TRAINING RBF NETWORKS WITH AN EXTENDED KALMAN FILTER OPTIMIZED USING FUZZY LOGIC

Jun Wang, Li Zhu, Zhihua Cai, Wenyin Gong , Xinwei Lu
School of Computer, China University of Geosciences, Wuhan 43007, P. R. China
junwang8151@163.com

Abstract In this paper we propose a novel training algorithm for RBF networks that is based on extended kalman filter and fuzzy logic.After the user choose how many prototypes to include in the network, the extended kalman filter simultancously solves for the prototype vectors and the weight matrix.The fuzzy logic is used to cope with the devergence problem caused by the insufficiently known a priori filter statistics. Results are presented on RBF networks as applied to the Iris classification problem. It is shown that the use of the extended Kalman filter and fuzzy logic results in faster learning and better results than conventional RBF networks.

Keywords: kalman filter, fuzzy logic, RBF networks

1. Introduction

Radial Basis Functions emerged as a variant of artificial neural network in late 1980's. Their excellent approximation capabilities have been studied in [1,2]. RBF networks have been successfully applied to a large diversity of applications including interpolation [6], system identification, control engineering [7], data fusion [8], etc.

Training a neural network is, in general, a challenging nonlinear optimization problem. Various derivative-based methods have been used to train neural networks, including gradient descent [3], Kalman filtering [4, 5], and backpropagation [9], etc. Gradient descent training of RBF networks has proven to be much more effective than more conventional methods [3].However; gradient descent training can be computationally expensive. Another method based on Kalman filtering proves to be quicker than gradient descent training [10]. However, a significant difficulty in designing a KF (refers to both LKF and EKF) can often be traced to incomplete a priori knowledge of the process noise covariance matrix Q and measurement noise covariance matrix R. It has been shown that insufficiently known a priori filter statistics can on the one hand reduce the precision of the estimated filter states or introduces biases to their estimates. In addition, incorrect a priori information can lead to practical divergence of the filter.

This paper extends the results of [10] and formulates a training method for RBFs based on extended kalman filter and fuzzy logic. The fuzzy logic techniques are used to adjust the R matrix of the extended kalman filter so that the method can be self-tuning and adaptive. This idea comes from [11, 12].We refer this method as FKF which means fuzzy adaptive kalman filter. There have been studies and applications on extended kalman filter and fuzzy logic. However, this paper is the first known use of these techniques to train the RBF network. We demonstrate the proformance of the method on the Iris classification problem and compare it with RBF optimization using gradient descent and extend kalman filter. It is shown that the new method converges more quickly than gradient descent and finds a better solution than extend kalman filter.

2. RBF Network

A radial basis function (RBF) neural network is trained to perform a mapping from an m-dimensional input space to an n-dimensional output space. Suppose there are c neurons in the hidden layer. Each of the c neurons in the hidden layer applies an activation function which is a function of the Euclidean distance between the input and an m-dimensional prototype vector. Each hidden neuron contains its own prototype vector as a parameter. The output of each hidden neuron is then weighted and passed to the output layer. The outputs of the network consist of sums of the weighted hidden layer neurons.

In this paper, the RBF network is used in supervised applications. we set the hidden layer functions of the form of Eq.(1)

$$g(\| x - v \|^2) = (\| x - v \|^2 + 1)^{-\frac{1}{3}} \tag{1}$$

Where x is the input matrix and v is the prototype matrix, and $\| \cdot \|^2$ is the sum of the squares of the elements of the matrix.Our task is to minimize the training error, we can define the error function:

$$E = \frac{1}{2} \| Y - \hat{Y} \|_F^2 \tag{2}$$

Where Y is the matrix of the target (desired) value for the RBF output, $\hat{Y}$ is the matrix of the actual value of the RBF output, and $\| \cdot \|_F^2$ is the square of the Froebinius norm of a matrix, which is equal to the sum of the squares of the elements of the matrix.

