Proceedings of the IV Brazilian Conference on Neural Networks - IV Congresso Brasileiro de Redes Neurais
pp. 247-251, July 20-22, 1999 - ITA, São José dos Campos - SP - Brazil

Preliminary Testing and Analysis of an Adaptive Neural Network Training Kalman Filtering Algorithm

Jaime Augusto da Silva[1], Atair Rios Neto[2]
[1]Instituto Nacional de Pesquisas Espaciais
Divisão de Mecânica Espacial e Controle
12228-000 São José dos Campos, SP
[2]Universidade do Vale do Paraíba
Instituto de Pesquisa e Desenvolvimento
12244-000 São José dos Campos, SP
E-mails: jaimes@univap.br, atair@univap.br

Abstract

An adaptive neural network training Kalman filtering algorithm is implemented. The XOR problem and a benchmark problem for diagnosis of breast cancer are used for testing and analysis of the algorithm behavior. Results show that the algorithm performs well on both problems having the desirable characteristics of beeing simple to implement, with parallel processing features and good numerical behavior due to the adaptive distribution of learning.

1. Introdution

Due to the slow nature of network training with the standard back-propagation algorithm, a great deal of research effort has been expended in the linear adaptive filtering literature with the objetive of using Kalman filter approach to train feedforward neural networks ([1], [2], [3], [4], [5]). It is known that the speed of convergence of the back-propagation algorithm is that of a gradient method based algorithm [6]. On the other hand, Kalman filtering algorithms utilize information contained in the input data more effectively. In this case, the use of Kalman filtering algorithms offers the desirable advantages of fast speed of convergence, a built-in learning-rate parameter and, insensitiveness to variations in the condition number of the input data.

Singhal and Wu [3] have proposed the use of the recursive least-squares RLS (Kalman) algorithm by defining a quadratic cost function and linearizing it with respect to synaptic weights, at each working point. The result of such an approach is a complex computational problem because it requires storage and updating of an error covariance matrix whose size is related with the square of the number of synaptic weights in the network. To overcome this problem, Shah and Palmieri [1] have suggested a simplified version of this algorithm. They considered that at neuron level, the activation function may be expanded about the current estimate of the weights by using Taylor series. Doing so, they arrived at a pair of equations that describes the linearized dynamic behavior of a specific neuron. Then it is possible to use standard RLS algorithm to make an estimate of the sinaptic weight vector of the neuron considered.

Here, Kalman filtering algorithm is also used but the approach employed is a more natural one. For one thing, it will allow us to immediately determine what kind of simplifications and approximations are being done. The simplified version of the considered algorithm features parallel processing, for training feedforward neural networks [7]. Such algorithms may present bad numerical behavior and divergence or slow speed of convergence usually occurs as a large number of input-output data sets are processed. In this case, the algorithm looses its capacity of learning as new data are processed. To prevent such a situation, a procedure to estimate the noise level adaptively in order to compensate for the errors and increase the speed of convergece, is incorporated [7].

The resulting algorithm is preliminarily tested on two well known problems, the XOR problem and a benchmark problem for diagnosis of breast cancer, for analysis and evaluation of its numerical performance in the supervised training of multilayer perceptron neural networks.

2. Kalman Filtering Supervised Training

Let us assume that an artificial neural network is to learn and represent a nonlinear continuous mapping

$$f \in C : x \in D \subset \Re^n \rightarrow y \in \Re^m \qquad (1)$$

Such a neural net can be viewed and treated as a parameter mapping like

$$\hat{y}(t) = \hat{f}(x(t), w) \qquad (2)$$

where w is the matrix of weight parameters to be identified by fitting a given data set of input-output patterns

$$\{(x(t), y(t)) : y(t) = f(x(t)), t = 1,2,\cdots, L\} \quad (3)$$

Considering that a multilayer perceptron is characterized by the weight matrix w that represents the free parameters of all the neurons in the network, we can now stablish a cost function, to be minimized during training, which is itself dependent on w. Let the following functional represent such a cost function:

$$J(w) = \frac{1}{2}\left[(w - \bar{w})^T \bar{P}^{-1} (w - \bar{w}) + \sum_{t=1}^{L} \left((y(t) - \hat{f}(x(t), w))^T R^{-1}(t) (y(t) - \hat{f}(x(t), w)) \right) \right] \quad (4)$$

given an a priori estimate of the weights $\bar{w}$, an input-output data set $\{x(t), y(t) : t = 1,2,\cdots, L\}$ and two weight matrices, $\bar{P}$ and R, that are related with the quality of the initial estimate $\bar{w}$ and the quality of the input-output data set.

