable to proposed ontogenic neural network with orthogonal activation functions.

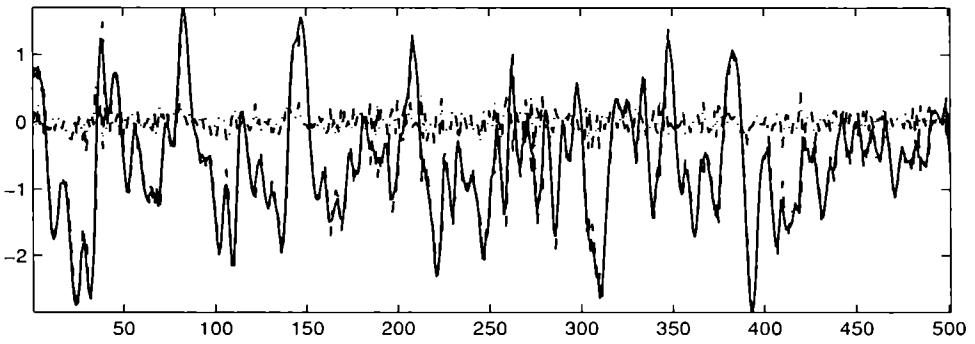

Fig. 1. Identification of a rat's brain activity during sleeping phase using proposed neural network with orthogonal activation functions – brain activity signal (*solid line*), network output (*dashed line*), and identification error (*dash-dot line*)

Results of identification can be found in table 1. Figure 1 shows the results of identification using proposed neural network. We used some different measures of identification quality. First, we analyse normalized root mean squared error, that is closely related to the learning criterion. Two other criteria used: "Wegstrecke" [Baumann 1996] characterizes the quality of the model for prediction/identification (+1 means perfect one), "Trefferquote" [Fueser 1995] is percent value of correctly predicted direction changes.

We can see that utilizing structure adaptation technique leads to somewhat worth results. This is the tradeoff for having less tunable parameters and possibility to process non-stationary signals.

Table 1. Identification results for different architectures

Decription	NRMSE	Trefferquote	Wegstrecke
OrthoNN, real-time processing	0.1834	82.3851	0.85221
OrthoNN, real-time processing, variable number of nodes	0.2187	77.6553	0.74872
MLP, offline learning (250 epochs), error on the training set	0.1685	83.9533	0.87192

6. Conclusion

A new computationally efficient neural network with orthogonal activation functions was proposed. It has a simple and compact architecture not affected by the curse of dimensionality, and provides high precision of nonlinear dynamic system identification. An apparent advantage is much easier implementation and lower computational load as compared to the conventional neural network architectures.

The approach presented in the paper can be used for nonlinear system modeling, control, and time series prediction. An interesting direction of further work is the use of the network with orthogonal activation functions as a part of hybrid multilayer architecture. Another possible application of proposed ontogenic neural network is its use as a basis for diagnostic systems.

References

Bateman H., Erdelyi A., *Higher Transcendental Functions*, Vol. 2, McGraw-Hill, 1953.

Baumann M., *Nutzung Neuronale Netze zur Prognose von Aktionkursen*, Report Nr. 2/96, TU Ilmenau, 1996, 113 S.

Bodyanskiy Ye.V., Kolodyazhniy V.V., Slipchenko O.M., *Forecasting Neural Network with Orthogonal Activation Functions*, [in:] Proc. of 1st Int. conf. *Intelligent decision-making systems and information technologies*, Chernivtsi, Ukraine, 2004, p. 57 (in Russian).

Cun Y.L., Denker J.S., Solla S.A., *Optimal Brain Damage*, Advances in Neural Information Processing Systems, 2, 1990, p. 598-605.

Fahlman S.E., Lebiere C., *The Cascade-correlation Learning Architecture*, Technical Report CMU-CS-90-100, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1990.

Fueser K., *Neuronale Neteze in der Finanzwirtshaft*, Gabler, Wiesbanden 1995, 437 S.

Gantmacher F.R., *The Theory of Matrices*, Chelsea Publ. Comp., New York 1977.

Handbook of Neural Computation, IOP Publishing and Oxford University Press, 1997.

Hassibi B., Stork D.G., *Second-order Derivatives for Network Pruning: Optimal Brain Surgeon*, [in:] *Advances in Neural Information Processing Systems*, Hanson et al. (eds.), 1993, p. 164-171.

Liying M., Khorasani K., *Constructive Feedforward Neural Network Using Hermite Polinomial Activation Functions*, IEEE Trans. On Neural Networks, 16, No. 4, 2005, p. 821–833.

Nag A., Ghosh J., *Flexible Resource Allocating Network for Noisy Data*, [in:] Proc. SPIE Conf. on Applications and Science of Computational Intelligence, SPIE Proc. Vol. 3390, Orlando, Fl., April 1998, p. 551-559.

Narendra K.S., Parthasarathy K., *Identification and Control of Dynamic Systems Using Neural Networks*, IEEE Trans. on Neural Networks, 1, 1990, p. 4-26.

Nelles O., *Nonlinear System Identification*, Springer, Berlin 2001.

Patra J.C., Kot A.C., *Nonlinear Dynamic System Identification Using Chebyshev Functional Link Artificial Neural Network*, IEEE Trans. on System, Man and Cybernetics – Part B, 32, 2002, p. 505-511.

Platt J., *A Resource Allocating Network for Function Interpolation*, Neural Computation, 3, 1991, p. 213-225.

Poggio T., Girosi F., *A Theory of Networks for Approximation and Learning*, A.I. Memo No. 1140, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1989.

Prechelt L., *Connection Pruning with Static and Adaptive Pruning Schedules*, Neurocomputing, 16, 1997, p. 49-61.

Scott I., Mulgrew B., *Orthonormal Function Neural Network for Nonlinear System Modeling*, [in:] Proceedings of the International Conference on Neural Networks (ICNN-96), June 1996.

Takagi T., Sugeno M., *Fuzzy Identification of Systems and its Application to Modeling and Control*, IEEE Trans. on System, Man and Cybernetics. 15, 1985, p. 116-132.

Yingwei L., Sundararajan N., Saratchandran P., *Performance Evaluation of a Sequential Minimal Radial Basis Function (RBF) Neural Network Learning Algorithm*, IEEE Trans. on Neural Networks, 9, 1998, p. 308-318.

ONTOGENICZNE SIECI NEURONOWE Z WYKORZYSTANIEM ORTOGONALNEJ FUNKCJI AKTYWACJI

Streszczenie

Artykuł zawiera propozycję utworzenia ontogenicznej inteligentnej sieci neuronowej (ANN). Sieć działa, opierając się na ortogonalnej funkcji aktywacji, co znakomicie przyczynia się do redukcji złożoności obliczeniowej. Inną korzyścią tego podejścia jest stabilność numeryczna, ponieważ system aktywacji funkcji z definicji jest liniowo niezależny. Dla sieci ANN opracowana została procedura ucząca zapewniająca przez parametr przestrzeni konwergencję z minimum globalnym funkcji błędu. Algorytm umożliwia dodanie lub usunięcie węzła w czasie rzeczywistym bez ponownego uczenia sieci. Otrzymane wyniki symulacji potwierdzają efektywność proponowanego podejścia.