3. Fuzzy adaptive kalman filter

3.1 Extended kalman filter

The extended kalman filter is a widely used estimation algorithm.In this section we briefly outline the algorithm and give the extended kalman recursion.More details of the extended kalman filter are widely available in the literature[14]. Consider a nonlinear finite dimensional discrete time system of the form:

$$\begin{aligned} \theta_{k+1} &= f(\theta_k) + \omega_k \\ y_k &= h(\theta_{k-1}) + \nu_k \end{aligned} \tag{3}$$

Where the vector θ_k is the state of the system at time k, $\omega_k$is the process noise, y_k is the observation vector, $\nu_k$is the observation noise, and $f(\cdot)$ and $h(\cdot)$ are nonlinear vector functions of the state. Assume that the initial state θ_0 and the noise sequences $\{\nu_k\}$ and $\{\omega_k\}$ are Gaussian and independent from each other with $AE(\theta_0) = \bar{\theta}_0, AE[(\theta_0 - \bar{\theta}_0)(\theta_0 - \bar{\theta}_0)^T] = p_0, AE(\omega_k) = 0, AE(\omega_k \omega_t^T) = Q\delta_{kt}, AE(\nu_k) = 0, AE(\nu_k \nu_t^T) = R\nu_{kt}$, where $AE(\cdot)$ is the expectation operator and δ_{kt} is the Kronecker delta. The problem addressed by the extended Kalman filter is to find an estimated $\hat{\theta}_{n+1}$ of θ_{k+1} given y_j $(j = 0, \ldots, k)$ by the recursion

$$\begin{aligned} \hat{\theta}_k &= f(\hat{\theta}_k) + K_k[y_k - h(\hat{\theta}_{k-1})] \\ K_k &= P_k H_k (R + H_k^T P_k H_k)^{-1} \\ P_{k+1} &= F_k(P_k - K_k H_k^T P_k) F_k^T + Q \end{aligned} \tag{4}$$

where F_k and H_k^T can be obtained by

$$\begin{aligned} F_k &= \frac{\partial f(\theta)}{\partial \theta}|_{\theta = \hat{\theta}_k} \\ H_k^T &= \frac{\partial h(\theta)}{\partial \theta}|_{\theta = \hat{\theta}_k} \end{aligned} \tag{5}$$

K_k is known as the kalman gain,Q is the process noise covariance matrix and R is the measurement noise covariance matrix.

3.2 Fuzzy adaptive Kalman Filter

The optimality of the estimation algorithm in the KF setting is closely connected to the quality of a priori information about the process and measurement noise. If a priori filter statistics is insufficiently known, the precision of the estimated filter states can be reduced and biases to the estimates may be introduced. In addition, incorrect a priori information can lead to practical divergence of the filter. From the aforementioned it may be argued that the conventional KF with fixed R and/or Q should be replaced by an adaptive estimation formulation. In this paper an innovation adaptive estimation (IAE) approach coupled with fuzzy logic techniques is used to adjust the R matrix of the KF. Here the innovation Inn_k at sample time k in the KF algorithm is the difference between the real measurement y_k,received by the filter and its estimated (predicted) $\hat{y}_k$, and is computed as follows:

$$Inn_k = y_k - \hat{y}_k \tag{6}$$

The predicted measurement is the projection of the filter predicted states $\hat{\theta}_{k-1}$ onto the measurement space,

$$y_k = h(\hat{\theta}_{k-1}) \tag{7}$$

The actual covariance is defined as an approximation of the Inn_k sample covariance through averaging inside a moving estimation window of size N which takes the following form:

$$\hat{C}_{r_i} = \frac{1}{M} \sum_{i=i_0}^{N} (Inn_k Inn_k^T) \tag{8}$$

Where $i_0 = k - M + 1$ is the first sample inside the estimation window. An empirical experiment is conducted to choose the window size M. From experimentation it was found that a good size for the moving window in Eq.(8) is 15.