Let us suppose that the mapping given by Eq. 2 may be expanded in Taylor series. Retaining only the first order terms of the series expantion we assume that

$$\hat{y}(t) \approx \bar{y}(t,i) + \hat{f}_w(x(t), \bar{w}(i))(w - \bar{w}(i)) + \vartheta(w) \quad (5)$$

Rios Neto [7] has proposed that in a typical ith iteration, the following linear perturbation condition may be adopted as a result of the expansion above:

$$\alpha(i)[\hat{y}(t) - \bar{y}(t,i)] \cong \hat{f}_w(x(t), \bar{w}(i))[w(i) - \bar{w}(i)] \quad (6)$$

where i=1,2,...,I; $\bar{w}(i)$ is the a priori estimate of w coming from the previous iteration, starting with $\bar{w}(1) = \bar{w}$; $\bar{y}(t,i) = \hat{f}(x(t), \bar{w}(i))$; $\hat{f}_w(x(t), \bar{w}(i))$ is the matrix of first partial derivatives with respect to w; and, $0 \le \alpha(i) \le 1$ is a parameter to be ajusted in order to guarantee the hypothesis of linear perturbation. The cost function J(w) then assumes the following form

$$J(w(i)) = \frac{1}{2}\left[(w(i) - \bar{w}(i))^T \bar{P}^{-1} (w(i) - \bar{w}(i)) + \sum_{t=1}^{L} (z(t,i) - H(t,i)w(i))^T R^{-1}(t) (z(t,i) - H(t,i)w(i)) \right] \quad (7)$$

where the following compact notation has been adopted:

$$z(t,i) \equiv \alpha(i)[y(t) - \bar{y}(t,i)] + \hat{f}_w(x(t), \bar{w}(i))\bar{w}(i) \quad (8)$$

$$H(t,i) \equiv \hat{f}_w(x(t), \bar{w}(i)) \quad (9)$$

The minimization of the functional given by Eq. 7 is formally equivalent to the following stochastic linear estimation problem:

$$\bar{w}(i) = w(i) + \bar{e} \quad (10)$$

$$z(t,i) = H(t,i)w(i) + \nu(t) \quad (11)$$

where $\bar{e}$ is the a priori estimation error vector and $\nu(t)$ is the observation error vector; $E[\nu(t)] = 0$ $E[\nu(t)\nu^T(t)] = R(t)$, $E[\bar{e}] = 0$, $E[\bar{e}\bar{e}^T] = \bar{P}$; $\bar{e}$ and $\nu(t)$ are assumed to be gaussian distributed and are not correlated; R(t) is usually diagonal; E[.] is the expectation value operator.

Now, following closely Rios Neto [7], a simplified version of the algorithm may be obtained if the following simplifications and approximations are made:

1 – Eqs. 10 and 11 will be applied at neuron level. So, the problem of training the neural network will be reduced to a local estimation problem;

2 – information relative to a priori errors in the a priori estimates of other weights is disregarded.

Then, Eqs. 10 and 11, when applied to a specific neuron at layer k, takes the form:

$$\bar{w}_{kl}(i) = w_{kl}(i) + \bar{e}_{kl} \quad (12)$$

$$\alpha(i)[y(t) - \bar{y}(t,i)] = \hat{f}_{w_{kl}}(x(t), \bar{w}(i))[w_{kl}(i) - \bar{w}_{kl}(i)] + \nu(t) \quad (13)$$

In a simplified notation, grouping the observations in a vector, for t=1,2,...,L, Eq. 13 becomes:

$$z_{kl}(i) = H_{kl}(i)w_{kl}(i) + \nu \quad (14)$$

Thus, in a typical iteration i, a Gauss-Markov estimator (Kalman form) may be applied resulting that:

$$\hat{w}_{kl}(i) = \bar{w}_{kl}(i) + K_{kl}(i)[z_{kl}(i) - H_{kl}(i)\bar{w}_{kl}(i)] \quad (15)$$

$$K_{kl}(i) = \bar{P}_{kl} H_{kl}^T(i)\left[H_{kl}(i)\bar{P}_{kl} H_{kl}^T(i) + R_{kl}\right]^{-1} \quad (16)$$

$$P_{kl}(i) = \left[I - K_{kl}(i)H_{kl}(i)\right]\bar{P}_{kl} \quad (17)$$

$$\bar{w}_{kl}(i+1) = \hat{w}_{kl}(i), \quad \alpha(i) \leftarrow \alpha(i+1) \quad (18)$$

Inversion matrix of Eq. 15 is avoided if Eq. 14 is recursively processed rowwise.