The theoretical covariance of the innovation sequence is defined as

$$S_k = H_k P_k^- H_k^T + R_k \tag{9}$$

The logic of the adaptation algorithm using covariance matching technique can be qualitatively described as follows. If the actual covariance value $\hat{C}_{r_i}$ is observed, whose value is within the range predicted by theory S_k and the difference is very near to zero, this indicates that both covariances match almost perfectly and only a small change is needed to be made on the value of R. If the actual covariance is greater than its theoretical value, the value of R should be decreased. On the contrary, if $\hat{C}_{r_i}$ is less than S_k, the value of R should be increased. This adjustment mechanism lends itself very well to being dealt with using a fuzzy-logic approach based on rules of the kind:

$$\text{IF} < antecedent > \text{THEN} < consequent > \tag{10}$$

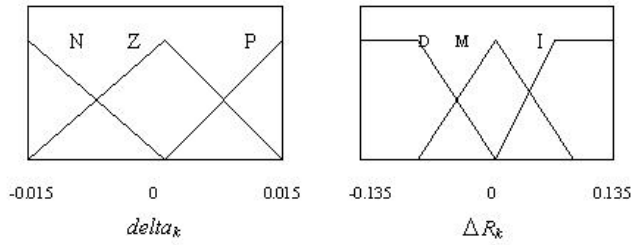

Figure 1. Membership function of $delta_k$ and $\triangle R_k$.

Where antecedent and consequent are of the form $x \in M_i$, $y \in N_i$, $i = 1, 2, \ldots$ respectively, where x and y are the input and output variables respectively, and M_i and N_i are the fuzzy sets.

To implement the above covariance matching technique using the fuzzy logic approach, a new variable called delta, is defined to detect the discrepancy between $\hat{C}_{r_i}$ and S_k. The following three fuzzy rules of the kind (10) are used :

IF $< delta_k \cong 0 >$ THEN $< R_k$ is unchanged $>$
IF $< delta_k > 0 >$ THEN $< R_k$ is unchanged $>$
IF $< delta_k < 0 >$ THEN $< R_k$ is unchanged $>$

Thus R is adjusted according to $R_k = R_{k-1} + \triangle R_k$, where $\triangle R_k$ is added or subtracted from R at each instant of time. Here $delta_k$ is the input to the fuzzy inference system (FIS) and $\triangle R_k$ is the output.

On the basis of the above adaptation hypothesis, the FIS can be implemented using three fuzzy sets for $delta_k$; $N = Negative, Z = Zero$ and $P = Positive$. For $\triangle R_k$ the fuzzy sets are specified as $I = Increase, M = Maintain$ and $D = decrease$. The membership functions of these fuzzy sets are shown in Fig.1.

By excuting the FIS, we obtain the adjusting value $\triangle R$ of R.Thus,the extended kalman filter recursion of Eq.(4) can be modified as follows:

$$\begin{aligned} \hat{\theta}_k &= f(\hat{\theta}_k) + K_k[y_k - h(\hat{\theta}_{k-1})] \\ K_k &= P_k H_k (R_k + H_k^T P_k H_k)^{-1} \\ R_k &= R_{k-1} + \triangle R_k \\ P_{k+1} &= F_k(P_k - K_k H_k^T P_k) F_k^T + Q \end{aligned} \tag{11}$$

4. Training RBF networks with the fuzzy adaptive kalman filter

In this section we apply the fuzzy adaptive Kalman filter to the training of RBF networks. In general, we can view the optimization of the weight matrix *W* and the prototypes v_j as a weighted least-squares minimization problem, where the error vector is the difference between the RBF outputs and the

target values for those outputs. Consider the RBF network with m inputs, c prototypes, and n outputs. We use y to denote the target vector for the RBF outputs,and $h(\hat{\theta_k})$ to denote the actual outputs at the kth iteration of the optimization algorithm.

$$\begin{aligned} y &= [y_{11} \dots y_{1M} \dots y_{n1} \dots y_{nM}]^T \\ h(\hat{\theta}_k) &= [\hat{y}_{11} \dots \hat{y}_{1M} \dots \hat{y}_{n1} \dots \hat{y}_{nM}]_k^T \end{aligned} \tag{12}$$

Note that the y and $\hat{y}$ vectors each consist of nM elements, where n is the dimension of the RBF output and M is the number of training samples. In order to cast the optimization problem in a form suitable for fuzzy adaptive kalman filter, we let the elements of the weight matrix W and the elements of the prototypes v_j constitute the state of a nonlinear system, and we let the output of the RBF network constitute the output of the nonlinear system to which the fuzzy adaptive kalman filter is applied. The state of the nonlinear system can then be represented as $\theta = [w_1^T \dots w_n^T v_1^T \dots v_c^T]^T$,the vector θ thus consists of all$(n(c+1)+mc)$ of the RBF parameters arranged in a linear array. The nonlinear system model to which the fuzzy adaptive kalman filter can be applied as

$$\begin{aligned} \theta_{k+1} &= \theta_k + \omega_k \\ y_k &= h(\theta_k) + \nu_k \end{aligned} \tag{13}$$

where $h(\theta_k)$ is the RBF network's nonlinear mapping between its parameters and its output, ω_k and ν_k are artificial process noise and measurement noise added to the system model. Now we can apply the fuzzy adaptive kalman filter recursion of Eq.(11).$f(\cdot)$is the identity mapping and y_k the target output of the RBF network.(Note that although y_k written as a function of the Kalman iteration number k, it is actually a constant.) $h(\theta_k)$ is the actual output of the RBF network given the RBF parameters at the kth iteration of the fuzzy Kalman recursion. H_k is the partial derivative of the RBF output with respect to the RBF network parameters at the kth iteration of the fuzzy Kalman recursion. F_k is the identity matrix(a constant even though it is written as a function of k). The Q and R matrices are tuning parameters which can be considered as covariance matrices of the artificial noise processes and , respectively. The partial derivative of the RBF output with respect to the RBF network parameters is given by

$$H_k = \begin{bmatrix} H_v \\ H_w \end{bmatrix} \tag{14}$$

where H_v is given by

$$H_v = \begin{bmatrix} -w_{11}g'_{11}2(x_1 - v_1) & \dots & -w_{11}g'_{m1}2(x_m - v_1) & \dots \\ \vdots & \vdots & \vdots & \vdots \\ -w_{11}g'_{11}2(x_1 - v_1) & \dots & -w_{11}g'_{m1}2(x_m - v_1) & \dots \\ -w_{n1}g'_{11}2(x_1 - v_1) & \dots & -w_{n1}g'_{m1}2(x_m - v_1) & \\ \vdots & \vdots & \vdots & \\ -w_{nc}g'_{1c}2(x_1 - v_c) & \dots & -w_{nc}g'_{mc}2(x_m - v_c) & \end{bmatrix} \tag{15}$$

where w_{ij} is the element in the ith row and jth column of the W weight matrix, $g'_{ij} = g'(\|x_i - v_j\|^2) = -\frac{1}{3}(\|x_i - v_j\|^2 + 1)^{-\frac{4}{3}}$ (where $g(\cdot)$ is the activation function at the hidden layer),x_i is the ith input vector, and v_j is the jth prototype vector. H_v in Eq.(15) is an $mc \times nM$ matrix. H_w is given by

$$H_w = \begin{bmatrix} H & 0 & \dots & 0 \\ 0 & H & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & H \end{bmatrix} \tag{16}$$

Where H is given by

$$H = [h_1 \dots h_M] = \begin{bmatrix} h_{01} & \dots & h_{0M} \\ h_{11} & \dots & h_{1M} \\ \vdots & \vdots & \vdots \\ h_{c1} & \dots & h_{cM} \end{bmatrix} \tag{17}$$

Where $h_{0k} = 1(k = 1, , M), h_{jk} = g(\|x_k - v_j\|^2)(k = 1, , M), (j = 1, , c)$. H_w in Eq.(16) is an $n(c + 1) \times nM$ matrix. And H_k in Eq.(14) is an $[n(c + 1) + mc] \times nM$ Matrix. Now that we have the H_k matrix, we can execute the recursion of Eq.(11), thus using the fuzzy adaptive kalman filter in order to determine the weight matrix W and the prototypes v_j.