3. Adaptive Solution

In practice, it is observed that as the Kalman estimator processes many data pairs, its performance can be seriously degraded due to un unrealistic decrease in the covariance matrix. Under normal conditions the filter should improve the estimates precision level while additional data is processed, with a correspondent decrease of the calculated variances. However, numerical errors and observation model errors can lead to divergence of the estimator as the observations are processed.

To avoid this condition and keep a distributed and uniform capacity of learning, Rios Neto [7] has proposed an adaptive procedure based on a criterion of statistical consistency to balance a priori information priority with that of new learning information. He considered that in a typical ith iteration and for t=1,2,...,L-1

$$w(i,t+1) = w(i,t) + \eta(t) \quad (19)$$

$$E[\eta(t)] = 0, \quad E\left[\eta(t)\eta^T(\tau)\right] = Q(t)\delta_{t\tau} \quad (20)$$

where $\delta_{t\tau}$ is the Kronecker symbol and Q is the diagonal matrix $Q(t) = \text{diag}\left[q_j(t):\ j = 1,2,\cdots,n_w\right]$.

With this modeling approximation for the neural weights, learning from the th input-output data pattern is transformed in the estimation problem:

$$w(i,t) = w(i,t) + \bar{e}(i,t) \quad (21)$$

$$z(t,i) = H(t,i)w(i,t) + v(t) \quad (22)$$

starting with $\bar{e}(i,1) = \bar{e}$, $\bar{w}(i,1) = \bar{w}(i)$ and for t=1,2,...,L.

To propagate estimates from t to t+1 Kalman filter predictor is used considering the dynamics of Eq. 19:

$$\bar{w}(i,t+1) = \hat{w}(i,t) \quad (23)$$

$$P(i,t+1) = P(i,t) + Q(t) \quad (24)$$

where $P(i,t+1) = E\left[\bar{e}(i,t+1)\bar{e}^T(i,t+1)\right]$ and P(i,t) is given by the filtering algorithm Eq. 17 where $\bar{P}(i,t)$ starts with $\bar{P}(i,1) = E\left[\bar{e}\bar{e}^T\right] = \bar{P}$.

The adaptation is done by adjusting the noise η(t) dispersion such as to keep statistical consistency to attain distributed learning:

$$\beta E\left[v_j^2(t+1)\right] = H_j(t+1,i)\left[P(i,t) + Q(t)\right] H_j^T(t+1,i) \quad (25)$$

where $1 \le \beta \le \beta_{l_{mx}}$, j=1,2,...,m and β is to be adjusted in order to have distributed learning.

In order to use the same Kalman filtering algorithm, the following associated estimation problem can be established [7] by considering Eq. 25 as a pseudo observation and imposing an a priori information on q(t):

$$0 = q(t) + \bar{e}^q \quad (26)$$

$$z^q(t+1,i,\beta) = H^q(t+1,i)q(t) + v^q(t+1) \quad (27)$$

$$E\left[\bar{e}^q\right] = 0, \quad E\left[\bar{e}^q\bar{e}^{q^T}\right] = I_{n_w} \quad (28a)$$

$$E\left[v^q(t+1)\right] = 0$$
$$E\left[v^q(t+1)v^{q^T}(t+1)\right] = R^q(t+1) = 0 \quad (28b)$$

which is a problem with exact observations that can be processed with a Kalman filtering. It should be observed that the a priori information on q(t) gives a $\hat{q}(t)$ which is close to zero in magnitude and that the components $\hat{q}_k(t)$ are taken to be zero whenever the solution obtained results in a value less than zero.

4. Test and Analysis of the Algorithm

4.1. Test description

A preliminary test of the algorithm was made with the Exclusive OR (XOR) problem, which may be viewed as a special case of a more general problem, namely, that of classifying points in the unit hypercube. The four corners of the unit square correspond to the input patterns and are the points (0,0), (0,1), (1,1), and (1,0). The first and third input patterns are in class 0, as shown by

0 XOR 0 = 0 1 XOR 1 = 0

The other two input pars are in class 1, that is

$$0 \text{ XOR } 1 = 1 \qquad 1 \text{ XOR } 0 = 1$$

In this case, the function f that the neural network is to learn is the mapping given by the truth table of the XOR problem.

A multilayer perceptron network with configuration MP_{2x5x1} (having 5 hidden units), fully forward connected between adjacent layers, with hidden and output units with bias, and the standard sigmoid as the activation function was used. Initial values of weights were randomly chosen between –0.5 and 0.5 with uniform distribution.

Solution of Exclusive OR problem was obtained with use of Kalman filtering supervised training algorithm solely. Test of the adaptive solution was done with a more involved benchmark problem, the cancer problem.

Cancer of PROBLEN 1 [8] is a pattern recognition and classification problem of diagnosis of breast cancer. It presents the classification of a tumor as either benign or malignant based on cell descriptions. It contains 9 real coefficient inputs, represented as a decimal number between 0 and 1, 2 boolean output coefficients always represented as either 0 (false) or 1 (true) and 699 examples. Suggested benchmark rules says that 350 patterns should be used for training, 175 for validation and 174 for testing. As before, the function f is given by the input-output data mapping used for the neural network training.

Solution of the cancer problem was obtained with a multilayer perceptron network with configuration $MP_{10x10x2}$. As before, the networks were fully forward connected between adjacent layers, with hidden and output units with bias and the standard sigmoid as the activation function. The initial value of the weights also were initialized with values between –0.5 and 0.5.

The software developed for numerical analysis of the algorithm was implemented in a PC, Pentium, 200 MHZ, in Windows 95 in FORTRAN 77 language. Criterion of completion of training was either a pre-stablished value of mean squared error or a limit number of iterations.

4.2. Simulation and results

Figure 1, a plot of the mean-squared error as function of the iteration number, shows results obtained when training with Kalman filtering-based algorithm, a network to solve the XOR problem. The algorithm was initialized with conditions: α=0.2, P_{kl}=I and, R=0.01I. Six simulations, considering different initial conditions for the weight matrices, were performed. Results suggest that, for this problem, the algorithm is insensitive to the initial value of the weight matrix.

On Figure 2, when no adaptation is used, for the cancer problem with $\overline{P}_{kl} = I$, R=0.0001I, α=0.2, the trace of the covariance matrix P was plot as a function of the number of processed patterns to evaluate the behavior of the estimated variances of errors. The curves shown were taken at different iterations.

Typical curves of mean-squared error as function of interation number for the cancer problem, when the

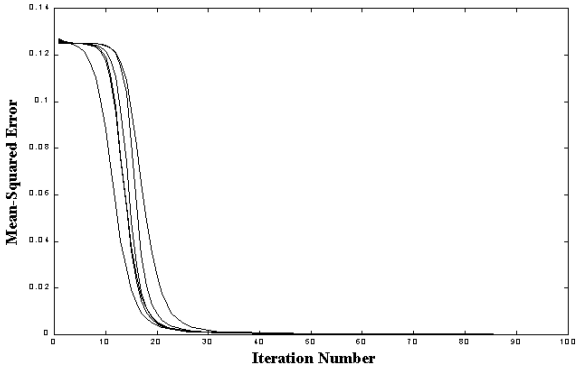

Figure 1: Extended OR Problem

adaptive algorithm is used, are shown on Figure 3.

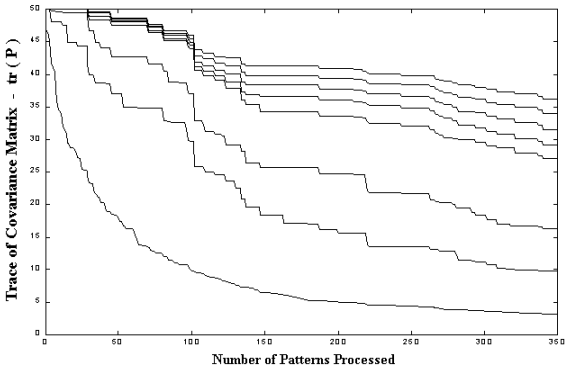

Figure 2: Cancer Problem

Initial conditions were set as: $\overline{P}_{kl} = I$ and, R=0.0001I. The learning parameter α was set from values ranging

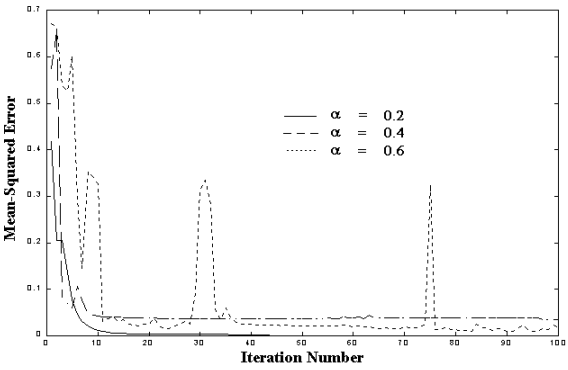

Figure 3: Cancer Problem

from 0.2 to 0.6, in order to verify its influence on the stability and speed of convergence of the algorithm.