5. Simulations

In this section we describe and illustrate the use of fuzzy adaptive kalman filter training for the parameters of an RBF network. We tested the algorithms on the classical Iris classification problem [13]. The networks were trained to respond with the target value $y_{ik} = 1$, and $y_{jk} = 0 \; \forall j \neq i$, when presented with an input vector x_k from the ith category. The reformulated RBF networks were trained using the hidden layer function of Eq.(1). The training algorithms were initialized with prototype vectors randomly selected from the input data, and with the weight W set to 0. In order to test the performance of the algorithm, we compare the results with gradient descend and extended kalman filter from the following aspects: percent of correctly classified (Fig.1), average number of iterations required for learning convergence (Fig. 2) and average CPU time

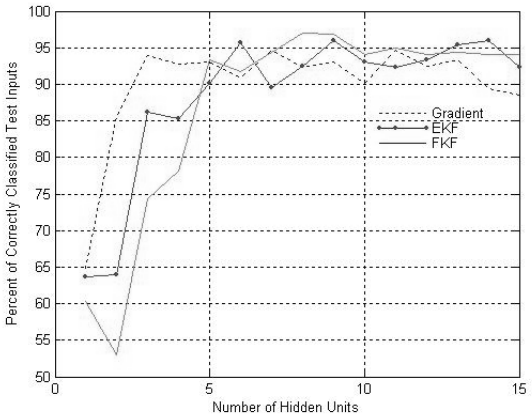

Figure 2. Average percent of correctly classified.

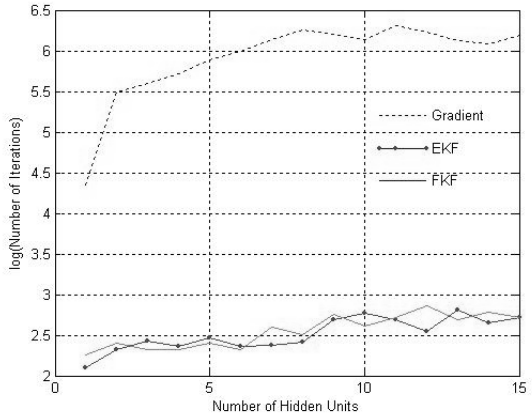

Figure 3. Average number of iterations required for learning convergence.

required for learning convergence (Fig. 3). The gradient descent optimization algorithm was terminated when the error function of Eq.(2) decreased by less than 0.1%. The extend kalman filter parameters of Eq.(9) were initialized with $P_0 = 40I$, $Q = 40I$,and $R = 40I$, where I is the identity matrix of appropriate dimensions. The extended kalman filter recursion was terminated when the error function of Eq.(2) decreased by less than 0.1%. The fuzzy adaptive kalman filter parameters of Eq.(15) were initialized with $P_0 = 40I$, $Q = 40I$,and $R_0 = 40I$,and where I is the identity matrix of appropriate dimensions.The number of hidden units in the RBF network was varied between 1 and 15.

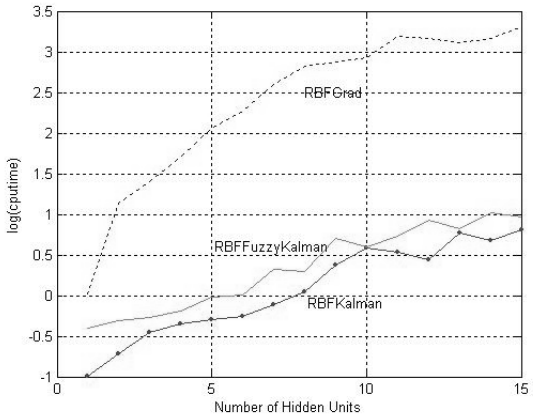

Figure 4. Average CPU time required for learning convergence.