Also, since the adaptive solution was used, the parameter β, which must be adequately specified in order to guarantee distributed learning, was set equal to 1.

An analysis of the results shown suggests the following trends:

- For the first iterations, the trace of the covariance matrix P decreases fast as the patterns are processed. This indicates that the algorithm learns too well the wrong information. As the number of iterations increases, the matrix trace decreases in one iteration, as expected, but now the rate of decrease is slower. When the adaptive solution is considered, it increases the value of the matrix trace at specified number of patterns processed intervals, allowing the increase of the speed of convergence.
- A smaller learning-rate parameter α results in a better behavior of the algorithm. Note that as α increases the algorithm tends to become unstable. A small α allows the algorithm locate a "deeper" local minima while bigger values of α causes oscillations in the mean-squared error during iterations and a higher value for the final mean-squared error at convergence, both of which are undesirable effects. Also, for bigger values of α the algorithm does not behave well because the linear perturbation condition is not being observed anymore.
- For the cancer problem, the cross-validation data showed that classification errors less than 4.5 per cent could be obtained, which are equivalent in quality to those obtained by Souza Filho and Rios Neto [10] using a more sophisticated and complicated Kalman filtering heuristic scheme.
- Instead of using the adaptive solution to increase the rate of convergence of the algorithm, an heuristic approach could be used. Begining with a small α this parameter could be increased as the iterations proceeded. This situation was tested but, besides being a more elegant solution, various initial conditions results showed that the adaptive solution performs better than the heuristic approach.

5. Conclusions

A simplified local processing and adaptive version of a neural network training Kalman filtering algorithm was implemented. Preliminary tests and analysis of the algorithm were done by applying it to two benchmark problems. The results obtained are encouraging showing that when the algorithm is used with the adaptive solution it is capable of processing a large number of patterns without loosing its capacity to continually extract information from new data.

On the ohter hand, the algoritm is simple to implement, with the desirable characteristics of parallel processing, and has showed a fast rate of convergence for the two problems analysed.

Acknowledgement

This research has been partially suported by CNPq grants.

References

[1] S. Shah, and F. Palmieri. MEKA – A fast, local algorithm for training feedforward neural networks. International Joint Conference on Neural Networks. San Diego, CA, Vol. 3, pages 42-46, 1990.

[2] Y. Iiguni, H. Sakai, and H. Tokumaru. A real-time learning algorithm for a multilayered neural network based on the extended Kalman filter. IEEE Transactions on Signal Processing, 40(4), pages 959-966, 1992.

[3] S. Singhal, and L. Wu. Training multilayer perceptrons with the extended Kalman algorithm. Advances in Neural Information Processing Systems, Vol. 1, pages 133-140, Morgan Kaufman Pub. Inc., 1989.

[4] G. Chen, and H. Ögmen. Modified extended Kalman filtering for supervised learning. Int. J. Systems Sci., 24(6), pages 1207-1214, 1993.

[5] K. Watanabe, T. Fukuda, and S. G. Tzafestas. Learning algorithms of layered neural networks via extended Kalman filters. Int. J. System Science, 22(4), pages 753-768, 1991.

[6] S. Haykin. Neural Networks – A Comprehensive Foundation. Macmillan College Publishing Company, Inc., NJ, 1994.

[7] A. Rios Neto. Stochastic optimal linear parameter estimation and neural nets training in systems modeling. RBCM – J. of the Braz. Soc. Mechanical Sciences, Vol. XIX, No. 2, pages 138-146, 1997.

[8] L. Prechelt. PROBEN 1 – A set of neural network benchmark problems and benchmarking rules. Technical Report 21/94, Fakultät für Informatik, Universität Karlsruhe, D-76128, Karlsruhe, Germany, 1994. Anonymous FTP: /pub/papers/techreports/1994/1994-21.ps.Z on ftp.ira.uka.de.

[9] A. Rios Neto, and H. K. Kuga. Kalman filtering state noise adaptive estimation. Proceedings of Second IASTED Int. Conference in Telecom and Control, TELECON'85, Rio de Janeiro, Brasil, pages 210-213, 1985.

[10] D. F. Souza Filho, and A. Rios Neto. Test of a local processing neural network training Kalman filtering algorithm on a breast cancer diagnosis benchmark problem. XII Congresso Brasileiro de Automática, Uberlândia, Brasil, 1998.