Fig.1. depicts the performance of the RBF network on the test data when the network was trained with gradient descent, extended kalman filter and fuzzy adaptive kalman filter. It can be seen from the figure that, at first, gradient descent training resulted in a better performing network than extended kalman filter and fuzzy adaptive kalman filter. But as the number of hidden units increases, the fuzzy adaptive kalman filter training resulted in better performance than gradient descent training and extended kalman filter training gradually. The RBF network reaches a peak performance of about 97% by the fuzzy adaptive kalman filter training while the best performance by extended kalman filter training and gradient descent training is about 96% and 95% respectively. Fig.2 shows the number of iterations required for convergence for gradient descent training, extended kalman filter training, and fuzzy adaptive kalman filter training. Fig. 3 compares the CPU time required for convergence for the three training methods. (The CPU time is measured in seconds on a Pentium(R) 4 2.93GHz CPU running MATLAB.) With just one or two hidden units, the CPU time is comparable for each of the three methods. But as the number of hidden units increases above one or two, the CPU time is required by gradient descent reaches a fully order of magnitude greater than that required by the extended kalman filter and the fuzzy adaptive kalman filter. The fuzzy adaptive kalman filter requires a little more amount of CPU time than required by the extended kalman filter. This is because that the fuzzy adaptive kalman filter requires a fraction of the computational effort to compute the fuzzy control system.

6. Conclusion and Future Work

This paper demonstrates that how to train RBF network kalman filter. The problem with incomplete a priori knowledge of Q and R matrices is considered. An adaptive Kalman filter approach, based on the filter innovation sequence coupled with fuzzy logic techniques is discussed. A performance comparison is made among gradient descent, extended kalman filter and fuzzy adaptive kalman filter. Simulation results show that fuzzy adaptive kalman filter training converges more quickly than gradient descent training and finds a better solution than extended kalman filter training.

Further research could focus on the application of fuzzy adaptive kalman filter to other type networks. Additional efforts could be focused on applying the technique to large problems.

References

[1]Park,J., Sandberg,J.W., (1991) "Universal approximation using radial basis functions network," Neural Computation, Vol.3, 246-257.

[2]Poggio,T., Girosi , F., (1990) "Networks for approximation and learning,"Proc IEEE vol.78, no.9, 1481-1497.

[3]N. Kraryiannis, (1999) "Reformulated radial basis neural networks trained by gradient descent," IEEE Trans. Neural Networks, 3, 657-671.

[4]J. sum, C. Leung, G. Young, W. Kan, (1999) "On the Kalman Filtering method in Neural network training and pruning," IEEE Trans. Neural Networks 10, 161-166.

[5]Y. Zhang, X. Li, (1999) "A fast U-D factorization-based learning algorithm with applications to nonlinear system modeling and identification," IEEE Trans. Neural Networks 10, 930-938.

[6]Broomhead, D.S., Lowe,D. (1988) "Multivarialble functional interpolation and adaptive networks," Complex Systems, vol. 2, 321-355.

[7]Sanner, R. M., Slotine, J. J. E., (1994) "Gaussian networks for direct adaptive control," IEEE Trans. on Neural Networks, vol. 3, no. 6, 837-863.

[8]Chatzis, V., Bors, A. G., Pitas, I., (1999) "Multimodal decision-level fusion for person authentification," IEEE Trans. on Systems, Man, and Cybernetics, part A: Systems and Humans,vol.29, no.6, 674-680.

[9]R. Duro, J. Reyes, (1999) "Discrete-time backprogagation for training synaptic delay-based artificial neural networks," IEEE Trans. Nerual Networks 10, 779-789.

[10]Dan Simon, (2002) "Training radial basis neural networks with the extended Kalman filter," Neurocomputing. Vol.48, 455-475.

[11]Loebis D., Sutton R. and Chudley J. (2004) "A Fuzzy Kalman Filter Optimized Using a Multiobjective Algorithm for an Enhanced Navigation System of an Autonomous Underwater Vehicle," Proceedings of the Institution of Mechanical Engineers Part M, 218 (M1), 53-69.

[12]Loebis D., Sutton R., Chudley J. and Naeem W. "Adaptive Tuning of a Kalman Filter via Fuzzy Logic for an Intelligent AUV Navigation," System Control Engineering Practice, 12(12), November, 1531-1539.

[13]J.Bezdek, J.Kelle, R.Krishnapuram, L.Kuncheva, H.Pal, (1999) "Will the real Iris data please stand up," IEEE Trans. Fuzzy Systems 7368-369.

[14]B.Anderson,J.Moore,Optimal Filtering,Prentice-Hall,Englewood Cliffs,NJ,1979.